CREATE OR REPLACE TYPE "XNG\_REPORTS"."CLOBAGG\_TYPE" as object(

text clob,

static function ODCIAggregateInitialize(

sctx in out clobagg\_type

)

return number,

member function ODCIAggregateIterate(

self in out clobagg\_type,

value in clob

)

return number,

member function ODCIAggregateTerminate(

self in clobagg\_type,

returnvalue out clob,

flags in number

)

return number,

member function ODCIAggregateMerge(

self in out clobagg\_type,

ctx2 in clobagg\_type

)

return number

);

/

CREATE OR REPLACE TYPE BODY "XNG\_REPORTS"."CLOBAGG\_TYPE"

is

static function ODCIAggregateInitialize(

sctx in out clobagg\_type

)

return number

is

begin

sctx := clobagg\_type(null) ;

return ODCIConst.Success ;

end;

member function ODCIAggregateIterate(

self in out clobagg\_type,

value in clob

)

return number

is

begin

self.text := self.text || value ;

return ODCIConst.Success;

end;

member function ODCIAggregateTerminate(

self in clobagg\_type,

returnvalue out clob,

flags in number

)

return number

is

begin

returnValue := self.text;

return ODCIConst.Success;

end;

member function ODCIAggregateMerge(

self in out clobagg\_type ,

ctx2 in clobagg\_type

)

return number

is

begin

self.text := self.text || ctx2.text;

return ODCIConst.Success;

end;

end;

/

--------------------------------------------------------

-- DDL for Type JUNIPER\_DETAIL\_REC

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."JUNIPER\_DETAIL\_REC" AS OBJECT (

ROWNUMBER NUMBER,

AREA VARCHAR2(50 CHAR),

REGION VARCHAR2(50 CHAR),

LEAF\_DOMAIN\_NAME VARCHAR2(80 CHAR),

HOST\_NAME VARCHAR2(50 CHAR),

DEVICE\_IP VARCHAR2(50 CHAR),

DEVICE\_ID NUMBER,

DEVICE\_VENDOR VARCHAR2(50 CHAR),

DEVICE\_MODEL VARCHAR2(50 CHAR),

DEVICE\_PARTITION VARCHAR2(50 CHAR),

MATCH\_CODE VARCHAR2(20 CHAR),

MATCH\_STATUS VARCHAR2(500 CHAR),

NE\_INST\_ID NUMBER,

DESCR VARCHAR2(100 CHAR),

SITE\_INST\_ID NUMBER,

SITE\_HUM\_ID VARCHAR2(100 CHAR),

TYPE VARCHAR2(50 CHAR),

MODEL VARCHAR2(50 CHAR),

VENDOR VARCHAR2(50 CHAR)

);

/

--------------------------------------------------------

-- DDL for Type JUNIPER\_DETAIL\_TBL

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."JUNIPER\_DETAIL\_TBL" AS TABLE OF JUNIPER\_DETAIL\_REC;

/

--------------------------------------------------------

-- DDL for Type JUNIPER\_SUMMARY\_REC

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."JUNIPER\_SUMMARY\_REC" AS OBJECT (

"AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"TOTAL\_NE" NUMBER,

"DUPLICATE\_NE" NUMBER,

"MATCHED\_NE" NUMBER,

"PERCENT\_MATCHED" NUMBER,

"PASSED\_AUDIT" NUMBER,

"NAMING\_COMPLIANCE" NUMBER

);

/

--------------------------------------------------------

-- DDL for Type JUNIPER\_SUMMARY\_TBL

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."JUNIPER\_SUMMARY\_TBL" AS TABLE OF JUNIPER\_SUMMARY\_REC;

/

--------------------------------------------------------

-- DDL for Type NETSMART\_DETAIL\_REC

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."NETSMART\_DETAIL\_REC" AS OBJECT (

ROWNUMBER NUMBER,

AREA VARCHAR2(50 CHAR),

REGION VARCHAR2(50 CHAR),

LEAF\_DOMAIN\_NAME VARCHAR2(80 CHAR),

TID VARCHAR2(50 CHAR),

TID\_TYPE VARCHAR2(50 CHAR),

LOCATION VARCHAR2(80 CHAR),

CONTACTS VARCHAR2(80 CHAR),

IP\_ADDR VARCHAR2(40 CHAR),

MATCH\_CODE VARCHAR2(20 CHAR),

MATCH\_STATUS VARCHAR2(500 CHAR),

NE\_INST\_ID NUMBER,

DESCR VARCHAR2(100 CHAR),

SITE\_INST\_ID NUMBER,

SITE\_HUM\_ID VARCHAR2(100 CHAR),

TYPE VARCHAR2(50 CHAR),

VENDOR VARCHAR2(50 CHAR),

MODEL VARCHAR2(50 CHAR),

NUM\_XCONNECT NUMBER,

DISCREPANCY NUMBER,

LIVE\_XCONNECTS NUMBER,

OTHER\_XCONNECTS NUMBER,

COMPLIANCE VARCHAR2(10 CHAR)

);

/

--------------------------------------------------------

-- DDL for Type NETSMART\_DETAIL\_TBL

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."NETSMART\_DETAIL\_TBL" AS TABLE OF NETSMART\_DETAIL\_REC;

/

--------------------------------------------------------

-- DDL for Type NETSMART\_SUMMARY\_REC

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."NETSMART\_SUMMARY\_REC" AS OBJECT (

"AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"TOTAL\_NE" NUMBER,

"DUPLICATE\_NE" NUMBER,

"MATCHED\_NE" NUMBER,

"PERCENT\_MATCHED" NUMBER,

"PASSED\_AUDIT" NUMBER,

"NAMING\_COMPLIANCE" NUMBER,

"DISCOVERED\_XCONNECTS" NUMBER,

"LIVE\_XCONNECTS" NUMBER,

"OTHER\_XCONNECTS" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

);

/

--------------------------------------------------------

-- DDL for Type NETSMART\_SUMMARY\_TBL

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."NETSMART\_SUMMARY\_TBL" AS TABLE OF NETSMART\_SUMMARY\_REC;

/

--------------------------------------------------------

-- DDL for Type NETSMART\_XCONNECT\_REC

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."NETSMART\_XCONNECT\_REC" AS OBJECT (

ROWNUMBER NUMBER,

TID VARCHAR2(50 CHAR),

DEVICE\_XCONNECTS VARCHAR2(80 CHAR),

AUDIT\_STATUS VARCHAR2(20 CHAR),

CIRC\_PATH\_INST\_ID NUMBER,

CIRC\_PATH\_HUM\_ID VARCHAR2(100 CHAR),

PATH\_STATUS VARCHAR2(30 CHAR),

BANDWIDTH VARCHAR2(30 CHAR),

COMMENTS VARCHAR2(500 CHAR)

);

/

--------------------------------------------------------

-- DDL for Type NETSMART\_XCONNECT\_TBL

--------------------------------------------------------

CREATE OR REPLACE TYPE "XNG\_REPORTS"."NETSMART\_XCONNECT\_TBL" AS TABLE OF NETSMART\_XCONNECT\_REC;

/

--------------------------------------------------------

-- DDL for Sequence CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN\_SEQ" MINVALUE 0 MAXVALUE 9223372036854775807 INCREMENT BY 1 START WITH 8394 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence CSR\_ID\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."CSR\_ID\_SEQ" MINVALUE 0 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 111328557 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence HOME\_GROWN\_APPS\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."HOME\_GROWN\_APPS\_SEQ" MINVALUE 0 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 40 NOCACHE NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence HOME\_GROWN\_PROCESS\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."HOME\_GROWN\_PROCESS\_SEQ" MINVALUE 0 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 65 NOCACHE NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_SEQ" MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 10474582 NOCACHE NOORDER CYCLE ;

--------------------------------------------------------

-- DDL for Sequence MOTO\_CDMA\_SPANS\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."MOTO\_CDMA\_SPANS\_SEQ" MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 6371234 NOCACHE NOORDER CYCLE ;

--------------------------------------------------------

-- DDL for Sequence MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT\_SEQ" MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 19390989 NOCACHE NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence NETSMART\_EQPT\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."NETSMART\_EQPT\_SEQ" MINVALUE 1 MAXVALUE 9999999999 INCREMENT BY 1 START WITH 249696 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence NETSMART\_SNCNE\_AUDIT\_WK\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT\_WK\_SEQ" MINVALUE 1 MAXVALUE 999999999 INCREMENT BY 1 START WITH 781 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence NETSMART\_SNCNE\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."NETSMART\_SNCNE\_SEQ" MINVALUE 1 MAXVALUE 9999999999 INCREMENT BY 1 START WITH 9860 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence NETSMART\_XCONNECT\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."NETSMART\_XCONNECT\_SEQ" MINVALUE 1 MAXVALUE 9999999999 INCREMENT BY 1 START WITH 723052 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence REPORT\_CONTROL\_1SQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."REPORT\_CONTROL\_1SQ" MINVALUE 1 MAXVALUE 9999999999 INCREMENT BY 1 START WITH 11136 CACHE 50 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence SAM\_EBH\_GEN\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."SAM\_EBH\_GEN\_SEQ" MINVALUE 1 MAXVALUE 9999999999 INCREMENT BY 1 START WITH 261554926 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence SAM\_EBH\_HEALTH\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."SAM\_EBH\_HEALTH\_SEQ" MINVALUE 1 MAXVALUE 9999999999 INCREMENT BY 1 START WITH 44349558 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence SAM\_EBH\_PORT\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."SAM\_EBH\_PORT\_SEQ" MINVALUE 1 MAXVALUE 9999999999 INCREMENT BY 1 START WITH 930031107 CACHE 20 NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Sequence SWITCH\_PATH\_T1\_TESTABILITY\_SEQ

--------------------------------------------------------

CREATE SEQUENCE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_SEQ" MINVALUE 0 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 307548268 NOCACHE NOORDER NOCYCLE ;

--------------------------------------------------------

-- DDL for Table ALL\_PROCESSES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ALL\_PROCESSES"

( "PROCESS\_ID" NUMBER,

"PROCESS\_NAME" VARCHAR2(50 BYTE),

"START\_TIME" DATE,

"END\_TIME" DATE,

"LAST\_FILE\_EXTRACT\_DATE" DATE,

"BASE\_EXEC\_TIME" NUMBER,

"EXEC\_STATUS" NUMBER DEFAULT 3,

"LAST\_SUCCESSFUL\_EXEC\_DATE" DATE,

"FREQUENCY" VARCHAR2(10 BYTE),

"SERVER\_NAME" VARCHAR2(20 BYTE),

"COMMAND" VARCHAR2(2000 BYTE),

"EXEC\_MIN" VARCHAR2(1000 BYTE),

"EXEC\_HOUR" VARCHAR2(1000 BYTE),

"EXEC\_DAY\_OF\_MONTH" VARCHAR2(1000 BYTE),

"EXEC\_MONTH" VARCHAR2(1000 BYTE),

"EXEC\_DAY\_OF\_WEEK" VARCHAR2(1000 BYTE),

"IS\_READY" VARCHAR2(1 BYTE),

"PROCESS\_LOG" VARCHAR2(1000 BYTE),

"OWNER" VARCHAR2(100 BYTE),

"EMAIL\_OWNER" VARCHAR2(100 BYTE),

"CREATE\_DATE" DATE DEFAULT sysdate,

"RUN\_CRON" VARCHAR2(1 BYTE) DEFAULT 'Y',

"LOG\_PREFIX" VARCHAR2(100 BYTE),

"PROCESS\_TYPE" VARCHAR2(1 BYTE),

"EMAIL\_CC" VARCHAR2(1000 BYTE),

"DB\_JOB" CHAR(1 BYTE) DEFAULT 'N',

"LAPSE\_PERIOD" NUMBER,

"RI\_COMMAND" VARCHAR2(2000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."START\_TIME" IS 'Start time of the process';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."END\_TIME" IS 'Time when this process completed execution. If the process is still executing then this time will be null';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."BASE\_EXEC\_TIME" IS 'time in minutes it, typically, takes to execute this process. For jobs less than 5 min +- 50% from this number, where as for jobs longer than that +-10%, should send out notifications.';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."EXEC\_STATUS" IS '0:fail 1:success 2: transient 3: ready';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."LAST\_SUCCESSFUL\_EXEC\_DATE" IS 'Date when the program completed sucessfully';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."EXEC\_MIN" IS '0-59';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."EXEC\_HOUR" IS '0-23';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."EXEC\_DAY\_OF\_MONTH" IS '1-31';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."EXEC\_MONTH" IS '1-12';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."EXEC\_DAY\_OF\_WEEK" IS '0-6 (0=Sunday)';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."IS\_READY" IS 'If the process is ready to be executed';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."PROCESS\_LOG" IS 'The folder where the logs reside';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."EMAIL\_OWNER" IS 'Emails details of the owner of the process';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."RUN\_CRON" IS 'Y or N based on whether we want to activate or deactivate cron';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."PROCESS\_TYPE" IS 'B(Base Process) or D(Dependand Process)';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."EMAIL\_CC" IS 'If someone needs to be notified other than owner';

COMMENT ON COLUMN "XNG\_REPORTS"."ALL\_PROCESSES"."DB\_JOB" IS 'to run stored proc - Y else N';

COMMENT ON TABLE "XNG\_REPORTS"."ALL\_PROCESSES" IS 'Please provide the execution time for only the leaf-level jobs. Other jobs will fire off, upon completion of all dependencies.';

--------------------------------------------------------

-- DDL for Table ANA\_MPLS\_LOAD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ANA\_MPLS\_LOAD"

( "CONTEXT" VARCHAR2(100 BYTE),

"A\_END\_POINT" VARCHAR2(100 BYTE),

"Z\_END\_POINT" VARCHAR2(100 BYTE),

"LINK\_TYPE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ANA\_MPLS\_PHYSICAL\_LINKS\_PARSED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ANA\_MPLS\_PHYSICAL\_LINKS\_PARSED"

( "A\_ROUTER\_NAME" VARCHAR2(11 BYTE),

"A\_END\_PORT" VARCHAR2(89 BYTE),

"A\_PORT\_ACCESS\_ID" VARCHAR2(89 BYTE),

"AP1" VARCHAR2(89 BYTE),

"AP2" VARCHAR2(89 BYTE),

"AP3" VARCHAR2(89 BYTE),

"AP4" VARCHAR2(89 BYTE),

"AP5" VARCHAR2(89 BYTE),

"Z\_ROUTER\_NAME" VARCHAR2(11 BYTE),

"Z\_END\_PORT" VARCHAR2(89 BYTE),

"Z\_PORT\_ACCESS\_ID" VARCHAR2(89 BYTE),

"ZP1" VARCHAR2(89 BYTE),

"ZP2" VARCHAR2(89 BYTE),

"ZP3" VARCHAR2(89 BYTE),

"ZP4" VARCHAR2(89 BYTE),

"ZP5" VARCHAR2(89 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ANA\_MPLS\_RTRS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ANA\_MPLS\_RTRS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"ROUTER\_NAME" VARCHAR2(11 BYTE),

"CLLI" VARCHAR2(8 BYTE),

"DOMAIN" VARCHAR2(7 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ANA\_PHYSICAL\_LINKS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ANA\_PHYSICAL\_LINKS"

( "A\_ROUTER\_NAME" VARCHAR2(50 BYTE),

"A\_END\_PORT" VARCHAR2(20 BYTE),

"Z\_ROUTER\_NAME" VARCHAR2(50 BYTE),

"Z\_END\_PORT" VARCHAR2(20 BYTE),

"LINK\_TYPE" VARCHAR2(50 BYTE) DEFAULT 'Physical Layer',

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table AUDITVIEW\_WINDOW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."AUDITVIEW\_WINDOW"

( "ELEMENT\_TYPE" CHAR(1 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(20 BYTE),

"SUBCLASS\_TYPE" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table AUDIT\_FIX\_IA\_NOT\_IUL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."AUDIT\_FIX\_IA\_NOT\_IUL"

( "INST\_ID" NUMBER(10,0),

"ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"VERSION\_CHGD" NUMBER(6,0),

"DEL\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table AUDIT\_FIX\_IA\_NOT\_IUL08302015

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."AUDIT\_FIX\_IA\_NOT\_IUL08302015"

( "INST\_ID" NUMBER(10,0),

"ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"VERSION\_CHGD" NUMBER(6,0),

"DEL\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table AUDIT\_FIX\_IUL\_NOT\_IA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."AUDIT\_FIX\_IUL\_NOT\_IA"

( "INST\_ID" NUMBER(10,0),

"ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"VERSION\_CHGD" NUMBER(6,0),

"DEL\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table AUDIT\_FIX\_IUL\_NOT\_IA08302015

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."AUDIT\_FIX\_IUL\_NOT\_IA08302015"

( "INST\_ID" NUMBER(10,0),

"ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"VERSION\_CHGD" NUMBER(6,0),

"DEL\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table AUDIT\_STATUS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."AUDIT\_STATUS"

( "INST\_ID" NUMBER,

"AUDIT\_NAME" VARCHAR2(30 BYTE),

"MATCH\_CODE" VARCHAR2(30 BYTE),

"UPDATE\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BILLED\_VS\_ESTIMATED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BILLED\_VS\_ESTIMATED"

( "LEAF\_DOMAIN" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"BTP\_CHRG\_AMT" NUMBER,

"XNG\_NON\_RECUR" NUMBER,

"BAN" VARCHAR2(50 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"BTP\_BILL\_DATE" DATE,

"XNG\_IN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BILLED\_VS\_ESTIMATED\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BILLED\_VS\_ESTIMATED\_WK"

( "LEAF\_DOMAIN" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"BTP\_CHRG\_AMT" NUMBER,

"XNG\_NON\_RECUR" NUMBER,

"BAN" VARCHAR2(50 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"BTP\_BILL\_DATE" DATE,

"XNG\_IN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BRIX\_CSR\_EXTRACT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BRIX\_CSR\_EXTRACT"

( "VERIFIER\_NAME" VARCHAR2(50 BYTE),

"CELL\_SITE\_NAME" VARCHAR2(50 BYTE),

"MAC" VARCHAR2(50 BYTE),

"CST\_MODIFYDATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BRIX\_FD\_SLA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BRIX\_FD\_SLA"

( "FRAME\_DELAY\_IN\_MICRO\_SEC" VARCHAR2(100 BYTE),

"MAC\_ADDRESS" VARCHAR2(30 BYTE),

"VLAN" NUMBER(9,0),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"BRIX\_SWITCH\_NAME" VARCHAR2(200 BYTE),

"NAURU\_ID" NUMBER(20,0),

"BC\_SOURCE" VARCHAR2(20 BYTE),

"ACTUAL\_VLAN" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BRIX\_FD\_SLA\_20141020

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BRIX\_FD\_SLA\_20141020"

( "FRAME\_DELAY\_IN\_MICRO\_SEC" VARCHAR2(100 BYTE),

"MAC\_ADDRESS" VARCHAR2(30 BYTE),

"VLAN" NUMBER,

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"BRIX\_SWITCH\_NAME" VARCHAR2(200 BYTE),

"NAURU\_ID" NUMBER(20,0),

"BC\_SOURCE" VARCHAR2(20 BYTE),

"ACTUAL\_VLAN" NUMBER,

"PBIT" VARCHAR2(5 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BRIX\_FD\_SLA\_20141219

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BRIX\_FD\_SLA\_20141219"

( "FRAME\_DELAY\_IN\_MICRO\_SEC" VARCHAR2(100 BYTE),

"MAC\_ADDRESS" VARCHAR2(30 BYTE),

"VLAN" NUMBER(9,0),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"BRIX\_SWITCH\_NAME" VARCHAR2(200 BYTE),

"NAURU\_ID" NUMBER(20,0),

"BC\_SOURCE" VARCHAR2(20 BYTE),

"ACTUAL\_VLAN" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BRIX\_FD\_SLA\_20150824

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BRIX\_FD\_SLA\_20150824"

( "FRAME\_DELAY\_IN\_MICRO\_SEC" VARCHAR2(100 BYTE),

"MAC\_ADDRESS" VARCHAR2(30 BYTE),

"VLAN" NUMBER(9,0),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"BRIX\_SWITCH\_NAME" VARCHAR2(200 BYTE),

"NAURU\_ID" NUMBER(20,0),

"BC\_SOURCE" VARCHAR2(20 BYTE),

"ACTUAL\_VLAN" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_CELL\_DCG\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_CELL\_DCG\_MAP"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONGNAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"CELL\_NUMBER" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE,

"RADIO\_CONFIG" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_CELL\_DCG\_MAP\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_CELL\_DCG\_MAP\_WRK"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONGNAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"CELL\_NUMBER" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE,

"RADIO\_CONFIG" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_DCG\_SPAN\_VOICE\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_DCG\_SPAN\_VOICE\_MAP"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONGNAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_DCG\_SPAN\_VOICE\_MAP\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_DCG\_SPAN\_VOICE\_MAP\_WRK"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONGNAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_ETHERNET\_INV

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_ETHERNET\_INV"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONGNAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_ETHERNET\_INV\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_ETHERNET\_INV\_WRK"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONGNAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_METRO\_INV

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_METRO\_INV"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONGNAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"BCKHTYPE" VARCHAR2(100 BYTE),

"CMTYPE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_METRO\_INV\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_METRO\_INV\_WRK"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONGNAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"BCKHTYPE" VARCHAR2(100 BYTE),

"CMTYPE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_MTX\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_MTX\_MAP"

( "VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_MTX" VARCHAR2(100 BYTE),

"MTX\_STATUS" VARCHAR2(20 BYTE) DEFAULT 'Live',

"BSM\_STATUS" VARCHAR2(20 BYTE) DEFAULT 'Live'

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BSM\_MTX\_MAP\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BSM\_MTX\_MAP\_WRK"

( "VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_MTX" VARCHAR2(100 BYTE),

"MTX\_STATUS" VARCHAR2(20 BYTE),

"BSM\_STATUS" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(60 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE) DEFAULT TO\_CHAR(SYSDATE, 'yyyymmdd')

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 75497472 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_20160319

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_20160319"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_20160320

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_20160320"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_20160321

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_20160321"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_20160322"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_20160323

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_20160323"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_20160324

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_20160324"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_CIRC\_ID

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_CIRC\_ID"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE) DEFAULT to\_char(sysdate,'yyyymmdd')

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_20160216

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_20160216"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_20160223

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_20160223"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_20160301

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_20160301"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_20160308

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_20160308"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_20160315

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_20160315"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_20160322"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_NEW"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_OLD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_OLD"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(50 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(50 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY\_IND" VARCHAR2(50 BYTE),

"LOGGED\_DATE" VARCHAR2(50 BYTE),

"INVOICE\_SEQUENCE\_ID" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_PROCESS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_PROCESS"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"LOGGED\_DATE" DATE,

"BTP\_AMT" NUMBER,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_PROC\_20160216

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_PROC\_20160216"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"LOGGED\_DATE" DATE,

"BTP\_AMT" NUMBER,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_PROC\_20160223

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_PROC\_20160223"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"LOGGED\_DATE" DATE,

"BTP\_AMT" NUMBER,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_PROC\_20160301

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_PROC\_20160301"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"LOGGED\_DATE" DATE,

"BTP\_AMT" NUMBER,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_PROC\_20160308

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_PROC\_20160308"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"LOGGED\_DATE" DATE,

"BTP\_AMT" NUMBER,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_PROC\_20160315

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_PROC\_20160315"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"LOGGED\_DATE" DATE,

"BTP\_AMT" NUMBER,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_FOR\_UPD\_PROC\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_PROC\_20160322"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"LOGGED\_DATE" DATE,

"BTP\_AMT" NUMBER,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_PP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_PP"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(60 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE) DEFAULT TO\_CHAR(SYSDATE, 'yyyymmdd'),

"ORIG\_COMPANY\_NUM" VARCHAR2(10 BYTE),

"ORIG\_COMPANY\_NAME" VARCHAR2(100 BYTE),

"INVOICE" VARCHAR2(50 BYTE),

"TERM\_LENGTH" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 75497472 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_PROCESS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_PROCESS"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(60 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_AMT" NUMBER,

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHAR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 75497472 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_PROCESS\_20160319

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_PROCESS\_20160319"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(60 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_AMT" NUMBER,

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHAR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_PROCESS\_20160320

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_PROCESS\_20160320"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(60 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_AMT" NUMBER,

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHAR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_PROCESS\_20160321

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_PROCESS\_20160321"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(60 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_AMT" NUMBER,

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHAR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_PROCESS\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_PROCESS\_20160322"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(60 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_AMT" NUMBER,

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHAR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_PROCESS\_20160323

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_PROCESS\_20160323"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(60 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_AMT" NUMBER,

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHAR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_PROCESS\_20160324

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_PROCESS\_20160324"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BTP\_CIRC\_STRIP" VARCHAR2(60 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BTP\_AMT" NUMBER,

"BTP\_BAN\_STRIP" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BTP\_BD" DATE,

"BTP\_BD\_CHAR" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"SEQNO" NUMBER,

"PROCESSED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_WRK"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(60 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE) DEFAULT TO\_CHAR(SYSDATE, 'yyyymmdd')

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 75497472 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_WRK\_20160319

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_WRK\_20160319"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_WRK\_20160320

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_WRK\_20160320"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_WRK\_20160321

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_WRK\_20160321"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_WRK\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_WRK\_20160322"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_WRK\_20160323

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_WRK\_20160323"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_DATA\_WRK\_20160324

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_DATA\_WRK\_20160324"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"SERV\_ESTAB\_DATE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"BILL\_DATE" VARCHAR2(50 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_GI\_MATCH\_RPT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_GI\_MATCH\_RPT"

( "XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(10 BYTE),

"MAX\_CIRC\_DATE" VARCHAR2(10 BYTE),

"SUMM\_TOT" NUMBER,

"SUMM\_AMT" NUMBER,

"MAX\_AUDIT\_DATE" VARCHAR2(10 BYTE),

"STATUS\_DATE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_GI\_MATCH\_RPT\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_GI\_MATCH\_RPT\_WRK"

( "XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(10 BYTE),

"MAX\_CIRC\_DATE" VARCHAR2(10 BYTE),

"SUMM\_TOT" NUMBER,

"SUMM\_AMT" NUMBER,

"MAX\_AUDIT\_DATE" VARCHAR2(10 BYTE),

"STATUS\_DATE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_SEG\_ID\_CLEANUP\_LOG

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_SEG\_ID\_CLEANUP\_LOG"

( "CIRC\_INST\_ID" NUMBER(9,0),

"BTP\_SEG\_ID" VARCHAR2(4000 BYTE),

"VERSION\_CHGD" NUMBER(6,0),

"DELETED\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 92274688 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_20160319

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_20160319"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_20160320

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_20160320"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_20160321

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_20160321"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_20160322"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_20160323

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_20160323"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_20160324

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_20160324"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_HIST

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_HIST"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 92274688 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_HIST\_20160106

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_HIST\_20160106"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_HIST\_20160203

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_HIST\_20160203"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_HIST\_20160303

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_HIST\_20160303"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_HIST\_BKUP101213

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_HIST\_BKUP101213"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 92274688 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUMMARY\_20160319

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUMMARY\_20160319"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUMMARY\_20160320

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUMMARY\_20160320"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUMMARY\_20160321

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUMMARY\_20160321"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUMMARY\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUMMARY\_20160322"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUMMARY\_20160323

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUMMARY\_20160323"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUMMARY\_20160324

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUMMARY\_20160324"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUM\_HST

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUM\_HST"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUM\_HST\_20160106

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUM\_HST\_20160106"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUM\_HST\_20160203

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUM\_HST\_20160203"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUM\_HST\_20160303

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUM\_HST\_20160303"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_SUM\_HST\_HLD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_SUM\_HST\_HLD"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CKT\_COUNT" NUMBER,

"CKTS\_MATCHED" NUMBER,

"CKTS\_EXCP" NUMBER,

"CKTS\_MATCHED\_PCT\_STR" VARCHAR2(15 BYTE),

"CKTS\_MATCHED\_PCT\_NUM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_AUDIT\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_WRK"

( "BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER,

"XNG\_EXTRACT\_DATE" DATE,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BILL\_DATE" DATE,

"BAN\_CUSTOM\_FIELD\_3" VARCHAR2(50 BYTE),

"IN\_SERVICE\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 92274688 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_DOMAIN\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_DOMAIN\_MAP"

( "BTP\_BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"XNG\_DOMAIN\_INST\_ID" NUMBER(9,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 65536 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_AUDIT\_LOG\_FIX

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_AUDIT\_LOG\_FIX"

( "INST\_ID" NUMBER(9,0),

"OLD\_VALUE" VARCHAR2(4000 BYTE),

"NEW\_VALUE" VARCHAR2(4000 BYTE),

"VERSION\_CHGD" NUMBER(6,0),

"CHG\_TS" DATE,

"DELETED\_IND" CHAR(1 BYTE),

"DELETED\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE" DATE,

"CHARGE\_AMOUNT" NUMBER,

"LOGGED\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER,

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"EXCP\_IND" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"XNG\_EXTRACT\_DATE" DATE DEFAULT trunc(sysdate),

"MATCH\_CODE" VARCHAR2(50 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_DETAILS\_20160216

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_20160216"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE" DATE,

"CHARGE\_AMOUNT" NUMBER,

"LOGGED\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER,

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"EXCP\_IND" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"XNG\_EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(50 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_DETAILS\_20160223

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_20160223"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE" DATE,

"CHARGE\_AMOUNT" NUMBER,

"LOGGED\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER,

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"EXCP\_IND" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"XNG\_EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(50 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_DETAILS\_20160301

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_20160301"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE" DATE,

"CHARGE\_AMOUNT" NUMBER,

"LOGGED\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER,

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"EXCP\_IND" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"XNG\_EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(50 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_DETAILS\_20160308

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_20160308"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE" DATE,

"CHARGE\_AMOUNT" NUMBER,

"LOGGED\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER,

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"EXCP\_IND" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"XNG\_EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(50 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_DETAILS\_20160315

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_20160315"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE" DATE,

"CHARGE\_AMOUNT" NUMBER,

"LOGGED\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER,

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"EXCP\_IND" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"XNG\_EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(50 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_DETAILS\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_20160322"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE" DATE,

"CHARGE\_AMOUNT" NUMBER,

"LOGGED\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER,

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"EXCP\_IND" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"XNG\_EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(50 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_DETAILS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_WK"

( "STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE" DATE,

"CHARGE\_AMOUNT" NUMBER,

"LOGGED\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER,

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"EXCP\_IND" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"XNG\_EXTRACT\_DATE" DATE DEFAULT trunc(sysdate),

"MATCH\_CODE" VARCHAR2(50 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_BAN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_REG\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_DISCOVERED" NUMBER,

"UPDATED\_CNT" NUMBER,

"EXCEPTION\_CNT" NUMBER,

"NO\_UPDATE\_NEEDED" NUMBER,

"PERCENT\_UPDATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON COLUMN "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUMMARY"."REGION" IS 'Based on Ban\_category\_1from BTP';

COMMENT ON COLUMN "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUMMARY"."UPDATED\_CNT" IS 'successful updates any/all of the 3 fields';

COMMENT ON COLUMN "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUMMARY"."EXCEPTION\_CNT" IS 'multiples, btp\_only, anything else???';

COMMENT ON COLUMN "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUMMARY"."NO\_UPDATE\_NEEDED" IS 'Uniq match and bill\_date is lesser than in Xng';

COMMENT ON COLUMN "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUMMARY"."PERCENT\_UPDATED" IS 'UPDATED\_CNT/(TOTAL\_DISCOVERED-NO\_UPDATE\_NEEDED)';

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_REG\_SUM\_20160216

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUM\_20160216"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_DISCOVERED" NUMBER,

"UPDATED\_CNT" NUMBER,

"EXCEPTION\_CNT" NUMBER,

"NO\_UPDATE\_NEEDED" NUMBER,

"PERCENT\_UPDATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_REG\_SUM\_20160223

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUM\_20160223"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_DISCOVERED" NUMBER,

"UPDATED\_CNT" NUMBER,

"EXCEPTION\_CNT" NUMBER,

"NO\_UPDATE\_NEEDED" NUMBER,

"PERCENT\_UPDATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_REG\_SUM\_20160301

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUM\_20160301"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_DISCOVERED" NUMBER,

"UPDATED\_CNT" NUMBER,

"EXCEPTION\_CNT" NUMBER,

"NO\_UPDATE\_NEEDED" NUMBER,

"PERCENT\_UPDATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_REG\_SUM\_20160308

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUM\_20160308"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_DISCOVERED" NUMBER,

"UPDATED\_CNT" NUMBER,

"EXCEPTION\_CNT" NUMBER,

"NO\_UPDATE\_NEEDED" NUMBER,

"PERCENT\_UPDATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_REG\_SUM\_20160315

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUM\_20160315"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_DISCOVERED" NUMBER,

"UPDATED\_CNT" NUMBER,

"EXCEPTION\_CNT" NUMBER,

"NO\_UPDATE\_NEEDED" NUMBER,

"PERCENT\_UPDATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_REG\_SUM\_20160322

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_REG\_SUM\_20160322"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_DISCOVERED" NUMBER,

"UPDATED\_CNT" NUMBER,

"EXCEPTION\_CNT" NUMBER,

"NO\_UPDATE\_NEEDED" NUMBER,

"PERCENT\_UPDATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table BTP\_XNG\_UPD\_XNG\_DATA\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_XNG\_DATA\_WK"

( "CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_STRIP\_CIRC\_ID" VARCHAR2(60 BYTE),

"INST\_VER" NUMBER(6,0),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_VENDOR\_CK" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_BAN" VARCHAR2(30 BYTE),

"XNG\_BAN\_CK" VARCHAR2(30 BYTE),

"XNG\_AMT" NUMBER,

"XNG\_AMT\_CK" NUMBER,

"UNIQUELY\_IDENTIFIED" CHAR(1 BYTE),

"BILL\_DATE" VARCHAR2(20 BYTE),

"VAL\_ATTR\_INST\_ID" NUMBER(9,0),

"XNG\_BD\_CHR" VARCHAR2(20 BYTE),

"XNG\_BD" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CAS\_5557\_5564\_5555

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CAS\_5557\_5564\_5555"

( "CIRC\_INST\_ID" NUMBER(10,0),

"VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"ATTR\_VALUE" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CELL\_AVAIL\_ACTIVE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CELL\_AVAIL\_ACTIVE"

( "VSM\_DEVICE\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CIRC\_EBH\_SLA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_EBH\_SLA"

( "VENDOR" VARCHAR2(100 BYTE),

"FRAME\_DELAY" VARCHAR2(10 BYTE),

"FRAME\_DELAY\_VARIATION" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON COLUMN "XNG\_REPORTS"."CIRC\_EBH\_SLA"."FRAME\_DELAY" IS '(in ms)';

--------------------------------------------------------

-- DDL for Table CIRC\_INST\_AUDIT\_TRAIL\_BAD\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_INST\_AUDIT\_TRAIL\_BAD\_DATA"

( "INST\_ID" NUMBER(10,0),

"INST\_VER" NUMBER(6,0),

"AUDIT\_TRAIL\_VERSION\_NUMBER" NUMBER,

"INSERT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CIRC\_PATH\_ATTR\_SETTINGS\_0608

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_PATH\_ATTR\_SETTINGS\_0608"

( "CIRC\_PATH\_INST\_ID" NUMBER(10,0),

"VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"ATTR\_VALUE" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CIRC\_PATH\_ATTR\_SETT\_WL\_0513

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_PATH\_ATTR\_SETT\_WL\_0513"

( "CIRC\_PATH\_INST\_ID" NUMBER(10,0),

"VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"ATTR\_VALUE" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CIRC\_SEG\_NEW\_DECOM\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL"

( "RPT\_ID" NUMBER(10,2) DEFAULT 0,

"DOMAIN\_INST\_ID" NUMBER(10,2),

"DOMAIN" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(10,2),

"SEGMENT\_NAME" VARCHAR2(60 BYTE),

"SEGMENT\_TYPE" VARCHAR2(30 BYTE),

"Z\_SITE\_ID" NUMBER(10,2),

"SITE\_NAME" VARCHAR2(100 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"ACTION\_TYPE" VARCHAR2(10 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"MRC" NUMBER(10,2),

"DISCONNECT\_PON" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_NUM" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_TS" VARCHAR2(30 BYTE),

"DISCONNECT\_TS" VARCHAR2(30 BYTE),

"BILLING\_CODE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT"

( "RPT\_ID" NUMBER(10,2),

"DOMAIN\_INST\_ID" NUMBER(10,2),

"DOMAIN" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(10,2),

"SEGMENT\_NAME" VARCHAR2(60 BYTE),

"SEGMENT\_TYPE" VARCHAR2(30 BYTE),

"Z\_SITE\_ID" NUMBER(10,2),

"SITE\_NAME" VARCHAR2(100 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"ACTION\_TYPE" VARCHAR2(10 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"MRC" NUMBER(10,2),

"DISCONNECT\_PON" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_NUM" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_TS" VARCHAR2(30 BYTE),

"DISCONNECT\_TS" VARCHAR2(30 BYTE),

"BILLING\_CODE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN"

( "RPT\_ID" NUMBER(10,2),

"PROC\_START\_TS" DATE,

"PROC\_END\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CIRC\_SEG\_NEW\_DECOM\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY"

( "PERIOD" VARCHAR2(50 BYTE),

"PERIOD\_TYPE" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"COUNT\_NEW" NUMBER(10,2),

"COUNT\_DECOM" NUMBER(10,2),

"COUNT\_NEW\_TOTAL" NUMBER(10,2) DEFAULT 0,

"COUNT\_NEW\_BY\_TESS" NUMBER(10,2) DEFAULT 0,

"COUNT\_NEW\_NOT\_TESS\_CONTRACT" NUMBER(10,2) DEFAULT 0,

"COUNT\_NEW\_NOT\_TESS\_NO\_CONTRACT" NUMBER(10,2) DEFAULT 0,

"PERCENTAGE\_NEW\_BY\_TESS" NUMBER(10,2) DEFAULT 0.00

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CIRC\_TEST\_RESULTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CIRC\_TEST\_RESULTS"

( "TEST\_ID" NUMBER,

"SERVICE\_NAME" VARCHAR2(50 BYTE),

"VLAN\_NUMBER" NUMBER,

"PORT\_NAME" VARCHAR2(100 BYTE),

"TEST\_TYPE" VARCHAR2(2 BYTE),

"FILE\_NAME" VARCHAR2(100 BYTE),

"TAM\_CUSTOM\_DATA" VARCHAR2(100 BYTE),

"URL" VARCHAR2(200 BYTE),

"TEST\_START\_DATE" VARCHAR2(50 BYTE),

"TEST\_END\_DATE" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"SERVICE\_ID" VARCHAR2(10 BYTE),

"FRAME\_DELAY" VARCHAR2(10 BYTE),

"FRAME\_DELAY\_VARIATION" VARCHAR2(10 BYTE),

"FRAME\_LOSS" VARCHAR2(10 BYTE),

"WL\_OBJECT\_ID" VARCHAR2(30 BYTE),

"WL\_STATUS" VARCHAR2(30 BYTE),

"FRAME\_SIZE" NUMBER,

"BC\_SOURCE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON COLUMN "XNG\_REPORTS"."CIRC\_TEST\_RESULTS"."FRAME\_DELAY" IS 'Value stored in msec';

COMMENT ON COLUMN "XNG\_REPORTS"."CIRC\_TEST\_RESULTS"."FRAME\_DELAY\_VARIATION" IS 'Value stored in msec';

--------------------------------------------------------

-- DDL for Table CLLI\_DOMAIN\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP"

( "CLLI" VARCHAR2(8 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"STATUS" VARCHAR2(10 BYTE) DEFAULT 'Active',

"COMMENTS" VARCHAR2(4000 BYTE),

"LAST\_MOD\_BY" VARCHAR2(50 BYTE),

"LAST\_MOD\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CLLI\_DOMAIN\_MAP\_BAK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP\_BAK"

( "CLLI" VARCHAR2(8 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"STATUS" VARCHAR2(10 BYTE),

"COMMENTS" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table COMP\_EBH\_DETAILS\_BY\_CLLI

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."COMP\_EBH\_DETAILS\_BY\_CLLI"

( "REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_IP" VARCHAR2(40 BYTE),

"XNG\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"NE\_STATUS" VARCHAR2(100 BYTE),

"NE\_SOURCE" VARCHAR2(100 BYTE),

"ISSUES" VARCHAR2(100 BYTE),

"IS\_CRITICAL" CHAR(1 BYTE),

"XNG\_EQUIPMENT" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"CSR\_LIVE\_IN\_XNG" CHAR(1 BYTE),

"GIGE\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"GIGE\_PATH\_INST\_ID" NUMBER(10,0),

"GIGE\_HAS\_ISSUES" VARCHAR2(1 BYTE),

"GIGE\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table COMP\_EBH\_DETAILS\_BY\_CLLI\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."COMP\_EBH\_DETAILS\_BY\_CLLI\_WK"

( "REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_IP" VARCHAR2(40 BYTE),

"XNG\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"NE\_STATUS" VARCHAR2(100 BYTE),

"NE\_SOURCE" VARCHAR2(100 BYTE),

"ISSUES" VARCHAR2(100 BYTE),

"IS\_CRITICAL" CHAR(1 BYTE),

"XNG\_EQUIPMENT" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"CSR\_LIVE\_IN\_XNG" CHAR(1 BYTE),

"GIGE\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"GIGE\_PATH\_INST\_ID" NUMBER(10,0),

"GIGE\_HAS\_ISSUES" VARCHAR2(1 BYTE),

"GIGE\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table COMP\_EBH\_SUMMARY\_BY\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."COMP\_EBH\_SUMMARY\_BY\_REGION"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_CSR" NUMBER,

"BOTH\_LIVE\_CSR" NUMBER,

"DUP\_LIVE\_CSR" NUMBER,

"LIVE\_CSR\_W\_LIVE\_GIGE" NUMBER,

"LIVE\_CSR\_W\_NLIVE\_GIGE" NUMBER,

"LIVE\_CSR\_NO\_GIGE" NUMBER,

"BOTH\_NLIVE\_CSR" NUMBER,

"LIVE\_CSR\_W\_EBH\_PATHSET" NUMBER,

"DEN\_FOR\_COMPLIANCE\_PER" NUMBER,

"COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table COMP\_EBH\_SUMMARY\_BY\_REGION\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."COMP\_EBH\_SUMMARY\_BY\_REGION\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_CSR" NUMBER,

"BOTH\_LIVE\_CSR" NUMBER,

"DUP\_LIVE\_CSR" NUMBER,

"LIVE\_CSR\_W\_LIVE\_GIGE" NUMBER,

"LIVE\_CSR\_W\_NLIVE\_GIGE" NUMBER,

"LIVE\_CSR\_NO\_GIGE" NUMBER,

"BOTH\_NLIVE\_CSR" NUMBER,

"LIVE\_CSR\_W\_EBH\_PATHSET" NUMBER,

"DEN\_FOR\_COMPLIANCE\_PER" NUMBER,

"COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table COPY\_WK\_TABLES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."COPY\_WK\_TABLES"

( "PROCESS\_ID" NUMBER,

"SEQ\_NO" NUMBER,

"SCHEMA\_NAME" VARCHAR2(100 BYTE),

"WK\_TABLE\_NAME" VARCHAR2(100 BYTE),

"PRD\_TABLE\_NAME" VARCHAR2(100 BYTE),

"LAST\_COPY\_DATE" DATE,

"WHERE\_CLAUSE" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_AUDIT\_ISSUE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE"

( "CSR\_ISSUE\_ID" NUMBER(\*,0),

"CSR\_ID" NUMBER,

"XNG\_EQUIP\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_AUDIT\_ISSUE\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE\_WK"

( "CSR\_ISSUE\_ID" NUMBER(\*,0),

"CSR\_ID" NUMBER,

"XNG\_EQUIP\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_DEVICES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_DEVICES"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_IP" VARCHAR2(40 BYTE),

"CSR\_CLASS" VARCHAR2(100 BYTE),

"LOAD\_DATE" DATE,

"CSR\_ID" NUMBER,

"NE\_STATUS" VARCHAR2(100 BYTE),

"NE\_SOURCE" VARCHAR2(100 BYTE),

"COMMON\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_DEVICES\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_DEVICES\_WK"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_IP" VARCHAR2(40 BYTE),

"CSR\_CLASS" VARCHAR2(100 BYTE),

"LOAD\_DATE" DATE DEFAULT sysdate,

"CSR\_ID" NUMBER,

"NE\_STATUS" VARCHAR2(100 BYTE),

"NE\_SOURCE" VARCHAR2(100 BYTE),

"COMMON\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_DEVICE\_AUDIT\_ISSUES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_DEVICE\_AUDIT\_ISSUES"

( "CSR\_DEVICE\_NAME" VARCHAR2(50 BYTE),

"CSR\_ISSUE\_ID" NUMBER,

"CSR\_VENDOR" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_DEVICE\_AUDIT\_ISSUES\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_DEVICE\_AUDIT\_ISSUES\_WK"

( "CSR\_DEVICE\_NAME" VARCHAR2(50 BYTE),

"CSR\_ISSUE\_ID" NUMBER,

"CSR\_VENDOR" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_ISSUE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_ISSUE"

( "CSR\_ISSUE\_ID" NUMBER(\*,0),

"DESCRIPTION" VARCHAR2(100 BYTE),

"IS\_CRITICAL" CHAR(1 BYTE) DEFAULT 'N',

"COMMENTS" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_MISSING\_IN\_HPOV

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_MISSING\_IN\_HPOV"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_NE\_CSR\_PER\_REGION" NUMBER,

"TOT\_CSR\_MATCHED\_PER\_REGION" NUMBER,

"TOT\_XO\_LIVE\_CSR\_PER\_REGION" NUMBER,

"PCT\_LIVE\_CSR\_MISSING\_IN\_HPOV" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_MISSING\_IN\_HPOV\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_MISSING\_IN\_HPOV\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_NE\_CSR\_PER\_REGION" NUMBER,

"TOT\_CSR\_MATCHED\_PER\_REGION" NUMBER,

"TOT\_XO\_LIVE\_CSR\_PER\_REGION" NUMBER,

"PCT\_LIVE\_CSR\_MISSING\_IN\_HPOV" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_SUMMARY\_BY\_REGIN\_BAK080913

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_SUMMARY\_BY\_REGIN\_BAK080913"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_CSR" NUMBER,

"BOTH\_LIVE\_CSR" NUMBER,

"BOTH\_NLIVE\_CSR" NUMBER,

"DUP\_LIVE\_CSR" NUMBER,

"DEN\_FOR\_COMPER" NUMBER,

"COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_SUMMARY\_BY\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_SUMMARY\_BY\_REGION"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_CSR" NUMBER,

"BOTH\_LIVE\_CSR" NUMBER,

"BOTH\_NLIVE\_CSR" NUMBER,

"DUP\_LIVE\_CSR" NUMBER,

"DEN\_FOR\_COMPER" NUMBER,

"COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_SUMMARY\_BY\_REGION\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_SUMMARY\_BY\_REGION\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_CSR" NUMBER,

"BOTH\_LIVE\_CSR" NUMBER,

"BOTH\_NLIVE\_CSR" NUMBER,

"DUP\_LIVE\_CSR" NUMBER,

"DEN\_FOR\_COMPER" NUMBER,

"COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_SUMMARY\_BY\_REGION\_WK\_OLD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_SUMMARY\_BY\_REGION\_WK\_OLD"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_CSR" NUMBER,

"BOTH\_LIVE\_CSR" NUMBER,

"DUP\_LIVE\_CSR" NUMBER,

"LIVE\_CSR\_W\_LIVE\_GIGE" NUMBER,

"LIVE\_CSR\_W\_NLIVE\_GIGE" NUMBER,

"LIVE\_CSR\_NO\_GIGE" NUMBER,

"BOTH\_NLIVE\_CSR" NUMBER,

"LIVE\_CSR\_W\_EBH\_PATHSET" NUMBER,

"DEN\_FOR\_COMPLIANCE\_PER" NUMBER,

"COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_VLAN\_AUDIT\_ISSUES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_VLAN\_AUDIT\_ISSUES"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"CSR\_ISSUE\_ID" NUMBER,

"CSR\_VENDOR" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table CSR\_VLAN\_AUDIT\_ISSUES\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."CSR\_VLAN\_AUDIT\_ISSUES\_WK"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"CSR\_ISSUE\_ID" NUMBER,

"CSR\_VENDOR" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DACS\_METROWATCH\_LOOPBACKS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DACS\_METROWATCH\_LOOPBACKS"

( "NE\_ID" NUMBER,

"NE\_SID" VARCHAR2(30 BYTE),

"NE\_NAME" VARCHAR2(30 BYTE),

"NE\_IPADDRESS" VARCHAR2(15 BYTE),

"LAST\_CONFIG\_RESYNC\_TIME" VARCHAR2(50 BYTE),

"LAST\_FAULT\_RESYNC\_TIME" VARCHAR2(50 BYTE),

"SHELFAID" VARCHAR2(50 BYTE),

"TP\_AID" VARCHAR2(50 BYTE),

"TP\_RAWPST" VARCHAR2(20 BYTE),

"TP\_TPTYPE" VARCHAR2(20 BYTE),

"TP\_PARTITION" VARCHAR2(20 BYTE),

"TP\_LPBKCMD" VARCHAR2(20 BYTE),

"TP\_LPBKRSP" VARCHAR2(20 BYTE),

"TP\_PMAID" VARCHAR2(50 BYTE),

"TP\_SDF" NUMBER,

"TP\_FACAID" VARCHAR2(50 BYTE),

"TP\_FACTYPE" VARCHAR2(20 BYTE),

"TP\_JOINAID" VARCHAR2(50 BYTE),

"CRS\_FROM\_SHELF\_AID" VARCHAR2(50 BYTE),

"CRS\_TO\_SHELF\_AID" VARCHAR2(50 BYTE),

"CRS\_FROM\_TP\_AID" VARCHAR2(50 BYTE),

"CRS\_TO\_TP\_AID" VARCHAR2(50 BYTE),

"CRS\_XCTYPE" VARCHAR2(20 BYTE),

"CRS\_XCSTATE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DACS\_QT\_TEST\_HEADS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DACS\_QT\_TEST\_HEADS"

( "QT\_NAME" VARCHAR2(100 BYTE),

"QT\_DISPLAY\_NAME" VARCHAR2(100 BYTE),

"QT\_IP" VARCHAR2(30 BYTE),

"TAM\_TYPE" VARCHAR2(100 BYTE),

"MSPP\_TAC\_PORT" VARCHAR2(100 BYTE),

"MSPP\_IP" VARCHAR2(30 BYTE),

"MSPP\_TCP\_PORT" VARCHAR2(100 BYTE),

"MSPP\_USER\_ACCOUNT" VARCHAR2(100 BYTE),

"DCS\_TYPE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DACS\_QT\_TEST\_HEADS\_TEMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DACS\_QT\_TEST\_HEADS\_TEMP"

( "TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"DCS\_TYPE" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DACS\_TEST\_HEADS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DACS\_TEST\_HEADS"

( "TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"DCS\_TYPE" VARCHAR2(100 BYTE),

"ID" NUMBER,

"TEST\_HEAD" NUMBER,

"T\_ACTIVE" NUMBER,

"NT\_ACTIVE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DACS\_TEST\_HEADS\_TEMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DACS\_TEST\_HEADS\_TEMP"

( "TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"DCS\_TYPE" VARCHAR2(100 BYTE),

"ID" NUMBER,

"TEST\_HEAD" NUMBER,

"T\_ACTIVE" NUMBER,

"NT\_ACTIVE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DARK\_FIBER\_STATUS\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_DEL"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"SITERRA\_TYPE" VARCHAR2(3100 BYTE),

"SITERRA\_TYPE\_FR" CHAR(7 BYTE),

"SITE\_ID" VARCHAR2(30 BYTE),

"PS\_ID" VARCHAR2(3100 BYTE),

"NWF\_ID" VARCHAR2(3100 BYTE),

"SITE\_STATUS" VARCHAR2(20 BYTE),

"SITE\_NAME" VARCHAR2(100 BYTE),

"CKT\_ID" NUMBER(10,0),

"CKT\_NAME" VARCHAR2(60 BYTE),

"CKT\_VENDOR" VARCHAR2(30 BYTE),

"CKT\_STATUS" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"DF\_PATH\_ID" NUMBER(10,0),

"DF\_PATH\_NAME" VARCHAR2(100 BYTE),

"DF\_PATH\_REV" NUMBER(6,0),

"DF\_PATH\_STATUS" VARCHAR2(20 BYTE),

"DF\_PATH\_TYPE" VARCHAR2(30 BYTE),

"RF\_PATH\_ID" NUMBER,

"RF\_PATH\_NAME" VARCHAR2(100 BYTE),

"RF\_PATH\_REV" NUMBER,

"RF\_PATH\_TYPE" VARCHAR2(30 BYTE),

"RF\_PATH\_STATUS" VARCHAR2(20 BYTE),

"RF\_PATH\_LVL" NUMBER,

"RF\_EQ\_ID" NUMBER,

"RF\_EQ\_TYPE" VARCHAR2(30 BYTE),

"RF\_EQ\_NAME" VARCHAR2(100 BYTE),

"RF\_EQ\_PORT" VARCHAR2(81 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DARK\_FIBER\_STATUS\_NEW\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_NEW\_DEL"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"SITE\_TYPE" VARCHAR2(3100 BYTE),

"SITERRA\_TYPE" VARCHAR2(3100 BYTE),

"SITERRA\_TYPE\_FR" CHAR(7 BYTE),

"SITE\_ID" VARCHAR2(30 BYTE),

"PS\_ID" VARCHAR2(3100 BYTE),

"NWF\_ID" VARCHAR2(3100 BYTE),

"SITE\_STATUS" VARCHAR2(20 BYTE),

"SITE\_NAME" VARCHAR2(100 BYTE),

"CKT\_ID" NUMBER(10,0),

"CKT\_NAME" VARCHAR2(60 BYTE),

"CKT\_VENDOR" VARCHAR2(30 BYTE),

"CKT\_STATUS" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"DF\_PATH\_ID" NUMBER(10,0),

"DF\_PATH\_NAME" VARCHAR2(100 BYTE),

"DF\_PATH\_REV" NUMBER(6,0),

"DF\_PATH\_STATUS" VARCHAR2(20 BYTE),

"DF\_PATH\_TYPE" VARCHAR2(30 BYTE),

"RF\_EQUIP\_ID" NUMBER(10,0),

"RF\_EQUIP\_TYPE" VARCHAR2(30 BYTE),

"RF\_EQUIP\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DARK\_FIBER\_STATUS\_WK\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_WK\_DEL"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"SITERRA\_TYPE" VARCHAR2(3100 BYTE),

"SITERRA\_TYPE\_FR" CHAR(7 BYTE),

"SITE\_ID" VARCHAR2(30 BYTE),

"PS\_ID" VARCHAR2(3100 BYTE),

"NWF\_ID" VARCHAR2(3100 BYTE),

"SITE\_STATUS" VARCHAR2(20 BYTE),

"SITE\_NAME" VARCHAR2(100 BYTE),

"CKT\_ID" NUMBER(10,0),

"CKT\_NAME" VARCHAR2(60 BYTE),

"CKT\_VENDOR" VARCHAR2(30 BYTE),

"CKT\_STATUS" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"DF\_PATH\_ID" NUMBER(10,0),

"DF\_PATH\_NAME" VARCHAR2(100 BYTE),

"DF\_PATH\_REV" NUMBER(6,0),

"DF\_PATH\_STATUS" VARCHAR2(20 BYTE),

"DF\_PATH\_TYPE" VARCHAR2(30 BYTE),

"RF\_PATH\_ID" NUMBER,

"RF\_PATH\_NAME" VARCHAR2(100 BYTE),

"RF\_PATH\_REV" NUMBER,

"RF\_PATH\_TYPE" VARCHAR2(30 BYTE),

"RF\_PATH\_STATUS" VARCHAR2(20 BYTE),

"RF\_PATH\_LVL" NUMBER,

"RF\_EQ\_ID" NUMBER,

"RF\_EQ\_TYPE" VARCHAR2(30 BYTE),

"RF\_EQ\_NAME" VARCHAR2(100 BYTE),

"RF\_EQ\_PORT" VARCHAR2(81 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DECOMM\_REQD\_FIELDS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DECOMM\_REQD\_FIELDS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"BAN" VARCHAR2(50 BYTE),

"DISCONNECT\_PON" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_DATE" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_NUMBER" VARCHAR2(3100 BYTE),

"DISCONNECT\_DATE" VARCHAR2(3100 BYTE),

"RPON" VARCHAR2(3100 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"WO\_INST\_ID" NUMBER(9,0),

"WO\_NAME" VARCHAR2(20 BYTE),

"TASK\_INST\_ID" NUMBER(9,0),

"TASK\_NAME" VARCHAR2(60 BYTE),

"QUEUE\_INST\_ID" NUMBER(9,0),

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BTP\_EXTRACT\_DATE" DATE,

"RUN\_DATE" DATE,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DEL\_ENODEBS\_TYPES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DEL\_ENODEBS\_TYPES"

( "EQUIP\_INST\_ID" NUMBER(10,0),

"TYPE" VARCHAR2(30 BYTE),

"VENDOR" VARCHAR2(40 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"REV" VARCHAR2(30 BYTE),

"SW\_REV" VARCHAR2(30 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"PARENT\_EQ\_INST\_ID" NUMBER(10,0),

"STATUS" VARCHAR2(20 BYTE),

"EQ\_CLASS" CHAR(1 BYTE),

"DIM\_HEIGHT" NUMBER(10,3),

"DIM\_WIDTH" NUMBER(10,3),

"DIM\_DEPTH" NUMBER(10,3),

"DIM\_DIST\_TO\_BASE" NUMBER(10,3),

"DIM\_DIST\_TO\_LEFT" NUMBER(10,3),

"DIM\_DIST\_TO\_FRONT" NUMBER(10,3),

"BATCH\_NO" VARCHAR2(20 BYTE),

"BAR\_CODE" VARCHAR2(30 BYTE),

"LINEUP" VARCHAR2(12 BYTE),

"FRAME" VARCHAR2(12 BYTE),

"SHELF" VARCHAR2(12 BYTE),

"NMS\_EMS" VARCHAR2(30 BYTE),

"TARGET\_ID" VARCHAR2(40 BYTE),

"PPM" VARCHAR2(12 BYTE),

"PPMM" VARCHAR2(12 BYTE),

"ORDER\_NUM" VARCHAR2(50 BYTE),

"ORDERED" DATE,

"DUE" DATE,

"INSTALLED" DATE,

"SCHED\_DATE" DATE,

"IN\_SERVICE" DATE,

"DECOMMISSION" DATE,

"MARKET" VARCHAR2(30 BYTE),

"SERIAL\_NO" VARCHAR2(30 BYTE),

"PURCHASE\_PRICE" NUMBER(10,2),

"PURCHASE\_DATE" DATE,

"ASSET\_LIFE" NUMBER(12,2),

"TG\_MEM\_ACTIVE" CHAR(1 BYTE),

"LAST\_TG\_MEM\_UPD" DATE,

"LAST\_TG\_MEM\_REVIEWED" DATE,

"COMMENTS" VARCHAR2(4000 BYTE),

"INST\_VER" NUMBER(6,0),

"LAST\_MOD\_TS" DATE,

"LAST\_MOD\_BY" VARCHAR2(50 BYTE),

"AUTOMATION" VARCHAR2(25 BYTE),

"PLANNING\_ENABLED" CHAR(1 BYTE),

"WARN\_USER\_MOD" CHAR(1 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"CLEI" VARCHAR2(50 BYTE),

"POINT\_CODE" VARCHAR2(12 BYTE),

"EQ\_TPLT\_INST\_ID" NUMBER(10,0),

"CUST\_INST\_ID" NUMBER(10,0),

"NPA" VARCHAR2(50 BYTE),

"ADM\_CO\_INST\_ID" NUMBER(10,0),

"AID" VARCHAR2(1000 BYTE),

"HECIG" VARCHAR2(50 BYTE),

"UNIT\_OF\_MEASURE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DF\_STATUS\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"LOC\_CAT" VARCHAR2(3100 BYTE),

"PS\_ID" VARCHAR2(3100 BYTE),

"NWF\_ID" VARCHAR2(3100 BYTE),

"SITE\_STATUS" VARCHAR2(20 BYTE),

"SITE\_NAME" VARCHAR2(100 BYTE),

"CKT\_ID" NUMBER(10,0),

"CKT\_NAME" VARCHAR2(60 BYTE),

"CKT\_VENDOR" VARCHAR2(30 BYTE),

"CKT\_STATUS" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DF\_STATUS\_DETAILS\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_NEW"

( "MARKET" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"LOC\_CAT" VARCHAR2(3100 BYTE),

"PS\_ID" VARCHAR2(3100 BYTE),

"NWF\_ID" VARCHAR2(3100 BYTE),

"SITE\_STATUS" VARCHAR2(20 BYTE),

"SITE\_NAME" VARCHAR2(100 BYTE),

"CKT\_ID" NUMBER(10,0),

"CKT\_NAME" VARCHAR2(60 BYTE),

"CKT\_VENDOR" VARCHAR2(30 BYTE),

"CKT\_STATUS" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE),

"TERRITORY" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DF\_STATUS\_DETAILS\_NEW\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_NEW\_WK"

( "MARKET" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"LOC\_CAT" VARCHAR2(3100 BYTE),

"PS\_ID" VARCHAR2(3100 BYTE),

"NWF\_ID" VARCHAR2(3100 BYTE),

"SITE\_STATUS" VARCHAR2(20 BYTE),

"SITE\_NAME" VARCHAR2(100 BYTE),

"CKT\_ID" NUMBER(10,0),

"CKT\_NAME" VARCHAR2(60 BYTE),

"CKT\_VENDOR" VARCHAR2(30 BYTE),

"CKT\_STATUS" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE),

"TERRITORY" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DF\_STATUS\_DETAILS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"LOC\_CAT" VARCHAR2(3100 BYTE),

"PS\_ID" VARCHAR2(3100 BYTE),

"NWF\_ID" VARCHAR2(3100 BYTE),

"SITE\_STATUS" VARCHAR2(20 BYTE),

"SITE\_NAME" VARCHAR2(100 BYTE),

"CKT\_ID" NUMBER(10,0),

"CKT\_NAME" VARCHAR2(60 BYTE),

"CKT\_VENDOR" VARCHAR2(30 BYTE),

"CKT\_STATUS" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DF\_STATUS\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DF\_STATUS\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"INBUILDING" NUMBER,

"MACRO" NUMBER,

"MICRO" NUMBER,

"NODE" NUMBER,

"VENUE" NUMBER,

"BLANK" NUMBER,

"TOTAL" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DF\_STATUS\_SUMMARY\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DF\_STATUS\_SUMMARY\_NEW"

( "TERRITORY" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"INBUILDING" NUMBER,

"MACRO" NUMBER,

"MICRO" NUMBER,

"NODE" NUMBER,

"VENUE" NUMBER,

"BLANK" NUMBER,

"TOTAL" NUMBER,

"SUB\_MARKET" VARCHAR2(50 BYTE),

"TERRITORY\_MARKET\_SUB" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DF\_STATUS\_SUMMARY\_NEW\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DF\_STATUS\_SUMMARY\_NEW\_WK"

( "TERRITORY" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"INBUILDING" NUMBER,

"MACRO" NUMBER,

"MICRO" NUMBER,

"NODE" NUMBER,

"VENUE" NUMBER,

"BLANK" NUMBER,

"TOTAL" NUMBER,

"SUB\_MARKET" VARCHAR2(50 BYTE),

"TERRITORY\_MARKET\_SUB" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DF\_STATUS\_SUMMARY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DF\_STATUS\_SUMMARY\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"INBUILDING" NUMBER,

"MACRO" NUMBER,

"MICRO" NUMBER,

"NODE" NUMBER,

"VENUE" NUMBER,

"BLANK" NUMBER,

"TOTAL" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DISCONNECT\_AGING

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DISCONNECT\_AGING"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"WO\_INST\_ID" NUMBER(9,0),

"WO\_NAME" VARCHAR2(20 BYTE),

"TASK\_INST\_ID" NUMBER(9,0),

"TASK\_NAME" VARCHAR2(60 BYTE),

"QUEUE\_INST\_ID" NUMBER(9,0),

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"A\_SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"Z\_SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"RUN\_DATE" DATE,

"DECOMMISSION\_DATE" DATE,

"RPON" VARCHAR2(3100 BYTE),

"DISCONNECT\_PON" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_DATE" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_NUMBER" VARCHAR2(3100 BYTE),

"DISCONNECT\_DATE\_UDA" DATE,

"LAST\_STATUS\_CHANGE" DATE,

"FOUND\_METHOD" VARCHAR2(20 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BILL\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DISCONNECT\_AGING\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DISCONNECT\_AGING\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"WO\_INST\_ID" NUMBER(9,0),

"WO\_NAME" VARCHAR2(20 BYTE),

"TASK\_INST\_ID" NUMBER(9,0),

"TASK\_NAME" VARCHAR2(60 BYTE),

"QUEUE\_INST\_ID" NUMBER(9,0),

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"A\_SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"Z\_SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"RPON" VARCHAR2(3100 BYTE),

"DISCONNECT\_PON" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_DATE" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_NUMBER" VARCHAR2(3100 BYTE),

"DISCONNECT\_DATE\_UDA" DATE,

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"BTP\_XNG\_MATCH" VARCHAR2(1 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"BILL\_DATE" DATE,

"AUDIT\_DATE" DATE,

"CIRC\_DECOMM\_DATE" DATE,

"TASK\_CK\_TYPE" NUMBER,

"TASK\_COMPL\_DATE" DATE,

"WO\_COMPL\_DATE" DATE,

"STATCK" VARCHAR2(20 BYTE),

"MATCH\_CODE" VARCHAR2(20 BYTE),

"MATCH\_STATUS" VARCHAR2(50 BYTE),

"INSERT\_DATE" DATE,

"FIRST\_MOD\_DATE" DATE,

"TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 16777216 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DISCONNECT\_REASON\_CODE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"DISC\_REASON\_CODE" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_DATE" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_NUMBER" VARCHAR2(3100 BYTE),

"DISCONNECT\_DATE" VARCHAR2(3100 BYTE),

"EARLY\_TERMINATION" VARCHAR2(3100 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"SITE\_A" VARCHAR2(100 BYTE),

"SITE\_Z" VARCHAR2(100 BYTE),

"BAN" VARCHAR2(50 BYTE),

"DISCONNECT\_PON" VARCHAR2(3100 BYTE),

"RPON" VARCHAR2(3100 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"RUN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table DISCONNECT\_REASON\_CODE\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"DISC\_REASON\_CODE" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_DATE" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_NUMBER" VARCHAR2(3100 BYTE),

"DISCONNECT\_DATE" VARCHAR2(3100 BYTE),

"EARLY\_TERMINATION" VARCHAR2(3100 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"SITE\_A" VARCHAR2(100 BYTE),

"SITE\_Z" VARCHAR2(100 BYTE),

"BAN" VARCHAR2(50 BYTE),

"DISCONNECT\_PON" VARCHAR2(3100 BYTE),

"RPON" VARCHAR2(3100 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BTP\_EXTRACT\_DATE" DATE,

"RUN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EBH\_CKTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EBH\_CKTS"

( "DOMAIN\_INST\_ID" NUMBER(9,0),

"Z\_SITE\_ID" NUMBER(9,0),

"CIRC\_INST\_ID" NUMBER(9,0),

"IN\_SERVICE" DATE,

"MRC" NUMBER(10,2),

"EBH\_CKT\_NAME" VARCHAR2(120 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EBH\_DISCONNECT\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EBH\_DISCONNECT\_DETAILS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"DF\_SEG\_ID" NUMBER(10,0),

"DF\_HAS\_BILL" CHAR(1 BYTE),

"NDF\_SEG\_ID" NUMBER(10,0),

"EBH\_DISCONNECT\_DATE" DATE,

"DF\_IN\_SERVICE\_DATE" DATE,

"DURATION" NUMBER,

"DF\_SITE\_ID" NUMBER(10,0),

"AUDIT\_DATE" DATE,

"EBH\_BILL\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EBH\_DISCONNECT\_DETAILS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EBH\_DISCONNECT\_DETAILS\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"DF\_SEG\_ID" NUMBER(10,0),

"DF\_HAS\_BILL" CHAR(1 BYTE),

"NDF\_SEG\_ID" NUMBER(10,0),

"EBH\_DISCONNECT\_DATE" DATE,

"DF\_IN\_SERVICE\_DATE" DATE,

"DURATION" NUMBER DEFAULT 0,

"DF\_SITE\_ID" NUMBER(10,0),

"AUDIT\_DATE" DATE,

"EBH\_BILL\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON COLUMN "XNG\_REPORTS"."EBH\_DISCONNECT\_DETAILS\_WK"."DURATION" IS 'no billing, then duration is for disconnectdate, otherwise inservicedate';

--------------------------------------------------------

-- DDL for Table EBH\_DISCONNECT\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EBH\_DISCONNECT\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NOINSERV" NUMBER,

"DUR0TO30" NUMBER,

"DUR30TO60" NUMBER,

"DUR60TO90" NUMBER,

"DUR90TO120" NUMBER,

"DUR120TO180" NUMBER,

"DUROVER180" NUMBER,

"CTREGIONTOTAL" NUMBER,

"AMTNOINSERV" NUMBER,

"AMTDUR0TO30" NUMBER,

"AMTDUR30TO60" NUMBER,

"AMTDUR60TO90" NUMBER,

"AMTDUR90TO120" NUMBER,

"AMTDUR120TO180" NUMBER,

"AMTDUROVER180" NUMBER,

"AMTREGIONTOTAL" NUMBER,

"FLAG" VARCHAR2(50 BYTE),

"NODISCONNECT" NUMBER,

"AMTNODISCONNECT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EBH\_DISCONNECT\_SUMMARY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EBH\_DISCONNECT\_SUMMARY\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NOINSERV" NUMBER,

"DUR0TO30" NUMBER,

"DUR30TO60" NUMBER,

"DUR60TO90" NUMBER,

"DUR90TO120" NUMBER,

"DUR120TO180" NUMBER,

"DUROVER180" NUMBER,

"CTREGIONTOTAL" NUMBER,

"AMTNOINSERV" NUMBER,

"AMTDUR0TO30" NUMBER,

"AMTDUR30TO60" NUMBER,

"AMTDUR60TO90" NUMBER,

"AMTDUR90TO120" NUMBER,

"AMTDUR120TO180" NUMBER,

"AMTDUROVER180" NUMBER,

"AMTREGIONTOTAL" NUMBER,

"FLAG" VARCHAR2(50 BYTE),

"NODISCONNECT" NUMBER,

"AMTNODISCONNECT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EBH\_LR\_TMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EBH\_LR\_TMP"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NOINSERV" NUMBER,

"DUR0TO30" NUMBER,

"DUR30TO60" NUMBER,

"DUR60TO90" NUMBER,

"DUR90TO120" NUMBER,

"DUR120TO180" NUMBER,

"DUROVER180" NUMBER,

"CTREGIONTOTAL" NUMBER,

"AMTNOINSERV" NUMBER,

"AMTDUR0TO30" NUMBER,

"AMTDUR30TO60" NUMBER,

"AMTDUR60TO90" NUMBER,

"AMTDUR90TO120" NUMBER,

"AMTDUR120TO180" NUMBER,

"AMTDUROVER180" NUMBER,

"AMTREGIONTOTAL" NUMBER,

"FLAG" VARCHAR2(50 BYTE),

"NODISCONNECT" NUMBER,

"AMTNODISCONNECT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EBH\_TDM\_REG\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EBH\_TDM\_REG\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"EBH\_CKTS" NUMBER,

"EBH\_COST" NUMBER,

"EBH\_CKTS\_N\_TERMINATED" NUMBER,

"EBH\_CKTS\_N\_INS\_DATE" NUMBER,

"EBH\_SITES" NUMBER,

"CNT\_T\_PD\_CKTS" NUMBER,

"COST\_T\_PD\_CKTS" NUMBER,

"CNT\_T\_N\_DISCO\_N\_PD\_CKTS" NUMBER,

"COST\_T\_N\_DISCO\_N\_PD\_CKTS" NUMBER,

"CNT\_T\_N\_DISCO\_CKTS" NUMBER,

"COST\_T\_N\_DISCO\_CKTS" NUMBER,

"CNT\_T\_WO\_INS\_PD\_CKTS" NUMBER,

"COST\_T\_WO\_INS\_PD\_CKTS" NUMBER,

"CNT\_T\_WO\_INS\_N\_DISCO\_N\_PD\_CKTS" NUMBER,

"COST\_T\_WO\_INS\_N\_DIS\_N\_PD\_CKTS" NUMBER,

"CNT\_T\_WO\_INS\_N\_DISCO\_CKTS" NUMBER,

"COST\_T\_WO\_INS\_N\_DISCO\_CKTS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ECP\_MAP\_LATEST\_XREF\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ECP\_MAP\_LATEST\_XREF\_WRK"

( "SERVER" VARCHAR2(50 BYTE),

"IP\_ADDRESS" VARCHAR2(50 BYTE),

"OMP\_LOC" VARCHAR2(50 BYTE),

"OMP\_NAME" VARCHAR2(50 BYTE),

"ECP\_ABBR" VARCHAR2(50 BYTE),

"SYS\_ID" NUMBER,

"ECP\_SID" NUMBER,

"STATUS" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"XNG\_DOMAIN\_INST\_ID" NUMBER,

"MTSO\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ECP\_SWITCH\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ECP\_SWITCH\_MAP"

( "SYS\_ID" NUMBER,

"ECP\_SID" NUMBER,

"SWITCH\_ID" VARCHAR2(8 BYTE),

"XNG\_DOMAIN\_INST\_ID" NUMBER,

"ECP\_ABBREV" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_ECP" VARCHAR2(50 BYTE),

"ECP\_STATUS" VARCHAR2(50 BYTE) DEFAULT 'Live'

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EHEALTH\_CSR\_LIST\_SAVE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EHEALTH\_CSR\_LIST\_SAVE"

( "HOSTNAME" VARCHAR2(50 BYTE),

"IP" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE,

"ERROR\_CODE" NUMBER,

"VENDOR" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EHEALTH\_CSR\_PERF\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EHEALTH\_CSR\_PERF\_WK"

( "ELEMENT\_NAME" VARCHAR2(100 BYTE),

"ELEMENT\_ID" VARCHAR2(10 BYTE),

"HOSTNAME" VARCHAR2(30 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"TR\_TOKEN" NUMBER,

"EXTRACT\_DATE" DATE,

"PARSE\_STATUS" NUMBER,

"SOURCE" VARCHAR2(30 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"ELEMENT\_TYPE" VARCHAR2(30 BYTE),

"VLAN" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EHEALTH\_PERF\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EHEALTH\_PERF\_WK"

( "ELEMENT\_NAME" VARCHAR2(100 BYTE),

"ELEMENT\_ID" VARCHAR2(10 BYTE),

"HOSTNAME" VARCHAR2(30 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"TR\_TOKEN" NUMBER,

"EXTRACT\_DATE" DATE,

"PARSE\_STATUS" NUMBER,

"SOURCE" VARCHAR2(30 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"ELEMENT\_TYPE" VARCHAR2(30 BYTE),

"VLAN" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ENB\_MODEL\_CLEANUP\_LOG

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ENB\_MODEL\_CLEANUP\_LOG"

( "EQUIP\_INST\_ID" NUMBER(9,0),

"DESCR" VARCHAR2(100 BYTE),

"CHASSIS\_TYPE" VARCHAR2(30 BYTE),

"VERSION\_CHGD" NUMBER(6,0),

"CLEANUP\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ENB\_SUMMARY\_BY\_MKT\_ID

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ENB\_SUMMARY\_BY\_MKT\_ID"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET\_ID" VARCHAR2(6 BYTE),

"LTE\_MARKET\_NAME" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"TOTAL\_ENB" NUMBER,

"BUILT\_AS\_SHELF" NUMBER,

"LIVE" NUMBER,

"OTHER" NUMBER,

"ARE\_DUPLICATED" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ENB\_SUMMARY\_BY\_MKT\_ID\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ENB\_SUMMARY\_BY\_MKT\_ID\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET\_ID" VARCHAR2(6 BYTE),

"LTE\_MARKET\_NAME" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"TOTAL\_ENB" NUMBER,

"BUILT\_AS\_SHELF" NUMBER,

"LIVE" NUMBER,

"OTHER" NUMBER,

"ARE\_DUPLICATED" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ENB\_SUMMARY\_BY\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ENB\_SUMMARY\_BY\_REGION"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_ENB" NUMBER,

"BUILT\_AS\_SHELF" NUMBER,

"LIVE" NUMBER,

"OTHER" NUMBER,

"ARE\_DUPLICATED" NUMBER,

"ENB\_REPORTED" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ENB\_SUMMARY\_BY\_REGION\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ENB\_SUMMARY\_BY\_REGION\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_ENB" NUMBER,

"BUILT\_AS\_SHELF" NUMBER,

"LIVE" NUMBER,

"OTHER" NUMBER,

"ARE\_DUPLICATED" NUMBER,

"ENB\_REPORTED" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EPIPE\_BW\_DS1\_LR

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EPIPE\_BW\_DS1\_LR"

( "CIRC\_PATH\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EQUIPMENT\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EQUIPMENT\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"VSM\_DACS\_COUNT" NUMBER,

"VSM\_SWITCH\_COUNT" NUMBER,

"XNG\_DACS\_MATCH\_VS\_VSM" NUMBER,

"XNG\_SWITCH\_MATCH\_VS\_VSM" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EST\_COST\_AUDIT\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EST\_COST\_AUDIT\_DETAIL"

( "ELEMENT\_TYPE" VARCHAR2(2 CHAR),

"INST\_ID" NUMBER(10,0),

"VERSION\_CHGD" NUMBER(6,0),

"FIELD\_CHGD" VARCHAR2(550 CHAR),

"SUBCLASS\_INST\_ID" NUMBER(10,0),

"SUBCLASS\_OPER" VARCHAR2(1 CHAR),

"SUBCLASS\_TYPE" VARCHAR2(2 CHAR),

"OLD\_VALUE" VARCHAR2(4000 CHAR),

"NEW\_VALUE" VARCHAR2(4000 CHAR),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EST\_COST\_USER\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EST\_COST\_USER\_DETAIL"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"DOMAIN\_NAME" VARCHAR2(50 CHAR),

"INST\_ID" NUMBER(10,0),

"VERSION\_CHGD" NUMBER(6,0),

"CIRC\_HUM\_ID" VARCHAR2(60 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"FIRST\_BILL\_DATE" DATE,

"OLD\_VALUE" VARCHAR2(50 CHAR),

"NEW\_VALUE" VARCHAR2(50 CHAR),

"MRC" VARCHAR2(50 CHAR),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_PREVIOUS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_PREVIOUS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_PREVIOUS\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_PREVIOUS\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_PREV\_BAK\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_PREV\_BAK\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_REPORTED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_REPORTED"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_REPORTED\_BAK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_REPORTED\_BAK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_REPORTED\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_REPORTED\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_RPTD\_BAK\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_RPTD\_BAK\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_SNAPSHOT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_SNAPSHOT"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_SNAPSHOT\_BAK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_SNAPSHOT\_BAK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_SNAP\_BAK\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_SNAP\_BAK\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION DEFERRED

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE( INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_WORK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_WORK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMARY\_WORK\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_WORK\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMPREVBAK\_SNAP\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMPREVBAK\_SNAP\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMRPTDBAK\_SNAP\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMRPTDBAK\_SNAP\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table EXECUTIVE\_SUMRPTD\_SNAP\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."EXECUTIVE\_SUMRPTD\_SNAP\_NEW"

( "TERRITORY" VARCHAR2(100 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE),

"OVERALL\_FAULT\_PCT" NUMBER,

"SEG\_WITH\_PATH\_PCT" NUMBER,

"EQUIP\_COUNT" NUMBER,

"EQUIP\_MATCH\_PCT" NUMBER,

"CKT\_MATCH\_PCT" NUMBER,

"EXTRACT\_DATE" DATE,

"COMPLIANT\_PCT" NUMBER,

"OVERALL\_COM\_PCT" NUMBER,

"PREV\_RPT\_OVERALL\_COM\_PCT" NUMBER,

"DELTA" NUMBER,

"OBS\_COMP\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_CLIENT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_CLIENT"

( "FIELD\_INST\_ID" NUMBER(9,0),

"ELEMENT\_TYPE" VARCHAR2(4 BYTE),

"LABEL" VARCHAR2(50 BYTE),

"COLUMN\_NAME" VARCHAR2(30 BYTE),

"TABLE\_NAME" VARCHAR2(30 BYTE),

"DEFAULT\_DATA\_TYPE" CHAR(1 BYTE),

"DEFAULT\_OPTIONS" VARCHAR2(10 BYTE),

"DEFAULT\_MASK" VARCHAR2(50 BYTE),

"IS\_UPPER\_CASE" CHAR(1 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(50 BYTE),

"DEFAULT\_MIN\_VALUE" NUMBER(13,3),

"DEFAULT\_MAX\_VALUE" NUMBER(13,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_CLIENT\_EQUIP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_CLIENT\_EQUIP"

( "FIELD\_INST\_ID" NUMBER(9,0),

"ELEMENT\_TYPE" VARCHAR2(4 BYTE),

"LABEL" VARCHAR2(50 BYTE),

"COLUMN\_NAME" VARCHAR2(30 BYTE),

"TABLE\_NAME" VARCHAR2(30 BYTE),

"DEFAULT\_DATA\_TYPE" CHAR(1 BYTE),

"DEFAULT\_OPTIONS" VARCHAR2(10 BYTE),

"DEFAULT\_MASK" VARCHAR2(50 BYTE),

"IS\_UPPER\_CASE" CHAR(1 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(50 BYTE),

"DEFAULT\_MIN\_VALUE" NUMBER(13,3),

"DEFAULT\_MAX\_VALUE" NUMBER(13,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_CLIENT\_EQUIP\_TYPE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_CLIENT\_EQUIP\_TYPE"

( "FIELD\_INST\_ID" NUMBER(9,0),

"ELEMENT\_TYPE" VARCHAR2(4 BYTE),

"EQUIP\_CATEGORY" VARCHAR2(30 BYTE),

"COLUMN\_LABEL" VARCHAR2(50 BYTE),

"COLUMN\_NAME" VARCHAR2(30 BYTE),

"TABLE\_NAME" VARCHAR2(30 BYTE),

"DEFAULT\_DATA\_TYPE" CHAR(1 BYTE),

"DEFAULT\_OPTIONS" VARCHAR2(10 BYTE),

"DEFAULT\_MASK" VARCHAR2(50 BYTE),

"IS\_UPPER\_CASE" CHAR(1 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(50 BYTE),

"DEFAULT\_MIN\_VALUE" NUMBER(13,3),

"DEFAULT\_MAX\_VALUE" NUMBER(13,3),

"POPULATED\_CNT" NUMBER,

"CAT\_POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER,

"PCT\_CAT\_POPULATED" NUMBER,

"CAT\_TOT\_CNT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_UDA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA"

( "VAL\_ATTR\_INST\_ID" NUMBER(9,0),

"APPL\_CLASS" VARCHAR2(2 BYTE),

"GROUP\_NAME" VARCHAR2(30 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_DATA\_TYPE" CHAR(1 BYTE),

"OPTION\_FLAGS" VARCHAR2(10 BYTE),

"MASK" VARCHAR2(255 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(3100 BYTE),

"MIN\_VALUE" NUMBER(12,3),

"MAX\_VALUE" NUMBER(12,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_UDA\_ALL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA\_ALL"

( "VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"APPL\_CLASS" VARCHAR2(2 BYTE),

"APPL\_CLASS\_NAME" VARCHAR2(10 BYTE),

"CLASS\_CATEGORY" VARCHAR2(50 BYTE),

"ATTR\_GROUP" VARCHAR2(30 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_DATA\_TYPE" VARCHAR2(23 BYTE),

"OPTION\_FLAGS" VARCHAR2(10 BYTE),

"MASK" VARCHAR2(255 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(3100 BYTE),

"MIN\_VALUE" NUMBER(12,3),

"MAX\_VALUE" NUMBER(12,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_UDA\_CARD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA\_CARD"

( "VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"CARD\_CATEGORY" VARCHAR2(50 BYTE),

"ATTR\_GROUP" VARCHAR2(30 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_DATA\_TYPE" VARCHAR2(23 BYTE),

"OPTION\_FLAGS" VARCHAR2(10 BYTE),

"MASK" VARCHAR2(255 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(3100 BYTE),

"MIN\_VALUE" NUMBER(12,3),

"MAX\_VALUE" NUMBER(12,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_UDA\_EQUIP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA\_EQUIP"

( "VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"EQUIP\_CATEGORY" VARCHAR2(30 BYTE),

"ATTR\_GROUP" VARCHAR2(30 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_DATA\_TYPE" VARCHAR2(23 BYTE),

"OPTION\_FLAGS" VARCHAR2(10 BYTE),

"MASK" VARCHAR2(255 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(3100 BYTE),

"MIN\_VALUE" NUMBER(12,3),

"MAX\_VALUE" NUMBER(12,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_UDA\_PATH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA\_PATH"

( "VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"PATH\_CATEGORY" VARCHAR2(30 BYTE),

"ATTR\_GROUP" VARCHAR2(30 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_DATA\_TYPE" VARCHAR2(23 BYTE),

"OPTION\_FLAGS" VARCHAR2(10 BYTE),

"MASK" VARCHAR2(255 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(3100 BYTE),

"MIN\_VALUE" NUMBER(12,3),

"MAX\_VALUE" NUMBER(12,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_UDA\_PORT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA\_PORT"

( "VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"PORT\_CATEGORY" VARCHAR2(50 BYTE),

"ATTR\_GROUP" VARCHAR2(30 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_DATA\_TYPE" VARCHAR2(23 BYTE),

"OPTION\_FLAGS" VARCHAR2(10 BYTE),

"MASK" VARCHAR2(255 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(3100 BYTE),

"MIN\_VALUE" NUMBER(12,3),

"MAX\_VALUE" NUMBER(12,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_UDA\_SEG

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA\_SEG"

( "VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"SEG\_CATEGORY" VARCHAR2(50 BYTE),

"ATTR\_GROUP" VARCHAR2(30 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_DATA\_TYPE" VARCHAR2(23 BYTE),

"OPTION\_FLAGS" VARCHAR2(10 BYTE),

"MASK" VARCHAR2(255 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(3100 BYTE),

"MIN\_VALUE" NUMBER(12,3),

"MAX\_VALUE" NUMBER(12,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FIELDS\_INFO\_UDA\_SITE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA\_SITE"

( "VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"SITE\_CATEGORY" VARCHAR2(50 BYTE),

"ATTR\_GROUP" VARCHAR2(30 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_DATA\_TYPE" VARCHAR2(23 BYTE),

"OPTION\_FLAGS" VARCHAR2(10 BYTE),

"MASK" VARCHAR2(255 BYTE),

"MAX\_LENGTH" NUMBER(9,0),

"DEFAULT\_VALUE" VARCHAR2(3100 BYTE),

"MIN\_VALUE" NUMBER(12,3),

"MAX\_VALUE" NUMBER(12,3),

"POPULATED\_CNT" NUMBER,

"TOT\_CNT" NUMBER,

"PERCENT\_POPULATED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FILTERED\_SEGMENTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"RECUR\_COSTS" NUMBER(10,2),

"VENDOR" VARCHAR2(30 BYTE),

"Z\_SITE\_INST\_ID" NUMBER(9,0),

"Z\_SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SERVICE" VARCHAR2(20 BYTE),

"PATH\_TYPE" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FILTERED\_SEGMENTS\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"RECUR\_COSTS" NUMBER(10,2),

"VENDOR" VARCHAR2(30 BYTE),

"Z\_SITE\_INST\_ID" NUMBER(9,0),

"Z\_SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SERVICE" VARCHAR2(20 BYTE),

"PATH\_TYPE" VARCHAR2(20 BYTE),

"BILL\_DATE" DATE,

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FWO\_90

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FWO\_90"

( "EXTRACT\_DATE" DATE,

"LOCALE\_ABBR" VARCHAR2(20 BYTE),

"WO\_NAME" VARCHAR2(20 BYTE),

"WO\_INST\_ID" NUMBER(9,0),

"PROJECT\_ID" VARCHAR2(200 BYTE),

"DESCRIPTION" VARCHAR2(80 BYTE),

"TASK\_NUMBER" NUMBER(3,0),

"TASK\_NAME" VARCHAR2(60 BYTE),

"TASK\_OPERATION" VARCHAR2(20 BYTE),

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"WO\_COMMENTS" VARCHAR2(4000 BYTE),

"TASK\_COMMENTS" VARCHAR2(2000 BYTE),

"FAILED\_ON" DATE,

"FAILED\_BY" VARCHAR2(50 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"PUBLISH\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table FWO\_90\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."FWO\_90\_WK"

( "EXTRACT\_DATE" DATE,

"LOCALE\_ABBR" VARCHAR2(20 BYTE),

"WO\_NAME" VARCHAR2(20 BYTE),

"WO\_INST\_ID" NUMBER(9,0),

"PROJECT\_ID" VARCHAR2(200 BYTE),

"DESCRIPTION" VARCHAR2(80 BYTE),

"TASK\_NUMBER" NUMBER(3,0),

"TASK\_NAME" VARCHAR2(60 BYTE),

"TASK\_OPERATION" VARCHAR2(20 BYTE),

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"WO\_COMMENTS" VARCHAR2(4000 BYTE),

"TASK\_COMMENTS" VARCHAR2(2000 BYTE),

"FAILED\_ON" DATE,

"FAILED\_BY" VARCHAR2(50 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"PUBLISH\_DATE" DATE

) SEGMENT CREATION DEFERRED

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GEOPLAN\_DISCOVERY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GEOPLAN\_DISCOVERY"

( "CELL\_NAME" VARCHAR2(200 BYTE),

"CELL\_NUMBER" VARCHAR2(100 BYTE),

"SWITCH\_VSM\_CLLI\_CODE" VARCHAR2(100 BYTE),

"SWITCH\_VENDOR" VARCHAR2(200 BYTE),

"STREET\_ADDRESS" VARCHAR2(200 BYTE),

"CITY" VARCHAR2(40 BYTE),

"STATE" VARCHAR2(40 BYTE),

"ZIP\_CODE" VARCHAR2(40 BYTE),

"CELL" VARCHAR2(256 BYTE),

"SWITCH\_NAME" VARCHAR2(200 BYTE),

"COUNTY" VARCHAR2(40 BYTE),

"ELEVATION\_AMSL\_FT" VARCHAR2(40 BYTE),

"LATITUDE\_DEGREES\_NAD83" VARCHAR2(40 BYTE),

"LONGITUDE\_DEGREES\_NAD83" VARCHAR2(40 BYTE),

"STRUCTURE\_TYPE" VARCHAR2(200 BYTE),

"VENDOR" VARCHAR2(100 BYTE),

"IS\_REPEATER" VARCHAR2(10 BYTE),

"LOCATION\_WORKFLOW\_ID" NUMBER,

"AUDIT\_DATE" DATE,

"AIR\_INTERFACE" VARCHAR2(100 BYTE),

"PEOPLESOFT\_LOCATION\_CODE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GEOPLAN\_DISCOVERY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GEOPLAN\_DISCOVERY\_WK"

( "CELL\_NAME" VARCHAR2(200 BYTE),

"CELL\_NUMBER" VARCHAR2(100 BYTE),

"SWITCH\_VSM\_CLLI\_CODE" VARCHAR2(100 BYTE),

"SWITCH\_VENDOR" VARCHAR2(200 BYTE),

"STREET\_ADDRESS" VARCHAR2(200 BYTE),

"CITY" VARCHAR2(40 BYTE),

"STATE" VARCHAR2(40 BYTE),

"ZIP\_CODE" VARCHAR2(40 BYTE),

"CELL" VARCHAR2(256 BYTE),

"SWITCH\_NAME" VARCHAR2(200 BYTE),

"COUNTY" VARCHAR2(40 BYTE),

"ELEVATION\_AMSL\_FT" VARCHAR2(40 BYTE),

"LATITUDE\_DEGREES\_NAD83" VARCHAR2(40 BYTE),

"LONGITUDE\_DEGREES\_NAD83" VARCHAR2(40 BYTE),

"STRUCTURE\_TYPE" VARCHAR2(200 BYTE),

"VENDOR" VARCHAR2(100 BYTE),

"IS\_REPEATER" VARCHAR2(10 BYTE),

"LOCATION\_WORKFLOW\_ID" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate,

"AIR\_INTERFACE" VARCHAR2(100 BYTE),

"PEOPLESOFT\_LOCATION\_CODE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GEOPLAN\_SMALLCELL\_DISCOVERY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GEOPLAN\_SMALLCELL\_DISCOVERY"

( "LOCATION\_WORKFLOW\_ID" NUMBER,

"CELL\_NUMBER" VARCHAR2(100 BYTE),

"CELL\_NAME" VARCHAR2(200 BYTE),

"CELL\_VERSION" VARCHAR2(20 BYTE),

"AIR\_INTERFACE" VARCHAR2(100 BYTE),

"BAND\_CLASS" VARCHAR2(100 BYTE),

"ENODEB" VARCHAR2(20 BYTE),

"ENODEB\_SITENAME" VARCHAR2(100 BYTE),

"ENODEB\_SECTOR" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(100 BYTE),

"LTE\_CELLTYPE" VARCHAR2(20 BYTE),

"IS\_SMALLCELL" VARCHAR2(10 BYTE),

"IS\_REPEATER" VARCHAR2(10 BYTE),

"MHZ2100ANTENNAMANUFACTURER" VARCHAR2(100 BYTE),

"MHZ1900ANTENNAMANUFACTURER" VARCHAR2(100 BYTE),

"MHZ850ANTENNAMANUFACTURER" VARCHAR2(100 BYTE),

"MHZ700ANTENNAMANUFACTURER" VARCHAR2(100 BYTE),

"MHZ2100ANTENNAMODEL" VARCHAR2(100 BYTE),

"MHZ1900ANTENNAMODEL" VARCHAR2(100 BYTE),

"MHZ850ANTENNAMODEL" VARCHAR2(100 BYTE),

"MHZ700ANTENNAMODEL" VARCHAR2(100 BYTE),

"AUDIT\_DATE" DATE,

"DISC\_FILENAME" VARCHAR2(100 BYTE),

"SECTOR\_CARRIER\_EQUIP\_TYPE" VARCHAR2(100 BYTE),

"SECTOR\_EQUIP\_TYPE" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GEOPLAN\_SMALLCELL\_DISCOVERY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GEOPLAN\_SMALLCELL\_DISCOVERY\_WK"

( "LOCATION\_WORKFLOW\_ID" NUMBER,

"CELL\_NUMBER" VARCHAR2(100 BYTE),

"CELL\_NAME" VARCHAR2(200 BYTE),

"CELL\_VERSION" VARCHAR2(20 BYTE),

"AIR\_INTERFACE" VARCHAR2(100 BYTE),

"BAND\_CLASS" VARCHAR2(100 BYTE),

"ENODEB" VARCHAR2(20 BYTE),

"ENODEB\_SITENAME" VARCHAR2(100 BYTE),

"ENODEB\_SECTOR" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(100 BYTE),

"LTE\_CELLTYPE" VARCHAR2(20 BYTE),

"IS\_SMALLCELL" VARCHAR2(10 BYTE),

"IS\_REPEATER" VARCHAR2(10 BYTE),

"MHZ2100ANTENNAMANUFACTURER" VARCHAR2(100 BYTE),

"MHZ1900ANTENNAMANUFACTURER" VARCHAR2(100 BYTE),

"MHZ850ANTENNAMANUFACTURER" VARCHAR2(100 BYTE),

"MHZ700ANTENNAMANUFACTURER" VARCHAR2(100 BYTE),

"MHZ2100ANTENNAMODEL" VARCHAR2(100 BYTE),

"MHZ1900ANTENNAMODEL" VARCHAR2(100 BYTE),

"MHZ850ANTENNAMODEL" VARCHAR2(100 BYTE),

"MHZ700ANTENNAMODEL" VARCHAR2(100 BYTE),

"AUDIT\_DATE" DATE,

"DISC\_FILENAME" VARCHAR2(100 BYTE),

"SECTOR\_CARRIER\_EQUIP\_TYPE" VARCHAR2(100 BYTE),

"SECTOR\_EQUIP\_TYPE" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GIGE\_VLAN\_AUDIT\_ISSUE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GIGE\_VLAN\_AUDIT\_ISSUE"

( "GIGE\_VLAN\_ISSUE\_ID" NUMBER,

"GIGE\_VLAN\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GIGE\_VLAN\_AUDIT\_ISSUE\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GIGE\_VLAN\_AUDIT\_ISSUE\_WK"

( "GIGE\_VLAN\_ISSUE\_ID" NUMBER,

"GIGE\_VLAN\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GIGE\_VLAN\_ISSUE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GIGE\_VLAN\_ISSUE"

( "DECSRIPTION" VARCHAR2(100 BYTE),

"IS\_CRITICAL" CHAR(1 BYTE) DEFAULT 'N',

"COMMENTS" VARCHAR2(256 BYTE),

"GIGE\_VLAN\_ISSUE\_ID" NUMBER(\*,0),

"STORED\_VLAN\_INST\_ID" CHAR(1 BYTE) DEFAULT 'N'

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GIGE\_VLAN\_MSPP\_NGMLS\_QT\_MAP\_LR

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GIGE\_VLAN\_MSPP\_NGMLS\_QT\_MAP\_LR"

( "GIGE\_INST\_ID" NUMBER(9,0),

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_NUMBER" NUMBER,

"PATH\_C\_NGMLS\_MSPP" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"EQUIP\_TYPE" VARCHAR2(5 BYTE),

"CARD\_INST\_ID" NUMBER,

"PORT\_INST\_ID" NUMBER(9,0),

"NE\_TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"EQ\_TEST\_HEAD\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table GRANITE\_IP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."GRANITE\_IP"

( "EQUIP\_INST\_ID" NUMBER(9,0),

"DESCR" VARCHAR2(100 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"IP" VARCHAR2(3100 BYTE),

"VENDOR" VARCHAR2(40 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"MAC\_ADDRESS" VARCHAR2(3100 BYTE),

"DOMAIN\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HIGH\_CAPACITY\_PATH\_CHAN

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN"

( "DOMAIN\_INST\_ID" NUMBER(9,0),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"A\_SIDE\_SITE" VARCHAR2(100 BYTE),

"Z\_SIDE\_SITE" VARCHAR2(100 BYTE),

"PATHPARENT\_NAME" VARCHAR2(100 BYTE),

"PATHPARENT\_ID" NUMBER(9,0),

"PATHPARENT\_TYPE" VARCHAR2(30 BYTE),

"PATHPARENT\_STATUS" VARCHAR2(20 BYTE),

"PATHPARENT\_BANDWIDTH" VARCHAR2(30 BYTE),

"PATHPARENT\_NCHL" NUMBER(6,0),

"PATHPARENT\_NCHLA" NUMBER(6,0),

"PATHPARENT\_TOT\_MRC" NUMBER(10,2),

"PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PATH\_TYPE" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"NBR\_CHANNELS" NUMBER(6,0),

"NBR\_CHAN\_ASSIGNED" NUMBER(6,0),

"TOT\_MRC" NUMBER(10,2),

"PATHPARENT\_PCT\_ASGN" NUMBER,

"PATHPARENT\_PCT\_GRP" NUMBER,

"RUN\_DATE" DATE,

"PATH\_TYPE\_CK" VARCHAR2(30 BYTE),

"CHANNEL\_USED" NUMBER(6,0),

"TOTAL\_BPS\_AVAIL" NUMBER(18,0),

"BPS\_ASSIGNED" NUMBER(18,0),

"BPS\_UTIL" NUMBER(5,0),

"BPS\_UTIL\_PREV" NUMBER(5,0),

"BPS\_UTIL\_MOD\_TS" DATE,

"PP\_TRANS\_ACT" VARCHAR2(50 BYTE),

"PP\_EST\_COMP" DATE,

"PP\_ACT\_JUST" VARCHAR2(256 BYTE),

"PP\_ACT\_DATE" DATE,

"PP\_EST\_COMP\_GRP" NUMBER,

"TRANS\_ACT" VARCHAR2(50 BYTE),

"EST\_COMP" DATE,

"ACT\_JUST" VARCHAR2(256 BYTE),

"ACT\_DATE" DATE,

"EST\_COMP\_GRP" NUMBER,

"PP\_ACT\_TOT\_MRC" NUMBER,

"ACT\_TOT\_MRC" NUMBER,

"P\_TOP\_AR" NUMBER,

"P\_TOP\_ARBW" NUMBER,

"EST\_COMP\_DAYS" NUMBER,

"PP\_EST\_COMP\_DAYS" NUMBER,

"PP\_ACT\_JUST\_COMT" VARCHAR2(256 BYTE),

"ACT\_JUST\_COMT" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 220200960 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HIGH\_CAPACITY\_PATH\_CHAN\_ACT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_ACT"

( "CIRC\_PATH\_INST\_ID" NUMBER(10,0),

"TRANS\_ACT" VARCHAR2(50 BYTE),

"EST\_COMP" DATE,

"ACT\_JUST" VARCHAR2(256 BYTE),

"ACT\_DATE" DATE,

"EST\_COMP\_GRP" NUMBER,

"ACT\_TOT\_MRC" NUMBER,

"EST\_COMP\_DAYS" NUMBER,

"ACT\_JUST\_COMT" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 52428800 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HIGH\_CAPACITY\_PATH\_CHAN\_CHILD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_CHILD"

( "CIRC\_PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE),

"PATH\_TYPE" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"NBR\_CHANNELS" NUMBER(6,0),

"NBR\_CHAN\_ASSIGNED" NUMBER,

"TOT\_MRC" NUMBER,

"PATH\_TYPE\_CK" VARCHAR2(30 BYTE),

"PARENT\_PATH\_INST\_ID" NUMBER(10,0),

"CHANNEL\_USED" NUMBER(6,0),

"TRANS\_ACT" VARCHAR2(50 BYTE),

"EST\_COMP" DATE,

"ACT\_JUST" VARCHAR2(256 BYTE),

"ACT\_DATE" DATE,

"EST\_COMP\_GRP" NUMBER,

"ACT\_TOT\_MRC" NUMBER,

"EST\_COMP\_DAYS" NUMBER,

"ACT\_JUST\_COMT" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 83886080 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HIGH\_CAPACITY\_PATH\_CHAN\_PARENT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_PARENT"

( "DOMAIN\_INST\_ID" NUMBER(10,0),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"A\_SIDE\_SITE" VARCHAR2(100 BYTE),

"Z\_SIDE\_SITE" VARCHAR2(100 BYTE),

"PATHPARENT\_NAME" VARCHAR2(100 BYTE),

"PATHPARENT\_ID" NUMBER(10,0),

"PATHPARENT\_TYPE" VARCHAR2(30 BYTE),

"PATHPARENT\_STATUS" VARCHAR2(20 BYTE),

"PATHPARENT\_BANDWIDTH" VARCHAR2(30 BYTE),

"PATHPARENT\_NCHL" NUMBER(6,0),

"PATHPARENT\_NCHLA" NUMBER,

"P\_TOT\_MRC" NUMBER,

"P\_PCT\_ASGN" NUMBER,

"P\_PCT\_GRP" NUMBER,

"TOTAL\_BPS\_AVAIL" NUMBER(18,0),

"BPS\_ASSIGNED" NUMBER(18,0),

"BPS\_UTIL" NUMBER(5,0),

"P\_TRANS\_ACT" VARCHAR2(50 BYTE),

"P\_EST\_COMP" DATE,

"P\_ACT\_JUST" VARCHAR2(256 BYTE),

"P\_ACT\_DATE" DATE,

"P\_EST\_COMP\_GRP" NUMBER,

"P\_ACT\_TOT\_MRC" NUMBER,

"P\_TOP\_AR" NUMBER,

"P\_TOP\_ARBW" NUMBER,

"P\_EST\_COMP\_DAYS" NUMBER,

"P\_ACT\_JUST\_COMT" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 31457280 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HIGH\_CAPACITY\_PATH\_CHAN\_UTIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_UTIL"

( "PATHPARENT\_ID" NUMBER(10,0),

"TOTAL\_BPS\_AVAIL" NUMBER(18,0),

"BPS\_ASSIGNED" NUMBER(18,0),

"BPS\_UTIL" NUMBER,

"LAST\_MOD\_BY" VARCHAR2(50 BYTE),

"LAST\_MOD\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 20971520 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HIGH\_CAPACITY\_PATH\_CHAN\_UTIL\_B

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_UTIL\_B"

( "PATHPARENT\_ID" NUMBER(10,0),

"TOTAL\_BPS\_AVAIL" NUMBER(18,0),

"BPS\_ASSIGNED" NUMBER(18,0),

"BPS\_UTIL" NUMBER,

"LAST\_MOD\_BY" VARCHAR2(50 BYTE),

"LAST\_MOD\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 20971520 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HIGH\_CAPACITY\_PATH\_CHAN\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_WK"

( "DOMAIN\_INST\_ID" NUMBER(9,0),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"A\_SIDE\_SITE" VARCHAR2(100 BYTE),

"Z\_SIDE\_SITE" VARCHAR2(100 BYTE),

"PATHPARENT\_NAME" VARCHAR2(100 BYTE),

"PATHPARENT\_ID" NUMBER(9,0),

"PATHPARENT\_TYPE" VARCHAR2(30 BYTE),

"PATHPARENT\_STATUS" VARCHAR2(20 BYTE),

"PATHPARENT\_BANDWIDTH" VARCHAR2(30 BYTE),

"PATHPARENT\_NCHL" NUMBER(6,0),

"PATHPARENT\_NCHLA" NUMBER(6,0),

"PATHPARENT\_TOT\_MRC" NUMBER(10,2),

"PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PATH\_TYPE" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"NBR\_CHANNELS" NUMBER(6,0),

"NBR\_CHAN\_ASSIGNED" NUMBER(6,0),

"TOT\_MRC" NUMBER(10,2),

"PATHPARENT\_PCT\_ASGN" NUMBER,

"PATHPARENT\_PCT\_GRP" NUMBER,

"RUN\_DATE" DATE,

"PATH\_TYPE\_CK" VARCHAR2(30 BYTE),

"CHANNEL\_USED" NUMBER(6,0),

"TOTAL\_BPS\_AVAIL" NUMBER(18,0),

"BPS\_ASSIGNED" NUMBER(18,0),

"BPS\_UTIL" NUMBER(5,0),

"BPS\_UTIL\_PREV" NUMBER(5,0),

"BPS\_UTIL\_MOD\_TS" DATE,

"PP\_TRANS\_ACT" VARCHAR2(50 BYTE),

"PP\_EST\_COMP" DATE,

"PP\_ACT\_JUST" VARCHAR2(256 BYTE),

"PP\_ACT\_DATE" DATE,

"PP\_EST\_COMP\_GRP" NUMBER,

"TRANS\_ACT" VARCHAR2(50 BYTE),

"EST\_COMP" DATE,

"ACT\_JUST" VARCHAR2(256 BYTE),

"ACT\_DATE" DATE,

"EST\_COMP\_GRP" NUMBER,

"PP\_ACT\_TOT\_MRC" NUMBER,

"ACT\_TOT\_MRC" NUMBER,

"P\_TOP\_AR" NUMBER,

"P\_TOP\_ARBW" NUMBER,

"EST\_COMP\_DAYS" NUMBER,

"PP\_EST\_COMP\_DAYS" NUMBER,

"PP\_ACT\_JUST\_COMT" VARCHAR2(256 BYTE),

"ACT\_JUST\_COMT" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 220200960 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HIGH\_CAP\_PATH\_CHAN\_05142014

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HIGH\_CAP\_PATH\_CHAN\_05142014"

( "DOMAIN\_INST\_ID" NUMBER(9,0),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"A\_SIDE\_SITE" VARCHAR2(100 BYTE),

"Z\_SIDE\_SITE" VARCHAR2(100 BYTE),

"PATHPARENT\_NAME" VARCHAR2(100 BYTE),

"PATHPARENT\_ID" NUMBER(9,0),

"PATHPARENT\_TYPE" VARCHAR2(30 BYTE),

"PATHPARENT\_STATUS" VARCHAR2(20 BYTE),

"PATHPARENT\_BANDWIDTH" VARCHAR2(30 BYTE),

"PATHPARENT\_NCHL" NUMBER(6,0),

"PATHPARENT\_NCHLA" NUMBER(6,0),

"PATHPARENT\_TOT\_MRC" NUMBER(10,2),

"PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PATH\_TYPE" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"NBR\_CHANNELS" NUMBER(6,0),

"NBR\_CHAN\_ASSIGNED" NUMBER(6,0),

"TOT\_MRC" NUMBER(10,2),

"PATHPARENT\_PCT\_ASGN" NUMBER,

"PATHPARENT\_PCT\_GRP" NUMBER,

"RUN\_DATE" DATE,

"PATH\_TYPE\_CK" VARCHAR2(30 BYTE),

"PCT\_ASGN" NUMBER,

"PCT\_GRP" NUMBER,

"PATHPARENT\_LAST\_CHG" DATE,

"PATH\_LAST\_CHG" DATE,

"CHANNEL\_USED" NUMBER(6,0),

"PATHPARENT\_TOP30" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HPOV\_CSR\_DEVICES\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HPOV\_CSR\_DEVICES\_WK"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_IP" VARCHAR2(40 BYTE),

"CSR\_CLASS" VARCHAR2(100 BYTE),

"LOAD\_DATE" DATE DEFAULT sysdate,

"NE\_STATUS" VARCHAR2(100 BYTE),

"NE\_SOURCE" VARCHAR2(100 BYTE),

"COMMON\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HPOV\_VS\_XNG\_CSR\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT"

( "CSR\_ID" NUMBER,

"NE\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"LIVE\_IN\_XNG" CHAR(1 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"CSR\_IP" VARCHAR2(40 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table HPOV\_VS\_XNG\_CSR\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT\_WK"

( "CSR\_ID" NUMBER,

"NE\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"LIVE\_IN\_XNG" CHAR(1 BYTE),

"EQUIP\_INST\_ID" NUMBER(10,0),

"CSR\_IP" VARCHAR2(40 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table IGNORED\_NE\_FOR\_REPORTING

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."IGNORED\_NE\_FOR\_REPORTING"

( "DEVICE\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK"

( "XNG\_EQUIP\_INST\_ID" NUMBER,

"NE\_TID" VARCHAR2(30 BYTE),

"AUDIT\_STATUS" VARCHAR2(30 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"DEVICE\_XCONN\_ELEMENTS" VARCHAR2(128 BYTE),

"MEMBER\_NBR" NUMBER,

"DISCREPANCY\_DETAILS" VARCHAR2(256 BYTE),

"XNG\_LEG\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK"

( "XNG\_SITE\_INST\_ID" NUMBER(9,0),

"XNG\_EQUIP\_INST\_ID" NUMBER(9,0),

"TID" VARCHAR2(30 BYTE),

"XNG\_DEVICE\_TYPE" VARCHAR2(20 BYTE),

"AUDIT\_STATUS" VARCHAR2(20 BYTE),

"AUDIT\_DATE" DATE,

"MATCHED\_XCONN\_COUNT" NUMBER(9,0),

"DISCREPANCY\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"MATCHED\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"DESCRIPTION" VARCHAR2(1024 BYTE),

"STATUS" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INCOMPLETE\_PATH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INCOMPLETE\_PATH"

( "DOMAIN\_INST\_ID" NUMBER(9,0),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"CIRC\_TYPE" VARCHAR2(30 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"NEXT\_PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE),

"NEXT\_PATH\_NAME" VARCHAR2(100 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"NEXT\_PATH\_STATUS" VARCHAR2(20 BYTE),

"A\_PORT" NUMBER,

"Z\_PORT" NUMBER,

"A\_SITE\_ID" NUMBER,

"Z\_SITE\_ID" NUMBER,

"A\_SITE" VARCHAR2(100 BYTE),

"Z\_SITE" VARCHAR2(100 BYTE),

"BILL\_DATE" VARCHAR2(3100 BYTE),

"ERROR\_IND" CHAR(1 BYTE),

"ERROR\_TYPE" VARCHAR2(68 BYTE),

"RUN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INCOMPLETE\_PATH\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INCOMPLETE\_PATH\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_CNT" NUMBER,

"ERROR\_CNT" NUMBER,

"COMPLIANT\_PCT" NUMBER,

"SORT\_ORD" NUMBER,

"ROW\_TYPE" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INCOMPLETE\_PATH\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INCOMPLETE\_PATH\_WRK"

( "DOMAIN\_INST\_ID" NUMBER(9,0),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"CIRC\_TYPE" VARCHAR2(30 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"NEXT\_PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE),

"NEXT\_PATH\_NAME" VARCHAR2(100 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"NEXT\_PATH\_STATUS" VARCHAR2(20 BYTE),

"A\_PORT" NUMBER,

"Z\_PORT" NUMBER,

"A\_SITE\_ID" NUMBER,

"Z\_SITE\_ID" NUMBER,

"A\_SITE" VARCHAR2(100 BYTE),

"Z\_SITE" VARCHAR2(100 BYTE),

"BILL\_DATE" VARCHAR2(3100 BYTE),

"ERROR\_IND" CHAR(1 BYTE),

"ERROR\_TYPE" VARCHAR2(68 BYTE),

"RUN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_DEVICES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_DEVICES"

( "NE\_ID" NUMBER,

"TID" VARCHAR2(100 BYTE),

"NE\_NAME" VARCHAR2(100 BYTE),

"NE\_ADDR1" VARCHAR2(100 BYTE),

"NE\_ADDR2" VARCHAR2(100 BYTE),

"NE\_REGION" VARCHAR2(100 BYTE),

"NE\_IP" VARCHAR2(100 BYTE),

"NE\_TYPE" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_ADM\_COMP\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_ADM\_COMP\_SUMMARY"

( "AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(128 BYTE),

"INC\_MSPP\_COUNT" NUMBER,

"XCONNECT\_COUNT" NUMBER,

"VSM\_DEVICE\_COUNT" NUMBER,

"INC\_VSM\_MATCH\_COUNT" NUMBER,

"INC\_VSM\_XNG\_MATCH\_COUNT" NUMBER,

"LIVE\_XCONNECT\_MATCH\_COUNT" NUMBER,

"NOT\_LIVE\_XCONNECT\_MATCH\_COUNT" NUMBER,

"MSPP\_REPORTED" NUMBER,

"TOTAL\_FOR\_MSPP\_REPORTED" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK"

( "AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(128 BYTE),

"INC\_MSPP\_COUNT" NUMBER,

"XCONNECT\_COUNT" NUMBER,

"VSM\_DEVICE\_COUNT" NUMBER,

"INC\_VSM\_MATCH\_COUNT" NUMBER,

"INC\_VSM\_XNG\_MATCH\_COUNT" NUMBER,

"LIVE\_XCONNECT\_MATCH\_COUNT" NUMBER,

"NOT\_LIVE\_XCONNECT\_MATCH\_COUNT" NUMBER,

"MSPP\_REPORTED" NUMBER,

"TOTAL\_FOR\_MSPP\_REPORTED" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_COMPLIANCE\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_COMPLIANCE\_SUMMARY"

( "AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(128 BYTE),

"INC\_MSPP\_COUNT" NUMBER,

"XCONNECT\_COUNT" NUMBER,

"VSM\_DEVICE\_COUNT" NUMBER,

"INC\_VSM\_MATCH\_COUNT" NUMBER,

"INC\_VSM\_XNG\_MATCH\_COUNT" NUMBER,

"LIVE\_XCONNECT\_MATCH\_COUNT" NUMBER,

"NOT\_LIVE\_XCONNECT\_MATCH\_COUNT" NUMBER,

"MSPP\_REPORTED" NUMBER,

"TOTAL\_FOR\_MSPP\_REPORTED" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK"

( "AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(128 BYTE),

"INC\_MSPP\_COUNT" NUMBER,

"XCONNECT\_COUNT" NUMBER,

"VSM\_DEVICE\_COUNT" NUMBER,

"INC\_VSM\_MATCH\_COUNT" NUMBER,

"INC\_VSM\_XNG\_MATCH\_COUNT" NUMBER,

"LIVE\_XCONNECT\_MATCH\_COUNT" NUMBER,

"NOT\_LIVE\_XCONNECT\_MATCH\_COUNT" NUMBER,

"MSPP\_REPORTED" NUMBER,

"TOTAL\_FOR\_MSPP\_REPORTED" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_CONNECTION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_CONNECTION"

( "CIRCUIT\_ID" VARCHAR2(256 BYTE),

"TID" VARCHAR2(30 BYTE),

"ALOGICALPORT" VARCHAR2(50 BYTE),

"ZLOGICALPORT" VARCHAR2(50 BYTE),

"SIGNAL\_TYPE" VARCHAR2(20 BYTE),

"CONNECTION\_TIME" DATE,

"WHO" VARCHAR2(25 BYTE),

"ORDER\_NUM" VARCHAR2(20 BYTE),

"USER\_NAME" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_CONNECTION\_03201603

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_CONNECTION\_03201603"

( "CIRCUIT\_ID" VARCHAR2(256 BYTE),

"TID" VARCHAR2(30 BYTE),

"ALOGICALPORT" VARCHAR2(50 BYTE),

"ZLOGICALPORT" VARCHAR2(50 BYTE),

"SIGNAL\_TYPE" VARCHAR2(20 BYTE),

"CONNECTION\_TIME" DATE,

"WHO" VARCHAR2(25 BYTE),

"ORDER\_NUM" VARCHAR2(20 BYTE),

"USER\_NAME" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_LOGICALPORT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT"

( "TID" VARCHAR2(50 BYTE),

"RACKADDRESS" VARCHAR2(15 BYTE),

"SUBRACKADDRESS" VARCHAR2(15 BYTE),

"SLOTADDRESS" VARCHAR2(30 BYTE),

"PORTADDRESS" VARCHAR2(50 BYTE),

"LOGICALPORTADDRESS" VARCHAR2(50 BYTE),

"LOGICALPORTTYPE" VARCHAR2(10 BYTE),

"CONNECTSTATUS" VARCHAR2(15 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_LOGICALPORT\_03201602

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT\_03201602"

( "TID" VARCHAR2(50 BYTE),

"RACKADDRESS" VARCHAR2(15 BYTE),

"SUBRACKADDRESS" VARCHAR2(15 BYTE),

"SLOTADDRESS" VARCHAR2(30 BYTE),

"PORTADDRESS" VARCHAR2(50 BYTE),

"LOGICALPORTADDRESS" VARCHAR2(50 BYTE),

"LOGICALPORTTYPE" VARCHAR2(10 BYTE),

"CONNECTSTATUS" VARCHAR2(15 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_LOGICALPORT\_05311602

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT\_05311602"

( "TID" VARCHAR2(50 BYTE),

"RACKADDRESS" VARCHAR2(15 BYTE),

"SUBRACKADDRESS" VARCHAR2(15 BYTE),

"SLOTADDRESS" VARCHAR2(30 BYTE),

"PORTADDRESS" VARCHAR2(50 BYTE),

"LOGICALPORTADDRESS" VARCHAR2(50 BYTE),

"LOGICALPORTTYPE" VARCHAR2(10 BYTE),

"CONNECTSTATUS" VARCHAR2(15 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION DEFERRED

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_LOGICALPORT\_08021602

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT\_08021602"

( "TID" VARCHAR2(50 BYTE),

"RACKADDRESS" VARCHAR2(15 BYTE),

"SUBRACKADDRESS" VARCHAR2(15 BYTE),

"SLOTADDRESS" VARCHAR2(30 BYTE),

"PORTADDRESS" VARCHAR2(50 BYTE),

"LOGICALPORTADDRESS" VARCHAR2(50 BYTE),

"LOGICALPORTTYPE" VARCHAR2(10 BYTE),

"CONNECTSTATUS" VARCHAR2(15 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION DEFERRED

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_NE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_NE"

( "TID" VARCHAR2(50 BYTE),

"NETYPE" VARCHAR2(50 BYTE),

"NESOFTWARERELEASE" VARCHAR2(30 BYTE),

"GNE\_TID" VARCHAR2(30 BYTE),

"DNO\_DATE" DATE,

"NEID" NUMBER,

"EXTRACT\_DATE" DATE,

"IP\_ADDRESS" VARCHAR2(30 BYTE),

"PORT" VARCHAR2(5 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_NE\_03201600

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_NE\_03201600"

( "TID" VARCHAR2(50 BYTE),

"NETYPE" VARCHAR2(50 BYTE),

"NESOFTWARERELEASE" VARCHAR2(30 BYTE),

"GNE\_TID" VARCHAR2(30 BYTE),

"DNO\_DATE" DATE,

"NEID" NUMBER,

"EXTRACT\_DATE" DATE,

"IP\_ADDRESS" VARCHAR2(30 BYTE),

"PORT" VARCHAR2(5 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_OPTICAL\_LINK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_OPTICAL\_LINK"

( "A\_TID" VARCHAR2(30 BYTE),

"A\_PORT\_ADDRESS" VARCHAR2(50 BYTE),

"Z\_TID" VARCHAR2(30 BYTE),

"Z\_PORT\_ADDRESS" VARCHAR2(50 BYTE),

"SIGNAL\_TYPE" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_OPTICAL\_LINK\_03201600

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_OPTICAL\_LINK\_03201600"

( "A\_TID" VARCHAR2(30 BYTE),

"A\_PORT\_ADDRESS" VARCHAR2(50 BYTE),

"Z\_TID" VARCHAR2(30 BYTE),

"Z\_PORT\_ADDRESS" VARCHAR2(50 BYTE),

"SIGNAL\_TYPE" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_PHYSICALPORT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_PHYSICALPORT"

( "TID" VARCHAR2(50 BYTE),

"RACKADDRESS" VARCHAR2(5 BYTE),

"SUBRACKADDRESS" VARCHAR2(15 BYTE),

"SLOTADDRESS" VARCHAR2(30 BYTE),

"PORTADDRESS" VARCHAR2(50 BYTE),

"PORTTYPE" VARCHAR2(15 BYTE),

"SERVICESTATE" VARCHAR2(15 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_PHYSICALPORT\_03201600

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_PHYSICALPORT\_03201600"

( "TID" VARCHAR2(50 BYTE),

"RACKADDRESS" VARCHAR2(5 BYTE),

"SUBRACKADDRESS" VARCHAR2(15 BYTE),

"SLOTADDRESS" VARCHAR2(30 BYTE),

"PORTADDRESS" VARCHAR2(50 BYTE),

"PORTTYPE" VARCHAR2(15 BYTE),

"SERVICESTATE" VARCHAR2(15 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_PROTECTPORT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_PROTECTPORT"

( "TID" VARCHAR2(30 BYTE),

"WORKING\_PORT" VARCHAR2(50 BYTE),

"PROTECT\_PORT" VARCHAR2(50 BYTE),

"PROTECTION\_TYPE" VARCHAR2(30 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_PROTECTPORT\_03201600

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_PROTECTPORT\_03201600"

( "TID" VARCHAR2(30 BYTE),

"WORKING\_PORT" VARCHAR2(50 BYTE),

"PROTECT\_PORT" VARCHAR2(50 BYTE),

"PROTECTION\_TYPE" VARCHAR2(30 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_RACK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_RACK"

( "TID" VARCHAR2(50 BYTE),

"RACKTYPE" VARCHAR2(15 BYTE),

"RACKPOSITION" NUMBER,

"RACKADDRESS" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_RACK\_03201600

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_RACK\_03201600"

( "TID" VARCHAR2(50 BYTE),

"RACKTYPE" VARCHAR2(15 BYTE),

"RACKPOSITION" NUMBER,

"RACKADDRESS" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_SLOT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_SLOT"

( "TID" VARCHAR2(50 BYTE),

"RACKADDRESS" VARCHAR2(5 BYTE),

"SUBRACKADDRESS" VARCHAR2(15 BYTE),

"SLOTSTATE" VARCHAR2(20 BYTE),

"SLOTADDRESS" VARCHAR2(50 BYTE),

"TYPE" VARCHAR2(100 BYTE),

"APPLICATION\_CODE" VARCHAR2(50 BYTE),

"SSN" VARCHAR2(20 BYTE),

"CLEI" VARCHAR2(20 BYTE),

"ECI" VARCHAR2(30 BYTE),

"SERIALNUMBER" VARCHAR2(25 BYTE),

"VERSION" VARCHAR2(20 BYTE),

"SERVICESTATE" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_SLOT\_03201600

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_SLOT\_03201600"

( "TID" VARCHAR2(50 BYTE),

"RACKADDRESS" VARCHAR2(5 BYTE),

"SUBRACKADDRESS" VARCHAR2(15 BYTE),

"SLOTSTATE" VARCHAR2(20 BYTE),

"SLOTADDRESS" VARCHAR2(50 BYTE),

"TYPE" VARCHAR2(100 BYTE),

"APPLICATION\_CODE" VARCHAR2(50 BYTE),

"SSN" VARCHAR2(20 BYTE),

"CLEI" VARCHAR2(20 BYTE),

"ECI" VARCHAR2(30 BYTE),

"SERIALNUMBER" VARCHAR2(25 BYTE),

"VERSION" VARCHAR2(20 BYTE),

"SERVICESTATE" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_XCONN\_AUDIT\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_DETAILS"

( "XNG\_EQUIP\_INST\_ID" NUMBER,

"NE\_TID" VARCHAR2(30 BYTE),

"AUDIT\_STATUS" VARCHAR2(30 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"DEVICE\_XCONN\_ELEMENTS" VARCHAR2(128 BYTE),

"MEMBER\_NBR" NUMBER,

"DISCREPANCY\_DETAILS" VARCHAR2(256 BYTE),

"XNG\_LEG\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_XCONN\_AUDIT\_DETAILS\_B

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_DETAILS\_B"

( "XNG\_EQUIP\_INST\_ID" NUMBER,

"NE\_TID" VARCHAR2(30 BYTE),

"AUDIT\_STATUS" VARCHAR2(30 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"DEVICE\_XCONN\_ELEMENTS" VARCHAR2(128 BYTE),

"MEMBER\_NBR" NUMBER,

"DISCREPANCY\_DETAILS" VARCHAR2(256 BYTE),

"XNG\_LEG\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_XCONN\_AUDIT\_DETAILS\_T

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_DETAILS\_T"

( "XNG\_EQUIP\_INST\_ID" NUMBER,

"NE\_TID" VARCHAR2(30 BYTE),

"AUDIT\_STATUS" VARCHAR2(30 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER,

"DEVICE\_XCONN\_ELEMENTS" VARCHAR2(128 BYTE),

"MEMBER\_NBR" NUMBER,

"DISCREPANCY\_DETAILS" VARCHAR2(256 BYTE),

"XNG\_LEG\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table INC\_MSPP\_XCONN\_AUDIT\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_SUMMARY"

( "XNG\_SITE\_INST\_ID" NUMBER(9,0),

"XNG\_EQUIP\_INST\_ID" NUMBER(9,0),

"TID" VARCHAR2(30 BYTE),

"XNG\_DEVICE\_TYPE" VARCHAR2(20 BYTE),

"AUDIT\_STATUS" VARCHAR2(20 BYTE),

"AUDIT\_DATE" DATE,

"MATCHED\_XCONN\_COUNT" NUMBER(9,0),

"DISCREPANCY\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"MATCHED\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"DESCRIPTION" VARCHAR2(1024 BYTE),

"STATUS" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JC\_NORTEL\_BTS\_DCG

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JC\_NORTEL\_BTS\_DCG"

( "SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_NAME" VARCHAR2(100 BYTE),

"CELL\_TYPE" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JUNIPER\_DEVICES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JUNIPER\_DEVICES"

( "HOST\_NAME" VARCHAR2(50 CHAR),

"DEVICE\_IP" VARCHAR2(50 CHAR),

"DEVICE\_ID" NUMBER(10,0),

"DEVICE\_VENDOR" VARCHAR2(50 CHAR),

"DEVICE\_MODEL" VARCHAR2(50 CHAR),

"DEVICE\_PARTITION" VARCHAR2(50 CHAR),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 CHAR),

"APPLICATION\_ID" VARCHAR2(50 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JUNIPER\_EQ\_SUMM\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JUNIPER\_EQ\_SUMM\_REGION"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"TOTAL\_NE" NUMBER,

"DUPLICATE\_NE" NUMBER,

"MATCHED\_NE" NUMBER,

"PERCENT\_MATCHED" NUMBER,

"PASSED\_AUDIT" NUMBER,

"NAMING\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JUNIPER\_EQ\_SUMM\_REGION\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JUNIPER\_EQ\_SUMM\_REGION\_WK"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"TOTAL\_NE" NUMBER,

"DUPLICATE\_NE" NUMBER,

"MATCHED\_NE" NUMBER,

"PERCENT\_MATCHED" NUMBER,

"PASSED\_AUDIT" NUMBER,

"NAMING\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JUNIPER\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JUNIPER\_NE\_VS\_XNG\_AUDIT"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"HOST\_NAME" VARCHAR2(50 CHAR),

"DEVICE\_IP" VARCHAR2(50 CHAR),

"DEVICE\_ID" NUMBER,

"DEVICE\_VENDOR" VARCHAR2(50 CHAR),

"DEVICE\_MODEL" VARCHAR2(50 CHAR),

"DEVICE\_PARTITION" VARCHAR2(50 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"HOST\_NAME" VARCHAR2(50 CHAR),

"DEVICE\_IP" VARCHAR2(50 CHAR),

"DEVICE\_ID" NUMBER,

"DEVICE\_VENDOR" VARCHAR2(50 CHAR),

"DEVICE\_MODEL" VARCHAR2(50 CHAR),

"DEVICE\_PARTITION" VARCHAR2(50 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"MATCH8" CHAR(1 BYTE),

"FULL\_MATCH" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"HOST\_NAME" VARCHAR2(50 CHAR),

"DEVICE\_IP" VARCHAR2(50 CHAR),

"DEVICE\_ID" NUMBER,

"DEVICE\_VENDOR" VARCHAR2(50 CHAR),

"DEVICE\_MODEL" VARCHAR2(50 CHAR),

"DEVICE\_PARTITION" VARCHAR2(50 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JUNIPER\_VLANS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JUNIPER\_VLANS"

( "HOST\_NAME" VARCHAR2(50 BYTE),

"DEVICE\_IP" VARCHAR2(20 BYTE),

"VLAN\_ID" NUMBER(10,0),

"VLAN\_NAME" VARCHAR2(50 BYTE),

"VLAN\_TYPE" VARCHAR2(50 BYTE),

"PRIVATE\_VLAN" VARCHAR2(50 BYTE),

"VLAN\_DESCRIPTION" VARCHAR2(50 BYTE),

"VLAN\_PARTITION" VARCHAR2(50 BYTE),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table JUNIPER\_XNG\_EQUIP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."JUNIPER\_XNG\_EQUIP"

( "NE\_INST\_ID" NUMBER,

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS\_TYPE" VARCHAR2(5 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(30 CHAR),

"TYPE" VARCHAR2(30 CHAR),

"STATUS" VARCHAR2(20 CHAR),

"VENDOR" VARCHAR2(40 CHAR),

"EQ\_CLASS" CHAR(1 CHAR),

"PARSE\_STATUS" VARCHAR2(250 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LDAP\_USERS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LDAP\_USERS"

( "USWIN\_USER\_ID" VARCHAR2(50 BYTE),

"USWIN\_FULLNAME" VARCHAR2(50 BYTE),

"USWIN\_FIRSTNAME" VARCHAR2(50 BYTE),

"USWIN\_LASTNAME" VARCHAR2(50 BYTE),

"USWIN\_TITLE" VARCHAR2(50 BYTE),

"USWIN\_DEPT" VARCHAR2(50 BYTE),

"USWIN\_TEL" VARCHAR2(50 BYTE),

"USWIN\_EMAIL" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LDAP\_USERS\_MANUAL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LDAP\_USERS\_MANUAL"

( "USWIN\_USER\_ID" VARCHAR2(50 BYTE),

"XNG\_FULLNAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LDAP\_XNG\_MATCH\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LDAP\_XNG\_MATCH\_WRK"

( "USWIN\_USER\_ID" VARCHAR2(50 BYTE),

"USER\_INST\_ID" NUMBER(9,0),

"USER\_NAME" VARCHAR2(50 BYTE),

"FULL\_NAME" VARCHAR2(50 BYTE),

"DEPARTMENT" VARCHAR2(30 BYTE),

"TELEPHONE" VARCHAR2(30 BYTE),

"EMAIL\_ADDR" VARCHAR2(200 BYTE),

"COMMENTS" VARCHAR2(255 BYTE),

"FLAGS" VARCHAR2(10 BYTE),

"MODIFY\_XNG" CHAR(1 BYTE),

"MATCH\_TYPE" NUMBER(6,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LDAP\_XNG\_MISMATCH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LDAP\_XNG\_MISMATCH"

( "USWIN\_USER\_ID" VARCHAR2(50 BYTE),

"USER\_INST\_ID" NUMBER(9,0),

"USER\_NAME" VARCHAR2(50 BYTE),

"FULL\_NAME" VARCHAR2(50 BYTE),

"DEPARTMENT" VARCHAR2(30 BYTE),

"TELEPHONE" VARCHAR2(30 BYTE),

"EMAIL\_ADDR" VARCHAR2(200 BYTE),

"COMMENTS" VARCHAR2(255 BYTE),

"FLAGS" VARCHAR2(10 BYTE),

"MODIFY\_XNG" CHAR(1 BYTE),

"UPD\_DONE" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LDAP\_XNG\_MISMATCH\_DUPS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LDAP\_XNG\_MISMATCH\_DUPS"

( "USWIN\_USER\_ID" VARCHAR2(50 BYTE),

"USER\_INST\_ID" NUMBER(9,0),

"USER\_NAME" VARCHAR2(50 BYTE),

"FULL\_NAME" VARCHAR2(50 BYTE),

"DEPARTMENT" VARCHAR2(30 BYTE),

"TELEPHONE" VARCHAR2(30 BYTE),

"EMAIL\_ADDR" VARCHAR2(200 BYTE),

"COMMENTS" VARCHAR2(255 BYTE),

"FLAGS" VARCHAR2(10 BYTE),

"MODIFY\_XNG" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LEASED\_SEG\_DF\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"DF\_SITE\_ID" NUMBER(10,0),

"DF\_SITE\_NAME" VARCHAR2(100 BYTE),

"NDF\_SEG\_ID" NUMBER(10,0),

"NDF\_SEG\_NAME" VARCHAR2(60 BYTE),

"NDF\_SEG\_STATUS" VARCHAR2(20 BYTE),

"NDF\_SEG\_TYPE" VARCHAR2(30 BYTE),

"NDF\_SEG\_BW" VARCHAR2(30 BYTE),

"IN\_SERVICE\_DATE" DATE,

"BTP\_EXTRACT\_DATE" DATE,

"UDA\_DISCONNECT\_DATE" DATE,

"NDF\_PATH\_ID" NUMBER(10,0),

"NDF\_PATH\_NAME" VARCHAR2(100 BYTE),

"NDF\_PATH\_STATUS" VARCHAR2(20 BYTE),

"NDF\_PATH\_BW" VARCHAR2(30 BYTE),

"TRANSPORT\_ACTION\_UDA" VARCHAR2(100 BYTE),

"COMP\_DATE\_UDA" DATE,

"ACTION\_JUST\_UDA" VARCHAR2(100 BYTE),

"IS\_EXEMPT" CHAR(1 BYTE),

"DF\_SEG\_ID" NUMBER(10,0),

"DF\_SEG\_NAME" VARCHAR2(60 BYTE),

"DF\_SEG\_STATUS" VARCHAR2(20 BYTE),

"DF\_SEG\_BW" VARCHAR2(30 BYTE),

"NDF\_SEG\_VENDOR" VARCHAR2(100 BYTE),

"DF\_SEG\_VENDOR" VARCHAR2(100 BYTE),

"NDF\_PATH\_TYPE" VARCHAR2(100 BYTE),

"DF\_RECUR\_COST" VARCHAR2(30 BYTE),

"DF\_NON\_RECUR\_COST" VARCHAR2(30 BYTE),

"NDF\_RECUR\_COST" VARCHAR2(30 BYTE),

"NDF\_NON\_RECUR\_COST" VARCHAR2(30 BYTE),

"DF\_SEG\_TYPE" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LEASED\_SEG\_DF\_DETAILS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"DF\_SITE\_ID" NUMBER(10,0),

"DF\_SITE\_NAME" VARCHAR2(100 BYTE),

"NDF\_SEG\_ID" NUMBER(10,0),

"NDF\_SEG\_NAME" VARCHAR2(60 BYTE),

"NDF\_SEG\_STATUS" VARCHAR2(20 BYTE),

"NDF\_SEG\_TYPE" VARCHAR2(30 BYTE),

"NDF\_SEG\_BW" VARCHAR2(30 BYTE),

"IN\_SERVICE\_DATE" DATE,

"BTP\_EXTRACT\_DATE" DATE,

"UDA\_DISCONNECT\_DATE" DATE,

"NDF\_PATH\_ID" NUMBER(10,0),

"NDF\_PATH\_NAME" VARCHAR2(100 BYTE),

"NDF\_PATH\_STATUS" VARCHAR2(20 BYTE),

"NDF\_PATH\_BW" VARCHAR2(30 BYTE),

"TRANSPORT\_ACTION\_UDA" VARCHAR2(100 BYTE),

"COMP\_DATE\_UDA" DATE,

"ACTION\_JUST\_UDA" VARCHAR2(100 BYTE),

"IS\_EXEMPT" CHAR(1 BYTE),

"DF\_SEG\_ID" NUMBER(10,0),

"DF\_SEG\_NAME" VARCHAR2(60 BYTE),

"DF\_SEG\_STATUS" VARCHAR2(20 BYTE),

"DF\_SEG\_BW" VARCHAR2(30 BYTE),

"NDF\_SEG\_VENDOR" VARCHAR2(100 BYTE),

"DF\_SEG\_VENDOR" VARCHAR2(100 BYTE),

"NDF\_PATH\_TYPE" VARCHAR2(100 BYTE),

"DF\_RECUR\_COST" VARCHAR2(30 BYTE),

"DF\_NON\_RECUR\_COST" VARCHAR2(30 BYTE),

"NDF\_RECUR\_COST" VARCHAR2(30 BYTE),

"NDF\_NON\_RECUR\_COST" VARCHAR2(30 BYTE),

"DF\_SEG\_TYPE" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LEASED\_SEG\_DF\_NDF\_PATH\_UDAS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_NDF\_PATH\_UDAS\_WK"

( "NDF\_PATH\_ID" NUMBER(10,0),

"TRANSPORT\_ACTION\_UDA" VARCHAR2(100 BYTE),

"COMP\_DATE\_UDA" DATE,

"ACTION\_JUST\_UDA" VARCHAR2(100 BYTE),

"IS\_EXEMPT" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LEASED\_SEG\_DF\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_NDF\_SEGMENTS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LEASED\_SEG\_DF\_SUMMARY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_SUMMARY\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_NDF\_SEGMENTS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJECT\_NL\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJECT\_NL\_PATHS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"PATH\_INST\_ID" NUMBER(9,0),

"PATH\_HUM\_ID" VARCHAR2(200 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"CATEGORY" VARCHAR2(20 BYTE),

"OBJECT\_TYPE" VARCHAR2(50 BYTE),

"SEQUENCE" VARCHAR2(20 BYTE),

"OBJ\_INST\_ID" NUMBER(9,0),

"OBJ\_HUM\_ID" VARCHAR2(200 BYTE),

"OBJ\_STATUS" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(100 BYTE),

"MRC" NUMBER(10,2),

"EXTRACT\_DATE" DATE,

"PATH\_TYPE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJECT\_NL\_PATHS\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJECT\_NL\_PATHS\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"SEGMENT\_COUNT" NUMBER(9,0),

"CABLE\_COUNT" NUMBER(9,0),

"PATH\_ELEMENT\_COUNT" NUMBER(9,0),

"EQUIP\_COUNT" NUMBER(9,0),

"EXTRACT\_DATE" DATE,

"SEGMENT\_EBH\_COUNT" NUMBER(9,0),

"CABLE\_EBH\_COUNT" NUMBER(9,0),

"PATH\_ELEMENT\_EBH\_COUNT" NUMBER(9,0),

"EQUIP\_EBH\_COUNT" NUMBER(9,0),

"SEGMENT\_NEBH\_COUNT" NUMBER(9,0),

"CABLE\_NEBH\_COUNT" NUMBER(9,0),

"PATH\_ELEMENT\_NEBH\_COUNT" NUMBER(9,0),

"EQUIP\_NEBH\_COUNT" NUMBER(9,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJECT\_NL\_PATHS\_TMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJECT\_NL\_PATHS\_TMP"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"PATH\_INST\_ID" NUMBER(9,0),

"PATH\_HUM\_ID" VARCHAR2(200 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"CATEGORY" VARCHAR2(20 BYTE),

"OBJECT\_TYPE" VARCHAR2(50 BYTE),

"SEQUENCE" VARCHAR2(20 BYTE),

"OBJ\_INST\_ID" NUMBER(9,0),

"OBJ\_HUM\_ID" VARCHAR2(200 BYTE),

"OBJ\_STATUS" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(100 BYTE),

"MRC" NUMBER(10,2),

"EXTRACT\_DATE" DATE,

"PATH\_TYPE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJECT\_NL\_PATHS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJECT\_NL\_PATHS\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"PATH\_INST\_ID" NUMBER(9,0),

"PATH\_HUM\_ID" VARCHAR2(200 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"CATEGORY" VARCHAR2(20 BYTE),

"OBJECT\_TYPE" VARCHAR2(50 BYTE),

"SEQUENCE" VARCHAR2(20 BYTE),

"OBJ\_INST\_ID" NUMBER(9,0),

"OBJ\_HUM\_ID" VARCHAR2(200 BYTE),

"OBJ\_STATUS" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(100 BYTE),

"MRC" NUMBER(10,2),

"EXTRACT\_DATE" DATE,

"PATH\_TYPE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJ\_NL\_SITES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"MARKET" VARCHAR2(50 CHAR),

"EQUIP\_CNT" NUMBER(10,0) DEFAULT 0,

"SEG\_CNT" NUMBER(10,0) DEFAULT 0,

"PATH\_CNT" NUMBER(10,0) DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJ\_NL\_SITES\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMMARY"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"MARKET" VARCHAR2(50 CHAR),

"EQUIP\_CNT" NUMBER(10,0) DEFAULT 0,

"SEG\_CNT" NUMBER(10,0) DEFAULT 0,

"PATH\_CNT" NUMBER(10,0) DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJ\_NL\_SITES\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMM\_WK"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"MARKET" VARCHAR2(50 CHAR),

"EQUIP\_CNT" NUMBER(10,0),

"SEG\_CNT" NUMBER(10,0),

"PATH\_CNT" NUMBER(10,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJ\_NL\_SITES\_SUMM\_WK\_PATH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMM\_WK\_PATH"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"MARKET" VARCHAR2(50 CHAR),

"PATH\_CNT" NUMBER(10,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LIVE\_OBJ\_NL\_SITES\_SUMM\_WK\_SEG

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMM\_WK\_SEG"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"MARKET" VARCHAR2(50 CHAR),

"SEG\_CNT" NUMBER(10,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LOC\_CAT\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LOC\_CAT\_DATA"

( "SITE\_NAME" VARCHAR2(200 BYTE),

"DOMAIN\_NAME" VARCHAR2(100 BYTE),

"LOCATION\_CATEGORY" VARCHAR2(50 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LR\_FTC\_2015

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LR\_FTC\_2015"

( "ORDER\_NUM" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 MAXSIZE UNLIMITED

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LR\_FTC\_2016

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LR\_FTC\_2016"

( "ORDER\_NUM" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 MAXSIZE UNLIMITED

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LR\_TMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LR\_TMP"

( "ORDER\_NUM" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LR\_TMP2

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LR\_TMP2"

( "ORDER\_NUM" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 MAXSIZE UNLIMITED

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LUCENT\_AUDIT\_REG\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LUCENT\_AUDIT\_REG\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_CSR\_MATCHED\_LIVE" NUMBER,

"EBH\_CSR\_MATCHED\_NONLIVE" NUMBER,

"EBH\_CSR\_REPORTED" NUMBER,

"EBH\_CSR\_COMP" NUMBER,

"OVERALL\_CSR\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LUCENT\_AUDIT\_REG\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LUCENT\_AUDIT\_REG\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_CSR\_MATCHED\_LIVE" NUMBER,

"EBH\_CSR\_MATCHED\_NONLIVE" NUMBER,

"EBH\_CSR\_REPORTED" NUMBER,

"EBH\_CSR\_COMP" NUMBER,

"OVERALL\_CSR\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LUCENT\_AUDIT\_SWITCH\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LUCENT\_AUDIT\_SWITCH\_SUMM"

( "REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SWITCH\_ID" VARCHAR2(8 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_CSR\_MATCHED\_LIVE" NUMBER,

"EBH\_CSR\_MATCHED\_NONLIVE" NUMBER,

"EBH\_CSR\_REPORTED" NUMBER,

"EBH\_CSR\_COMP" NUMBER,

"OVERALL\_CSR\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LUCENT\_AUDIT\_SWITCH\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LUCENT\_AUDIT\_SWITCH\_SUMM\_WRK"

( "REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SWITCH\_ID" VARCHAR2(8 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_CSR\_MATCHED\_LIVE" NUMBER,

"EBH\_CSR\_MATCHED\_NONLIVE" NUMBER,

"EBH\_CSR\_REPORTED" NUMBER,

"EBH\_CSR\_COMP" NUMBER,

"OVERALL\_CSR\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LUCENT\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDIT"

( "NE\_BTS\_NAME" VARCHAR2(50 BYTE),

"NE\_CELL\_TYPE" VARCHAR2(10 BYTE),

"SYS\_ID" NUMBER,

"ECP\_SID" NUMBER,

"NE\_CELL\_NUM" NUMBER,

"NE\_URC\_NUM" NUMBER,

"NE\_DS1\_NUM" NUMBER,

"NE\_VERSION" VARCHAR2(50 BYTE),

"SWITCH\_ID" VARCHAR2(8 BYTE),

"SERVICE" VARCHAR2(50 BYTE),

"TERMINATION\_TYPE" VARCHAR2(8 BYTE),

"XNG\_CELL\_NUM" NUMBER,

"XNG\_URC\_NUM" NUMBER,

"XNG\_DS1\_NUM" NUMBER,

"CIRC\_PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(20 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(10 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(256 BYTE),

"AUDIT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LUCENT\_NE\_VS\_XNG\_AUDIT\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDIT\_WRK"

( "NE\_BTS\_NAME" VARCHAR2(50 BYTE),

"NE\_CELL\_TYPE" VARCHAR2(10 BYTE),

"SYS\_ID" NUMBER,

"ECP\_SID" NUMBER,

"NE\_CELL\_NUM" NUMBER,

"NE\_URC\_NUM" NUMBER,

"NE\_DS1\_NUM" NUMBER,

"NE\_VERSION" VARCHAR2(50 BYTE),

"SWITCH\_ID" VARCHAR2(8 BYTE),

"SERVICE" VARCHAR2(50 BYTE),

"TERMINATION\_TYPE" VARCHAR2(8 BYTE),

"XNG\_CELL\_NUM" NUMBER,

"XNG\_URC\_NUM" NUMBER,

"XNG\_DS1\_NUM" NUMBER,

"CIRC\_PATH\_INST\_ID" NUMBER,

"PATH\_NAME" VARCHAR2(100 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(20 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(10 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(256 BYTE),

"AUDIT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table LU\_NSP\_JUNK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."LU\_NSP\_JUNK"

( "NAME" VARCHAR2(30 BYTE),

"SID" VARCHAR2(30 BYTE),

"SWITCH" VARCHAR2(30 BYTE),

"BTS" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MANAGED\_TOWERS\_UPD2\_ALL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MANAGED\_TOWERS\_UPD2\_ALL"

( "XNG\_FIELDS" VARCHAR2(50 BYTE),

"SITE" VARCHAR2(50 BYTE),

"UPDATE\_YN" VARCHAR2(50 BYTE),

"ASR\_REGISTRATION\_NUMBER" VARCHAR2(50 BYTE),

"LIGHTING\_REQUIRED" VARCHAR2(50 BYTE),

"COLUMN\_F" VARCHAR2(50 BYTE),

"LIGHTING\_INFORMATION" VARCHAR2(50 BYTE),

"FAA\_TOWER\_PAINTING\_REQUIRED" VARCHAR2(50 BYTE),

"SUPPORT\_ST\_HEIGH" VARCHAR2(50 BYTE),

"HEIGHT\_AGL" VARCHAR2(50 BYTE),

"HEIGHT\_AMSL" VARCHAR2(50 BYTE),

"NWF\_ST\_ID" VARCHAR2(50 BYTE),

"OWNER" VARCHAR2(50 BYTE),

"NAD\_83\_LAT" VARCHAR2(50 BYTE),

"NAD\_83\_LONG" VARCHAR2(50 BYTE),

"ADDRESS" VARCHAR2(50 BYTE),

"DUP\_XNG\_SITE" CHAR(1 BYTE),

"XNG\_UPDATED" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MARKET\_PREFIX\_ENB\_NAMING

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MARKET\_PREFIX\_ENB\_NAMING"

( "MARKET\_ID" VARCHAR2(6 BYTE),

"LTE\_MARKET\_NAME" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"VENDOR" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MICROWAVE\_UTIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PATH\_NAME" VARCHAR2(100 BYTE),

"PATH\_BANDWIDTH" VARCHAR2(30 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"PATH\_CATEGORY" VARCHAR2(30 BYTE),

"EQUIP\_MODEL" VARCHAR2(30 BYTE),

"EQUIP\_CATEGORY" VARCHAR2(30 BYTE),

"EQUIP\_VENDOR" VARCHAR2(40 BYTE),

"SITE\_A" VARCHAR2(100 BYTE),

"SITE\_LAT\_A" VARCHAR2(20 BYTE),

"SITE\_LONG\_A" VARCHAR2(20 BYTE),

"SITE\_Z" VARCHAR2(100 BYTE),

"SITE\_LAT\_Z" VARCHAR2(20 BYTE),

"SITE\_LONG\_Z" VARCHAR2(20 BYTE),

"CHANNELS\_ASSIGNED\_LONG" NUMBER(6,0),

"NBR\_CHANNELS" NUMBER(6,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MICROWAVE\_UTIL\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PATH\_NAME" VARCHAR2(100 BYTE),

"PATH\_BANDWIDTH" VARCHAR2(30 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"PATH\_CATEGORY" VARCHAR2(30 BYTE),

"EQUIP\_MODEL" VARCHAR2(30 BYTE),

"EQUIP\_CATEGORY" VARCHAR2(30 BYTE),

"EQUIP\_VENDOR" VARCHAR2(40 BYTE),

"SITE\_A" VARCHAR2(100 BYTE),

"SITE\_LAT\_A" VARCHAR2(20 BYTE),

"SITE\_LONG\_A" VARCHAR2(20 BYTE),

"SITE\_Z" VARCHAR2(100 BYTE),

"SITE\_LAT\_Z" VARCHAR2(20 BYTE),

"SITE\_LONG\_Z" VARCHAR2(20 BYTE),

"CHANNELS\_ASSIGNED\_LONG" NUMBER(6,0),

"NBR\_CHANNELS" NUMBER(6,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MONTHLY\_MRC\_BY\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION"

( "VENDOR\_NAME" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(100 BYTE),

"BAN\_CATEGORY1" VARCHAR2(100 BYTE),

"BAN\_CATEGORY2" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD1" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD3" VARCHAR2(100 BYTE),

"BAN" VARCHAR2(100 BYTE),

"STATE" VARCHAR2(2 BYTE),

"OCL" VARCHAR2(100 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"CHARGE\_AMOUNT\_JAN\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_FEB\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_MAR\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_APR\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_MAY\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JUN\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JUL\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_AUG\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_SEP\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_OCT\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_NOV\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_DEC\_2010" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JAN\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_FEB\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_MAR\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_APR\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_MAY\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JUN\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JUL\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_AUG\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_SEP\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_OCT\_2011" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_NOV\_2011" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MONTHLY\_MRC\_BY\_REGION\_FIRST

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION\_FIRST"

( "VENDOR\_NAME" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(100 BYTE),

"BAN\_CATEGORY1" VARCHAR2(100 BYTE),

"BAN\_CATEGORY2" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD1" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD3" VARCHAR2(100 BYTE),

"BAN" VARCHAR2(100 BYTE),

"STATE" VARCHAR2(2 BYTE),

"OCL" VARCHAR2(100 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"CHARGE\_AMOUNT\_JAN" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_FEB" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_MAR" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_APR" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_MAY" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JUN" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JUL" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_AUG" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_SEP" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_OCT" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_NOV" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_DEC" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_YEAR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MONTHLY\_MRC\_BY\_REGION\_OLD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION\_OLD"

( "VENDOR\_NAME" VARCHAR2(100 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(100 BYTE),

"BAN\_CATEGORY1" VARCHAR2(100 BYTE),

"BAN\_CATEGORY2" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD1" VARCHAR2(100 BYTE),

"BAN\_CUSTOM\_FIELD3" VARCHAR2(100 BYTE),

"BAN" VARCHAR2(100 BYTE),

"STATE" VARCHAR2(2 BYTE),

"OCL" VARCHAR2(100 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(100 BYTE),

"CHARGE\_AMOUNT\_JAN" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_FEB" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_MAR" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_APR" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_MAY" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JUN" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_JUL" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_AUG" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_SEP" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_OCT" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_NOV" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_DEC" VARCHAR2(20 BYTE),

"CHARGE\_AMOUNT\_YEAR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_BTS\_LOCATION\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_BTS\_LOCATION\_MAP"

( "BTS\_NUMBER" NUMBER,

"BTS\_NAME" VARCHAR2(50 BYTE),

"SANE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE,

"BTS\_TYPE" VARCHAR2(50 BYTE),

"CBSC\_NUMBER" NUMBER,

"LATITUDE" VARCHAR2(20 BYTE),

"LONGITUDE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_AUDIT\_DEV\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_AUDIT\_DEV\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"LEAF\_DOMAIN" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_MTX" VARCHAR2(100 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER(7,2) DEFAULT 0.00,

"EBH\_COMP" NUMBER(7,2) DEFAULT 0.00,

"OVERALL\_COMP" NUMBER(7,2) DEFAULT 0.00,

"EBH\_MATCHED\_NONLIVE\_CSR" NUMBER,

"EBH\_MATCHED\_LIVE\_CSR" NUMBER,

"UBS\_COUNT" NUMBER(\*,0),

"NON\_UBS\_COUNT" NUMBER(\*,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_AUDIT\_DEV\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_AUDIT\_DEV\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"LEAF\_DOMAIN" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_MTX" VARCHAR2(100 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER(7,2) DEFAULT 0.00,

"EBH\_COMP" NUMBER(7,2) DEFAULT 0.00,

"OVERALL\_COMP" NUMBER(7,2) DEFAULT 0.00,

"EBH\_MATCHED\_NONLIVE\_CSR" NUMBER,

"EBH\_MATCHED\_LIVE\_CSR" NUMBER,

"UBS\_COUNT" NUMBER(\*,0),

"NON\_UBS\_COUNT" NUMBER(\*,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_AUDIT\_REG\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_AUDIT\_REG\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER(7,2) DEFAULT 0.00,

"EBH\_COMP" NUMBER(7,2) DEFAULT 0.00,

"OVERALL\_COMP" NUMBER(7,2) DEFAULT 0.00,

"T1\_REPORTED" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_MATCHED\_NONLIVE\_CSR" NUMBER,

"EBH\_MATCHED\_LIVE\_CSR" NUMBER,

"UBS\_COUNT" NUMBER,

"NON\_UBS\_COUNT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_AUDIT\_REG\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_AUDIT\_REG\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER(7,2) DEFAULT 0.00,

"EBH\_COMP" NUMBER(7,2) DEFAULT 0.00,

"OVERALL\_COMP" NUMBER(7,2) DEFAULT 0.00,

"T1\_REPORTED" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_MATCHED\_NONLIVE\_CSR" NUMBER,

"EBH\_MATCHED\_LIVE\_CSR" NUMBER,

"UBS\_COUNT" NUMBER,

"NON\_UBS\_COUNT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_ETHERNET\_INV

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_ETHERNET\_INV"

( "BTS\_NUMBER" NUMBER,

"CLUSTER\_NUMBER" NUMBER,

"ROUTER\_GRP" NUMBER,

"BANDWIDTH" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT"

( "NE\_OMCR" VARCHAR2(50 BYTE),

"NE\_BTS\_NUMBER" NUMBER,

"NE\_CLUSTER\_NUMBER" NUMBER,

"NE\_BTS\_ROUTER\_GROUP" NUMBER,

"NE\_BANDWIDTH" VARCHAR2(50 BYTE),

"NE\_SPAN\_NUMBER" NUMBER,

"NE\_ACCESS\_NODE\_NUMBER" NUMBER,

"NE\_AG\_NODE\_NUMBER" NUMBER,

"NE\_AG\_SPAN\_NUMBER" NUMBER,

"NE\_T1\_LINE\_NUMBER" NUMBER,

"TERMINATION\_TYPE" VARCHAR2(8 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"MTX" VARCHAR2(100 BYTE),

"XNG\_PATH\_NAME" VARCHAR2(100 BYTE),

"XNG\_PATH\_INST\_ID" NUMBER,

"XNG\_BTS\_NUMBER" NUMBER,

"XNG\_CLUSTER\_NUMBER" NUMBER,

"XNG\_SPAN\_NUMBER" NUMBER,

"XNG\_BANDWIDTH" VARCHAR2(50 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(200 BYTE),

"EXTRACT\_DATE" DATE,

"AUDIT\_DATE" DATE,

"NE\_BTS\_TYPE" VARCHAR2(50 BYTE),

"NE\_BTS\_STATUS" VARCHAR2(20 BYTE),

"XNG\_IPBSCDO\_NUMBER" NUMBER,

"NE\_SPAN\_TYPE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK"

( "NE\_OMCR" VARCHAR2(50 BYTE),

"NE\_BTS\_NUMBER" NUMBER,

"NE\_CLUSTER\_NUMBER" NUMBER,

"NE\_BTS\_ROUTER\_GROUP" NUMBER,

"NE\_BANDWIDTH" VARCHAR2(50 BYTE),

"NE\_SPAN\_NUMBER" NUMBER,

"NE\_ACCESS\_NODE\_NUMBER" NUMBER,

"NE\_AG\_NODE\_NUMBER" NUMBER,

"NE\_AG\_SPAN\_NUMBER" NUMBER,

"NE\_T1\_LINE\_NUMBER" NUMBER,

"TERMINATION\_TYPE" VARCHAR2(8 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"MTX" VARCHAR2(100 BYTE),

"XNG\_PATH\_NAME" VARCHAR2(100 BYTE),

"XNG\_PATH\_INST\_ID" NUMBER,

"XNG\_BTS\_NUMBER" NUMBER,

"XNG\_CLUSTER\_NUMBER" NUMBER,

"XNG\_SPAN\_NUMBER" NUMBER,

"XNG\_BANDWIDTH" VARCHAR2(50 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(200 BYTE),

"EXTRACT\_DATE" DATE,

"AUDIT\_DATE" DATE,

"NE\_BTS\_TYPE" VARCHAR2(50 BYTE),

"NE\_BTS\_STATUS" VARCHAR2(20 BYTE),

"XNG\_IPBSCDO\_NUMBER" NUMBER,

"NE\_SPAN\_TYPE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_SPANS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_SPANS"

( "ID" NUMBER(\*,0),

"BTS\_NAME" VARCHAR2(50 BYTE),

"BTS\_NUMBER" NUMBER(\*,0),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"CLUSTER\_NUMBER" NUMBER(\*,0),

"BTS\_ROUTER\_GROUP" NUMBER(\*,0),

"SPAN\_NUMBER" NUMBER(\*,0),

"ACCESS\_NODE\_NUMBER" NUMBER(\*,0),

"AG\_NODE\_NUMBER" NUMBER(\*,0),

"AG\_SPAN\_NUMBER" NUMBER(\*,0),

"T1\_LINE\_NUMBER" NUMBER(\*,0),

"EXTRACT\_DATE" DATE,

"MCCDO\_IDENTIFIER" NUMBER(\*,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_CDMA\_T1\_INV

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_CDMA\_T1\_INV"

( "BTS\_NUMBER" NUMBER(\*,0),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"CLUSTER\_NUMBER" NUMBER(\*,0),

"BTS\_ROUTER\_GROUP" NUMBER(\*,0),

"SPAN\_NUMBER" NUMBER(\*,0),

"ACCESS\_NODE\_NUMBER" NUMBER(\*,0),

"AG\_NODE\_NUMBER" NUMBER(\*,0),

"AG\_SPAN\_NUMBER" NUMBER(\*,0),

"T1\_LINE\_NUMBER" NUMBER(\*,0),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_EVDO\_AUDIT\_DEV\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_EVDO\_AUDIT\_DEV\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"LEAF\_DOMAIN" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_AEMS" VARCHAR2(100 BYTE),

"IPBSCDO\_NUMBER" NUMBER,

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"EBH\_MATCHED\_NONLIVE\_CSR" NUMBER,

"EBH\_MATCHED\_LIVE\_CSR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_EVDO\_AUDIT\_DEV\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_EVDO\_AUDIT\_DEV\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"LEAF\_DOMAIN" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_AEMS" VARCHAR2(100 BYTE),

"IPBSCDO\_NUMBER" NUMBER,

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"EBH\_MATCHED\_NONLIVE\_CSR" NUMBER,

"EBH\_MATCHED\_LIVE\_CSR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_EVDO\_AUDIT\_REG\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_EVDO\_AUDIT\_REG\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"T1\_REPORTED" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_MATCHED\_NONLIVE\_CSR" NUMBER,

"EBH\_MATCHED\_LIVE\_CSR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_EVDO\_AUDIT\_REG\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_EVDO\_AUDIT\_REG\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"T1\_REPORTED" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_MATCHED\_NONLIVE\_CSR" NUMBER,

"EBH\_MATCHED\_LIVE\_CSR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_EVDO\_ETHERNET\_INV

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_EVDO\_ETHERNET\_INV"

( "VSM\_DEVICE\_NAME\_AEMS" VARCHAR2(50 BYTE),

"IPBSCDO\_NUMBER" NUMBER,

"MCCDO\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT"

( "AEMS" VARCHAR2(20 BYTE),

"NE\_IPBSCDO\_NUMBER" NUMBER,

"NE\_MCCDO\_IDENTIFIER" NUMBER,

"NE\_SPAN\_NUMBER" NUMBER(\*,0),

"NE\_STATUS" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(500 BYTE),

"XNG\_PATH\_NAME" VARCHAR2(100 BYTE),

"XNG\_PATH\_INST\_ID" NUMBER(\*,0),

"XNG\_PATH\_TYPE" VARCHAR2(20 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(20 BYTE),

"XNG\_IPBSCDO\_NUMBER" NUMBER,

"XNG\_MCCDO\_IDENTIFIER" NUMBER,

"XNG\_SPAN\_NUMBER" NUMBER(\*,0),

"EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"AUDIT\_DATE" DATE,

"XNG\_BANDWIDTH" VARCHAR2(50 BYTE),

"TERMINATION\_TYPE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT\_WRK"

( "AEMS" VARCHAR2(20 BYTE),

"NE\_IPBSCDO\_NUMBER" NUMBER,

"NE\_MCCDO\_IDENTIFIER" NUMBER,

"NE\_SPAN\_NUMBER" NUMBER(\*,0),

"NE\_STATUS" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(500 BYTE),

"XNG\_PATH\_NAME" VARCHAR2(100 BYTE),

"XNG\_PATH\_INST\_ID" NUMBER(\*,0),

"XNG\_PATH\_TYPE" VARCHAR2(20 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(20 BYTE),

"XNG\_IPBSCDO\_NUMBER" NUMBER,

"XNG\_MCCDO\_IDENTIFIER" NUMBER,

"XNG\_SPAN\_NUMBER" NUMBER(\*,0),

"EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(20 BYTE),

"AUDIT\_DATE" DATE,

"XNG\_BANDWIDTH" VARCHAR2(50 BYTE),

"TERMINATION\_TYPE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_EVDO\_T1\_INV

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_EVDO\_T1\_INV"

( "VSM\_DEVICE\_NAME\_AEMS" VARCHAR2(50 BYTE),

"IPBSCDO\_NUMBER" NUMBER,

"MCCDO\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER(\*,0),

"STATUS" VARCHAR2(30 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_OMCR\_MCCDO\_MAP\_SUM\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_OMCR\_MCCDO\_MAP\_SUM\_DEL"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(20 BYTE),

"BTS\_NUMBER" NUMBER(\*,0),

"MCCDO\_IDENTIFIER" NUMBER(\*,0),

"IPBSCDO\_NUMBER" NUMBER(\*,0),

"MATCH\_STATUS" VARCHAR2(20 BYTE),

"AUDIT\_DATE" DATE DEFAULT TRUNC( SYSDATE )

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_UBS\_PATHS\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_UBS\_PATHS\_MAP"

( "CIRC\_PATH\_INST\_ID" NUMBER,

"LEGACY\_PATH\_NAME" VARCHAR2(100 BYTE),

"STANDARD\_PATH\_NAME" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE DEFAULT SYSDATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MOTO\_VSM\_NAME\_AEMS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MOTO\_VSM\_NAME\_AEMS"

( "VSM\_DEVICE\_NAME\_AEMS" VARCHAR2(30 BYTE),

"IPBSCDO\_NUMBER" NUMBER(\*,0),

"IPBSCDO\_NAME" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MPLS\_NE\_VS\_XNG\_LINK\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MPLS\_NE\_VS\_XNG\_LINK\_AUDIT"

( "A\_ROUTER\_NAME" VARCHAR2(11 BYTE),

"A\_END\_PORT" VARCHAR2(89 BYTE),

"A\_PORT\_ACCESS\_ID" VARCHAR2(100 BYTE),

"AP1" VARCHAR2(89 BYTE),

"AP2" VARCHAR2(89 BYTE),

"AP3" VARCHAR2(89 BYTE),

"AP4" VARCHAR2(89 BYTE),

"AP5" VARCHAR2(89 BYTE),

"APX" VARCHAR2(89 BYTE),

"Z\_ROUTER\_NAME" VARCHAR2(11 BYTE),

"Z\_END\_PORT" VARCHAR2(89 BYTE),

"Z\_PORT\_ACCESS\_ID" VARCHAR2(100 BYTE),

"ZP1" VARCHAR2(89 BYTE),

"ZP2" VARCHAR2(89 BYTE),

"ZP3" VARCHAR2(89 BYTE),

"ZP4" VARCHAR2(89 BYTE),

"ZP5" VARCHAR2(89 BYTE),

"ZPX" VARCHAR2(89 BYTE),

"PATH\_LIVE\_IN\_XNG" CHAR(1 BYTE),

"A\_EQUIP" NUMBER(9,0),

"A\_PORT" NUMBER(9,0),

"A\_PORT\_NAME" VARCHAR2(50 BYTE),

"A\_PORT\_AID" VARCHAR2(100 BYTE),

"A\_PATH" NUMBER(9,0),

"A\_NEXT\_PATH" NUMBER(9,0),

"Z\_EQUIP" NUMBER(9,0),

"Z\_PORT" NUMBER(9,0),

"Z\_PORT\_NAME" VARCHAR2(50 BYTE),

"Z\_PORT\_AID" VARCHAR2(100 BYTE),

"Z\_PATH" NUMBER(9,0),

"Z\_NEXT\_PATH" NUMBER(9,0),

"MATCH\_CODE" VARCHAR2(7 BYTE),

"MATCH\_STATUS" VARCHAR2(49 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MPLS\_NE\_VS\_XNG\_RTRS\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MPLS\_NE\_VS\_XNG\_RTRS\_AUDIT"

( "ROUTER\_NAME" VARCHAR2(11 BYTE),

"CLLI" VARCHAR2(8 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"LIVE\_IN\_XNG" CHAR(1 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"EXTRACT\_DATE" DATE,

"MATCH\_CODE" VARCHAR2(30 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MPLS\_RTRS\_DETAILS\_TAB

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MPLS\_RTRS\_DETAILS\_TAB"

( "A\_RTR\_AREA" VARCHAR2(50 BYTE),

"A\_RTR\_REGION" VARCHAR2(50 BYTE),

"A\_LEAF\_DOMAIN" VARCHAR2(30 BYTE),

"A\_ROUTER\_NAME" VARCHAR2(11 BYTE),

"A\_END\_PORT" VARCHAR2(89 BYTE),

"Z\_RTR\_AREA" VARCHAR2(50 BYTE),

"Z\_RTR\_REGION" VARCHAR2(50 BYTE),

"Z\_LEAD\_DOMAIN" VARCHAR2(30 BYTE),

"Z\_ROUTER\_NAME" VARCHAR2(11 BYTE),

"Z\_END\_PORT" VARCHAR2(89 BYTE),

"MATCH\_CODE" VARCHAR2(7 BYTE),

"MATCH\_STATUS" VARCHAR2(49 BYTE),

"PATH\_LIVE\_IN\_XNG" CHAR(1 BYTE),

"A\_XNG\_EQUIP" VARCHAR2(100 BYTE),

"A\_XNG\_PORT" VARCHAR2(50 BYTE),

"A\_PORT\_BW" VARCHAR2(30 BYTE),

"A\_PATH\_NAME" VARCHAR2(100 BYTE),

"A\_PATH\_STATUS" VARCHAR2(20 BYTE),

"A\_NEXT\_PATH\_NAME" VARCHAR2(100 BYTE),

"A\_NEXT\_PATH\_STATUS" VARCHAR2(20 BYTE),

"Z\_XNG\_EQUIP" VARCHAR2(100 BYTE),

"Z\_XNG\_PORT" VARCHAR2(50 BYTE),

"Z\_PORT\_BW" VARCHAR2(30 BYTE),

"Z\_PATH\_NAME" VARCHAR2(100 BYTE),

"Z\_PATH\_STATUS" VARCHAR2(20 BYTE),

"Z\_NEXT\_PATH\_NAME" VARCHAR2(100 BYTE),

"Z\_NEXT\_PATH\_STATUS" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MPLS\_SUMMARY\_BY\_DEVICE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MPLS\_SUMMARY\_BY\_DEVICE"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN" VARCHAR2(30 BYTE),

"ROUTER\_NAME" VARCHAR2(11 BYTE),

"DISTINCT\_RTR\_MATCH\_IN\_XNG" CHAR(1 BYTE),

"DUP\_PORT" CHAR(1 BYTE),

"RTR\_LIVE\_IN\_XNG" CHAR(1 BYTE),

"TOTAL\_LINKS" NUMBER,

"LINKS\_LIVE\_IN\_SAME\_PATH" NUMBER,

"LINKS\_NONLIVE\_IN\_SAME\_PATH" NUMBER,

"LIVE\_COMPLETE\_LINKS" NUMBER,

"NON\_LIVE\_COMPLETE\_LINKS" NUMBER,

"COMPLIANCE\_PERC" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MPLS\_SUMMARY\_BY\_DEVICE\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MPLS\_SUMMARY\_BY\_DEVICE\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN" VARCHAR2(30 BYTE),

"ROUTER\_NAME" VARCHAR2(11 BYTE),

"DISTINCT\_RTR\_MATCH\_IN\_XNG" CHAR(1 BYTE),

"DUP\_PORT" CHAR(1 BYTE),

"RTR\_LIVE\_IN\_XNG" CHAR(1 BYTE),

"TOTAL\_LINKS" NUMBER,

"LINKS\_LIVE\_IN\_SAME\_PATH" NUMBER,

"LINKS\_NONLIVE\_IN\_SAME\_PATH" NUMBER,

"LIVE\_COMPLETE\_LINKS" NUMBER,

"NON\_LIVE\_COMPLETE\_LINKS" NUMBER,

"COMPLIANCE\_PERC" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MPLS\_SUMMARY\_BY\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MPLS\_SUMMARY\_BY\_REGION"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_ROUTERS" NUMBER,

"LIVE\_IN\_XNG" NUMBER,

"OTHER\_STATUS\_IN\_XNG" NUMBER,

"MATCHED\_IN\_XNG\_PERC" NUMBER,

"TOTAL\_PORTS" NUMBER,

"BOTH\_ENDS\_IN\_SAME\_LIVE\_PATH" NUMBER,

"BOTH\_ENDS\_IN\_SAME\_N\_LIVE\_PATH" NUMBER,

"MATCHED\_PORTS\_PERC" NUMBER,

"LIVE\_COMPLETE\_PATHS" NUMBER,

"NON\_LIVE\_COMPLETE\_PATHS" NUMBER,

"COMPLIANCE\_PERC" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MPLS\_SUMMARY\_BY\_REGION\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MPLS\_SUMMARY\_BY\_REGION\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_ROUTERS" NUMBER,

"LIVE\_IN\_XNG" NUMBER,

"OTHER\_STATUS\_IN\_XNG" NUMBER,

"MATCHED\_IN\_XNG\_PERC" NUMBER,

"TOTAL\_PORTS" NUMBER,

"BOTH\_ENDS\_IN\_SAME\_LIVE\_PATH" NUMBER,

"BOTH\_ENDS\_IN\_SAME\_N\_LIVE\_PATH" NUMBER,

"MATCHED\_PORTS\_PERC" NUMBER,

"LIVE\_COMPLETE\_PATHS" NUMBER,

"NON\_LIVE\_COMPLETE\_PATHS" NUMBER,

"COMPLIANCE\_PERC" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MSPP\_CIRC\_PATH\_ELEMENTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MSPP\_CIRC\_PATH\_ELEMENTS"

( "CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"LEG\_INST\_ID" NUMBER(9,0),

"MEMBER\_NBR" NUMBER(6,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"PREV\_PATH\_INST\_ID" NUMBER(9,0),

"NE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"VENDOR" VARCHAR2(30 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"NODE\_ID" VARCHAR2(200 BYTE),

"TID" VARCHAR2(30 BYTE),

"PATH\_ELEMENTS" VARCHAR2(4000 BYTE),

"MSPP\_PATH\_ELEMENTS" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MSPP\_CIRC\_PATH\_ELEMENTS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MSPP\_CIRC\_PATH\_ELEMENTS\_WK"

( "CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"LEG\_INST\_ID" NUMBER(9,0),

"MEMBER\_NBR" NUMBER(6,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"PREV\_PATH\_INST\_ID" NUMBER(9,0),

"NE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"VENDOR" VARCHAR2(30 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"NODE\_ID" VARCHAR2(200 BYTE),

"TID" VARCHAR2(30 BYTE),

"PATH\_ELEMENTS" VARCHAR2(4000 BYTE),

"MSPP\_PATH\_ELEMENTS" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MSPP\_SUMMARY\_REGION\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"XNG\_SITE\_INST\_ID" NUMBER(9,0),

"XNG\_EQUIP\_INST\_ID" NUMBER(9,0),

"TID" VARCHAR2(30 BYTE),

"XNG\_DEVICE\_TYPE" VARCHAR2(30 BYTE),

"AUDIT\_STATUS" VARCHAR2(20 BYTE),

"AUDIT\_DATE" DATE,

"MATCHED\_XCONN\_COUNT" NUMBER(9,0),

"DISCREPANCY\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"MATCHED\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"DESCRIPTION" VARCHAR2(300 BYTE),

"STATUS" VARCHAR2(30 BYTE),

"CARDTYPE" VARCHAR2(50 BYTE),

"IP\_ADDRESS" VARCHAR2(15 BYTE),

"EQUIP\_STATUS" VARCHAR2(30 BYTE),

"LIVE\_COUNT" NUMBER,

"NOT\_LIVE\_COUNT" NUMBER,

"NAMING\_STANDARD" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MSPP\_SUMMARY\_REGION\_MAP\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"XNG\_SITE\_INST\_ID" NUMBER(9,0),

"XNG\_EQUIP\_INST\_ID" NUMBER(9,0),

"TID" VARCHAR2(30 BYTE),

"XNG\_DEVICE\_TYPE" VARCHAR2(30 BYTE),

"AUDIT\_STATUS" VARCHAR2(20 BYTE),

"AUDIT\_DATE" DATE,

"MATCHED\_XCONN\_COUNT" NUMBER(9,0),

"DISCREPANCY\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"MATCHED\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"DESCRIPTION" VARCHAR2(300 BYTE),

"STATUS" VARCHAR2(30 BYTE),

"CARDTYPE" VARCHAR2(50 BYTE),

"IP\_ADDRESS" VARCHAR2(15 BYTE),

"EQUIP\_STATUS" VARCHAR2(30 BYTE),

"LIVE\_COUNT" NUMBER,

"NOT\_LIVE\_COUNT" NUMBER,

"NAMING\_STANDARD" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MSPP\_VSM\_MATCH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MSPP\_VSM\_MATCH"

( "VSM\_NODE\_ID" VARCHAR2(100 BYTE),

"TID" VARCHAR2(50 BYTE),

"NETYPE" VARCHAR2(50 BYTE),

"CARDTYPE" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"PARENT\_DOMAIN\_INST\_ID" NUMBER,

"AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MSPP\_VSM\_MATCH\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MSPP\_VSM\_MATCH\_WK"

( "VSM\_NODE\_ID" VARCHAR2(100 BYTE),

"TID" VARCHAR2(50 BYTE),

"NETYPE" VARCHAR2(50 BYTE),

"CARDTYPE" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"PARENT\_DOMAIN\_INST\_ID" NUMBER,

"AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_AEMS\_IPBSCDO

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_AEMS\_IPBSCDO"

( "SANE\_NAME\_AEMS" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_AEMS" VARCHAR2(50 BYTE),

"RAW\_STRING" VARCHAR2(50 BYTE),

"IPBSCDO\_NAME" VARCHAR2(50 BYTE),

"IPBSCDO\_NUMBER" NUMBER,

"STATUS" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT SYSDATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_AEMS\_OMCR\_EMH\_IP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_AEMS\_OMCR\_EMH\_IP"

( "SANE\_NAME\_AEMS" VARCHAR2(50 BYTE),

"DEVICE\_ID" VARCHAR2(50 BYTE),

"IP" VARCHAR2(50 BYTE),

"IPBSCDO\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE DEFAULT SYSDATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_AEMS\_OMCR\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_AEMS\_OMCR\_MAP"

( "SANE\_NAME\_AEMS" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_AEMS" VARCHAR2(50 BYTE),

"RAW\_STRING" VARCHAR2(50 BYTE),

"RAW\_NAME\_OMCR" VARCHAR2(50 BYTE),

"OMCR\_NAME" VARCHAR2(50 BYTE),

"OMCR\_NUMBER" NUMBER,

"STATUS" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT SYSDATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_CDMA\_AUDIT\_REG\_SUMMARY\_TDE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_CDMA\_AUDIT\_REG\_SUMMARY\_TDE"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_LIVE" NUMBER,

"T1\_N\_LIVE" NUMBER,

"ETHERNET\_DISCOVERED" NUMBER,

"ETHERNET\_LIVE" NUMBER,

"ETHERNET\_N\_LIVE" NUMBER,

"ETHERNET\_REPORTED" NUMBER,

"ETHERNET\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_EMH\_MCCDO\_BTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_EMH\_MCCDO\_BTS"

( "IPBSCDO\_NUMBER" NUMBER,

"MCCDO\_NUMBER" NUMBER,

"BTS\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_OMCR\_BTS\_CONCONF

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_OMCR\_BTS\_CONCONF"

( "SANE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"BTS\_NUMBER" NUMBER,

"BTS\_MLPPP\_NUMBER" NUMBER,

"BTS\_CON\_NUMBER" NUMBER,

"PBTS\_NUMBER" NUMBER,

"PBTS\_SPAN\_NUMBER" NUMBER,

"ACCESS\_NODE\_NUMBER" NUMBER,

"AG\_NODE\_NUMBER" NUMBER,

"AG\_SPAN\_NUMBER" NUMBER,

"T1\_LINE\_NUMBER" NUMBER,

"PKT\_MCCDO\_CON\_1\_NUMBER" NUMBER,

"PKT\_MCCDO\_CON\_2\_NUMBER" NUMBER,

"PKT\_MCCDO\_CON\_3\_NUMBER" NUMBER,

"PKT\_MCCDO\_CON\_4\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_OMCR\_BTS\_DO\_INFO

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_OMCR\_BTS\_DO\_INFO"

( "SANE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"BTS\_NUMBER" NUMBER,

"IPBSCDO\_NUMBER" NUMBER,

"HD\_MODEM\_DO\_ID" NUMBER,

"BSCDO\_FRAME" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT SYSDATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON TABLE "XNG\_REPORTS"."MT\_OMCR\_BTS\_DO\_INFO" IS 'This table is being used to determine which UBS sites have EvDO.';

--------------------------------------------------------

-- DDL for Table MT\_OMCR\_BTS\_RTR\_GRP\_CONF

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_OMCR\_BTS\_RTR\_GRP\_CONF"

( "SANE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"BTS\_NUMBER" NUMBER,

"BTS\_RTR\_CLUSTER" NUMBER,

"BTS\_RTR\_GRP" NUMBER,

"OTI\_BACKHAUL" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_OMCR\_BTS\_STATUS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_OMCR\_BTS\_STATUS"

( "SANE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"BTS\_NUMBER" NUMBER,

"BTS\_MLPPP\_NUMBER" NUMBER,

"BTS\_CON\_NUMBER" NUMBER,

"PBTS\_NUMBER" NUMBER,

"PBTS\_SPAN\_NUMBER" NUMBER,

"CBSC\_NUMBER" NUMBER,

"TEL\_STATE" VARCHAR2(20 BYTE),

"ADMIN\_STATE" VARCHAR2(20 BYTE),

"OP\_STATE" VARCHAR2(20 BYTE),

"USAGE\_STATE" VARCHAR2(20 BYTE),

"CONTROL\_STATUS" VARCHAR2(120 BYTE),

"AVAIL\_STATUS" VARCHAR2(20 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_OMCR\_IPS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_OMCR\_IPS"

( "SANE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"IP" VARCHAR2(50 BYTE),

"HOSTNAME" VARCHAR2(256 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table MT\_VSM\_SANE\_RAW\_OMCR\_NAME

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."MT\_VSM\_SANE\_RAW\_OMCR\_NAME"

( "SANE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"VSM\_DEVICE\_NAME\_OMCR" VARCHAR2(50 BYTE),

"OMCR\_RAW\_NAME" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table N2N\_OVERALL\_SUMM\_INTER

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."N2N\_OVERALL\_SUMM\_INTER"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LIVE" NUMBER,

"RPTD" NUMBER,

"ENB\_TOTAL\_COMPLIANCE" NUMBER,

"E2BTS\_TOTAL\_COMPLIANCE" NUMBER,

"EBH\_TOTAL\_COMPLIANCE" NUMBER,

"T1\_TOTAL\_COMPLIANCE" NUMBER,

"CSR\_TOTAL\_COMPLIANCE" NUMBER,

"OVERALL\_COMPLIANCE" NUMBER,

"REGIONAL\_MONTHLY\_DELTA" NUMBER,

"VLAN\_TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table N2N\_OVERALL\_SUMM\_INTER\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."N2N\_OVERALL\_SUMM\_INTER\_DEL"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LIVE" NUMBER,

"RPTD" NUMBER,

"ENB\_TOTAL\_COMPLIANCE" NUMBER,

"MSPP\_TOTAL\_COMPLIANCE" NUMBER,

"E2BTS\_TOTAL\_COMPLIANCE" NUMBER,

"CSR\_TOTAL\_COMPLIANCE" NUMBER,

"T1\_TOTAL\_COMPLIANCE" NUMBER,

"HPOV\_TOTAL\_COMPLIANCE" NUMBER,

"OVERALL\_COMPLIANCE" NUMBER,

"REGIONAL\_MONTHLY\_DELTA" NUMBER,

"VLAN\_TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table N2N\_OVERALL\_SUMM\_PREV\_MONTH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."N2N\_OVERALL\_SUMM\_PREV\_MONTH"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LIVE" NUMBER,

"RPTD" NUMBER,

"ENB\_TOTAL\_COMPLIANCE" NUMBER,

"E2BTS\_TOTAL\_COMPLIANCE" NUMBER,

"EBH\_TOTAL\_COMPLIANCE" NUMBER,

"T1\_TOTAL\_COMPLIANCE" NUMBER,

"CSR\_TOTAL\_COMPLIANCE" NUMBER,

"OVERALL\_COMPLIANCE" NUMBER,

"REGIONAL\_MONTHLY\_DELTA" NUMBER,

"VLAN\_TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table N2N\_OVERALL\_SUMM\_PREV\_MONTH\_D

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."N2N\_OVERALL\_SUMM\_PREV\_MONTH\_D"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LIVE" NUMBER,

"RPTD" NUMBER,

"ENB\_TOTAL\_COMPLIANCE" NUMBER,

"MSPP\_TOTAL\_COMPLIANCE" NUMBER,

"E2BTS\_TOTAL\_COMPLIANCE" NUMBER,

"CSR\_TOTAL\_COMPLIANCE" NUMBER,

"T1\_TOTAL\_COMPLIANCE" NUMBER,

"HPOV\_TOTAL\_COMPLIANCE" NUMBER,

"OVERALL\_COMPLIANCE" NUMBER,

"REGIONAL\_MONTHLY\_DELTA" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table N2N\_OVERALL\_SUMM\_PREV\_MTH0809

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."N2N\_OVERALL\_SUMM\_PREV\_MTH0809"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LIVE" NUMBER,

"RPTD" NUMBER,

"ENB\_TOTAL\_COMPLIANCE" NUMBER,

"E2BTS\_TOTAL\_COMPLIANCE" NUMBER,

"CSR\_TOTAL\_COMPLIANCE" NUMBER,

"T1\_TOTAL\_COMPLIANCE" NUMBER,

"HPOV\_TOTAL\_COMPLIANCE" NUMBER,

"OVERALL\_COMPLIANCE" NUMBER,

"REGIONAL\_MONTHLY\_DELTA" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CI\_CSR\_DEVICES\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CI\_CSR\_DEVICES\_DEL"

( "CSR\_DEVICE\_NAME" VARCHAR2(50 BYTE),

"IP" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CSR\_VLAN\_AUDITS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDITS"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_VENDOR" VARCHAR2(50 BYTE),

"NE\_HOSTNAME" VARCHAR2(100 BYTE),

"DISPLAYED\_NAME" VARCHAR2(50 BYTE),

"NE\_VLAN\_NUMBER" NUMBER,

"DEVICE\_IP" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(300 BYTE),

"XNG\_VLAN\_NUMBER" NUMBER,

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"AREA" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CLLI" VARCHAR2(50 BYTE),

"DESCRIPTION" VARCHAR2(250 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CSR\_VLAN\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDIT\_WK"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_VENDOR" VARCHAR2(50 BYTE),

"NE\_HOSTNAME" VARCHAR2(100 BYTE),

"DISPLAYED\_NAME" VARCHAR2(50 BYTE),

"NE\_VLAN\_NUMBER" NUMBER,

"DEVICE\_IP" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(300 BYTE),

"XNG\_VLAN\_NUMBER" NUMBER,

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"AREA" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CLLI" VARCHAR2(50 BYTE),

"DESCRIPTION" VARCHAR2(250 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CSR\_VLAN\_CLLI\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_CLLI\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CSR\_VLAN\_CLLI\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_CLLI\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CSR\_VLAN\_REGION\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_REGION\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CSR\_VLAN\_REGION\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_REGION\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CSR\_VLAN\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_WK"

( "HOSTNAME" VARCHAR2(100 BYTE),

"DEVICE\_IP" VARCHAR2(100 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"SUB\_INTERFACE\_NAME" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"PORT\_TYPE" VARCHAR2(100 BYTE),

"PORT\_IP" VARCHAR2(100 BYTE),

"DESCRIPTION" VARCHAR2(250 BYTE),

"PARTITION" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"PARSE\_STATUS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_CSR\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_CSR\_WK"

( "HOSTNAME" VARCHAR2(100 BYTE),

"DEVICE\_IP" VARCHAR2(100 BYTE),

"DEVICE\_VENDOR" VARCHAR2(100 BYTE),

"DEVICE\_MODEL" VARCHAR2(100 BYTE),

"PARTITION" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE,

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"LAST\_SUCCESSFUL\_SNAPSHOT" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_GI\_CI\_NGMLS\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_GI\_CI\_NGMLS\_AUDIT"

( "NGMLS\_VENDOR" CHAR(2 BYTE),

"NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NCM\_NGMLS\_HOSTNAME" VARCHAR2(100 BYTE),

"NCM\_NGMLS\_PARTITION" VARCHAR2(100 BYTE),

"NCM\_NGMLS\_DEVICE\_IP" VARCHAR2(100 BYTE),

"NCM\_NGMLS\_VLAN\_HOSTNAME" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(1000 BYTE),

"GI\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NE\_INST\_ID" NUMBER,

"LIVE\_IN\_XNG" CHAR(1 BYTE),

"EQ\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_GI\_CI\_NGMLS\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_GI\_CI\_NGMLS\_AUDIT\_WK"

( "NGMLS\_VENDOR" CHAR(2 BYTE),

"NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NCM\_NGMLS\_HOSTNAME" VARCHAR2(100 BYTE),

"NCM\_NGMLS\_PARTITION" VARCHAR2(100 BYTE),

"NCM\_NGMLS\_DEVICE\_IP" VARCHAR2(100 BYTE),

"NCM\_NGMLS\_VLAN\_HOSTNAME" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(1000 BYTE),

"GI\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NE\_INST\_ID" NUMBER,

"LIVE\_IN\_XNG" CHAR(1 BYTE),

"EQ\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_GI\_CI\_NGMLS\_REGION\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_GI\_CI\_NGMLS\_REGION\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NCM\_NGMLSS" NUMBER,

"NCM\_NGMLSS\_W\_VLAN" NUMBER,

"NCM\_NGMLSS\_WO\_VLAN" NUMBER,

"NCM\_NGMLSS\_W\_VLAN\_PER" NUMBER,

"GI\_NGMLSS" NUMBER,

"GI\_L\_NGMLSS" NUMBER,

"GI\_NL\_NGMLSS" NUMBER,

"NCM\_GI\_MATCH" NUMBER,

"NCM\_GI\_L\_MATCH" NUMBER,

"NCM\_GI\_NL\_MATCH" NUMBER,

"NCM\_GI\_MISMATCH" NUMBER,

"NCM\_GI\_PER" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_GI\_CI\_NGMLS\_REGION\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_GI\_CI\_NGMLS\_REGION\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NCM\_NGMLSS" NUMBER,

"NCM\_NGMLSS\_W\_VLAN" NUMBER,

"NCM\_NGMLSS\_WO\_VLAN" NUMBER,

"NCM\_NGMLSS\_W\_VLAN\_PER" NUMBER,

"GI\_NGMLSS" NUMBER,

"GI\_L\_NGMLSS" NUMBER,

"GI\_NL\_NGMLSS" NUMBER,

"NCM\_GI\_MATCH" NUMBER,

"NCM\_GI\_L\_MATCH" NUMBER,

"NCM\_GI\_NL\_MATCH" NUMBER,

"NCM\_GI\_MISMATCH" NUMBER,

"NCM\_GI\_PER" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT"

( "CSR\_VENDOR" CHAR(2 BYTE),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NCM\_CSR\_HOSTNAME" VARCHAR2(100 BYTE),

"NCM\_CSR\_PARTITION" VARCHAR2(100 BYTE),

"NCM\_CSR\_DEVICE\_IP" VARCHAR2(100 BYTE),

"NCM\_CSR\_VLAN\_HOSTNAME" VARCHAR2(100 BYTE),

"NCM\_CSR\_VLAN\_PARTITION" VARCHAR2(50 BYTE),

"NCM\_CSR\_VLAN\_DEVICE\_IP" VARCHAR2(100 BYTE),

"HP\_CSR\_HOSTNAME" VARCHAR2(100 BYTE),

"HP\_CSR\_IP" VARCHAR2(40 BYTE),

"GI\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 BYTE),

"LIVE\_IN\_XNG" CHAR(1 BYTE),

"MATCH\_CODE" VARCHAR2(9 BYTE),

"EH\_CSR\_HOSTNAME" VARCHAR2(100 BYTE),

"EH\_CSR\_IPADDRESS" VARCHAR2(100 BYTE),

"BRIX\_CSR\_SITE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK"

( "CSR\_VENDOR" CHAR(2 BYTE),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NCM\_CSR\_HOSTNAME" VARCHAR2(100 BYTE),

"NCM\_CSR\_PARTITION" VARCHAR2(100 BYTE),

"NCM\_CSR\_DEVICE\_IP" VARCHAR2(100 BYTE),

"NCM\_CSR\_VLAN\_HOSTNAME" VARCHAR2(100 BYTE),

"NCM\_CSR\_VLAN\_PARTITION" VARCHAR2(50 BYTE),

"NCM\_CSR\_VLAN\_DEVICE\_IP" VARCHAR2(100 BYTE),

"HP\_CSR\_HOSTNAME" VARCHAR2(100 BYTE),

"HP\_CSR\_IP" VARCHAR2(40 BYTE),

"GI\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 BYTE),

"LIVE\_IN\_XNG" CHAR(1 BYTE),

"MATCH\_CODE" VARCHAR2(9 BYTE),

"EH\_CSR\_HOSTNAME" VARCHAR2(100 BYTE),

"EH\_CSR\_IPADDRESS" VARCHAR2(100 BYTE),

"BRIX\_CSR\_SITE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CLLI\_6" VARCHAR2(6 BYTE),

"NCM\_CSRS" NUMBER,

"NCM\_CSRS\_W\_VLAN" NUMBER,

"HPOV\_CSRS" NUMBER,

"GI\_CSRS" NUMBER,

"GI\_L\_CSRS" NUMBER,

"GI\_NL\_CSRS" NUMBER,

"NCM\_CSRS\_W\_ISSUES" NUMBER,

"NCM\_CSRS\_W\_VLAN\_PER" NUMBER,

"NCM\_HPOV\_MATCH" NUMBER,

"NCM\_HPOV\_ISSUES" NUMBER,

"NCM\_HPOV\_PER" NUMBER,

"NCM\_GI\_MATCH" NUMBER,

"NCM\_GI\_L\_MATCH" NUMBER,

"NCM\_GI\_NL\_MATCH" NUMBER,

"NCM\_GI\_ISSUES" NUMBER,

"NCM\_GI\_PER" NUMBER,

"HPOV\_GI\_MATCH" NUMBER,

"HPOV\_GI\_L\_MATCH" NUMBER,

"HPOV\_GI\_NL\_MATCH" NUMBER,

"HPOV\_GI\_ISSUES" NUMBER,

"HPOV\_GI\_PER" NUMBER,

"AUDIT\_DATE" DATE,

"EH\_GI\_MATCH" NUMBER,

"EH\_GI\_L\_MATCH" NUMBER,

"EH\_GI\_NL\_MATCH" NUMBER,

"EH\_GI\_ISSUES" NUMBER,

"EH\_GI\_PER" NUMBER,

"EH\_CSRS" NUMBER,

"BRIX\_CSRS" NUMBER DEFAULT 0,

"BRIX\_GI\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_L\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_NL\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_ISSUES" NUMBER DEFAULT 0,

"BRIX\_GI\_PER" NUMBER DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CLLI\_6" VARCHAR2(6 BYTE),

"NCM\_CSRS" NUMBER,

"NCM\_CSRS\_W\_VLAN" NUMBER,

"HPOV\_CSRS" NUMBER,

"GI\_CSRS" NUMBER,

"GI\_L\_CSRS" NUMBER,

"GI\_NL\_CSRS" NUMBER,

"NCM\_CSRS\_W\_ISSUES" NUMBER,

"NCM\_CSRS\_W\_VLAN\_PER" NUMBER,

"NCM\_HPOV\_MATCH" NUMBER,

"NCM\_HPOV\_ISSUES" NUMBER,

"NCM\_HPOV\_PER" NUMBER,

"NCM\_GI\_MATCH" NUMBER,

"NCM\_GI\_L\_MATCH" NUMBER,

"NCM\_GI\_NL\_MATCH" NUMBER,

"NCM\_GI\_ISSUES" NUMBER,

"NCM\_GI\_PER" NUMBER,

"HPOV\_GI\_MATCH" NUMBER,

"HPOV\_GI\_L\_MATCH" NUMBER,

"HPOV\_GI\_NL\_MATCH" NUMBER,

"HPOV\_GI\_ISSUES" NUMBER,

"HPOV\_GI\_PER" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate,

"EH\_GI\_MATCH" NUMBER,

"EH\_GI\_L\_MATCH" NUMBER,

"EH\_GI\_NL\_MATCH" NUMBER,

"EH\_GI\_ISSUES" NUMBER,

"EH\_GI\_PER" NUMBER,

"EH\_CSRS" NUMBER,

"BRIX\_CSRS" NUMBER DEFAULT 0,

"BRIX\_GI\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_L\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_NL\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_ISSUES" NUMBER DEFAULT 0,

"BRIX\_GI\_PER" NUMBER DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NCM\_CSRS" NUMBER,

"NCM\_CSRS\_W\_VLAN" NUMBER,

"HPOV\_CSRS" NUMBER,

"GI\_CSRS" NUMBER,

"GI\_L\_CSRS" NUMBER,

"GI\_NL\_CSRS" NUMBER,

"NCM\_CSRS\_W\_ISSUES" NUMBER,

"NCM\_CSRS\_W\_VLAN\_PER" NUMBER,

"NCM\_HPOV\_MATCH" NUMBER,

"NCM\_HPOV\_ISSUES" NUMBER,

"NCM\_HPOV\_PER" NUMBER,

"NCM\_GI\_MATCH" NUMBER,

"NCM\_GI\_L\_MATCH" NUMBER,

"NCM\_GI\_NL\_MATCH" NUMBER,

"NCM\_GI\_ISSUES" NUMBER,

"NCM\_GI\_PER" NUMBER,

"HPOV\_GI\_MATCH" NUMBER,

"HPOV\_GI\_L\_MATCH" NUMBER,

"HPOV\_GI\_NL\_MATCH" NUMBER,

"HPOV\_GI\_ISSUES" NUMBER,

"HPOV\_GI\_PER" NUMBER,

"AUDIT\_DATE" DATE,

"EH\_CSRS" NUMBER,

"EH\_GI\_MATCH" NUMBER,

"EH\_GI\_L\_MATCH" NUMBER,

"EH\_GI\_NL\_MATCH" NUMBER,

"EH\_GI\_ISSUES" NUMBER,

"EH\_GI\_PER" NUMBER,

"BRIX\_CSRS" NUMBER DEFAULT 0,

"BRIX\_GI\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_L\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_NL\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_PER" NUMBER DEFAULT 0,

"BRIX\_GI\_ISSUES" NUMBER DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NCM\_CSRS" NUMBER,

"NCM\_CSRS\_W\_VLAN" NUMBER,

"HPOV\_CSRS" NUMBER,

"GI\_CSRS" NUMBER,

"GI\_L\_CSRS" NUMBER,

"GI\_NL\_CSRS" NUMBER,

"NCM\_CSRS\_W\_ISSUES" NUMBER,

"NCM\_CSRS\_W\_VLAN\_PER" NUMBER,

"NCM\_HPOV\_MATCH" NUMBER,

"NCM\_HPOV\_ISSUES" NUMBER,

"NCM\_HPOV\_PER" NUMBER,

"NCM\_GI\_MATCH" NUMBER,

"NCM\_GI\_L\_MATCH" NUMBER,

"NCM\_GI\_NL\_MATCH" NUMBER,

"NCM\_GI\_ISSUES" NUMBER,

"NCM\_GI\_PER" NUMBER,

"HPOV\_GI\_MATCH" NUMBER,

"HPOV\_GI\_L\_MATCH" NUMBER,

"HPOV\_GI\_NL\_MATCH" NUMBER,

"HPOV\_GI\_ISSUES" NUMBER,

"HPOV\_GI\_PER" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate,

"EH\_CSRS" NUMBER,

"EH\_GI\_MATCH" NUMBER,

"EH\_GI\_L\_MATCH" NUMBER,

"EH\_GI\_NL\_MATCH" NUMBER,

"EH\_GI\_ISSUES" NUMBER,

"EH\_GI\_PER" NUMBER,

"BRIX\_CSRS" NUMBER DEFAULT 0,

"BRIX\_GI\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_L\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_NL\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_PER" NUMBER DEFAULT 0,

"BRIX\_GI\_ISSUES" NUMBER DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_NGMLS\_VLAN\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_AUDIT"

( "NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_VENDOR" VARCHAR2(50 BYTE),

"NE\_HOSTNAME" VARCHAR2(100 BYTE),

"NE\_SUB\_INTERFACE\_NAME" VARCHAR2(50 BYTE),

"NE\_VLAN\_NUMBER" NUMBER,

"NE\_PORT\_IP" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"NE\_INST\_ID" NUMBER,

"XNG\_VLAN\_NUMBER" NUMBER,

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_NGMLS\_VLAN\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_AUDIT\_WK"

( "NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_VENDOR" VARCHAR2(50 BYTE),

"NE\_HOSTNAME" VARCHAR2(100 BYTE),

"NE\_SUB\_INTERFACE\_NAME" VARCHAR2(50 BYTE),

"NE\_VLAN\_NUMBER" NUMBER,

"NE\_PORT\_IP" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"NE\_INST\_ID" NUMBER,

"XNG\_VLAN\_NUMBER" NUMBER,

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_NGMLS\_VLAN\_DEVICE\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_DEVICE\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_VENDOR" VARCHAR2(2 BYTE),

"NE\_TOTAL\_NGMLS\_VLAN" NUMBER,

"MATCHED\_L\_NGMLS\_VLAN" NUMBER,

"MATCHED\_NL\_NGMLS\_VLAN" NUMBER,

"MISMATCHED\_NGMLS\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_VENDOR" VARCHAR2(2 BYTE),

"NE\_TOTAL\_NGMLS\_VLAN" NUMBER,

"MATCHED\_L\_NGMLS\_VLAN" NUMBER,

"MATCHED\_NL\_NGMLS\_VLAN" NUMBER,

"MISMATCHED\_NGMLS\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_NGMLS\_VLAN\_REGION\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_REGION\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NE\_TOTAL\_NGMLS\_VLAN" NUMBER,

"MATCHED\_L\_NGMLS\_VLAN" NUMBER,

"MATCHED\_NL\_NGMLS\_VLAN" NUMBER,

"MISMATCHED\_NGMLS\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_NGMLS\_VLAN\_REGION\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_REGION\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NE\_TOTAL\_NGMLS\_VLAN" NUMBER,

"MATCHED\_L\_NGMLS\_VLAN" NUMBER,

"MATCHED\_NL\_NGMLS\_VLAN" NUMBER,

"MISMATCHED\_NGMLS\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_NGMLS\_VLAN\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_WK"

( "HOSTNAME" VARCHAR2(100 BYTE),

"DEVICE\_IP" VARCHAR2(100 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"SUB\_INTERFACE\_NAME" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"PORT\_TYPE" VARCHAR2(100 BYTE),

"PORT\_IP" VARCHAR2(50 BYTE),

"DESCRIPTION" VARCHAR2(250 BYTE),

"PARTITION" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate,

"PARSE\_STATUS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NCM\_NGMLS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NCM\_NGMLS\_WK"

( "HOSTNAME" VARCHAR2(30 BYTE),

"DEVICE\_IP" VARCHAR2(30 BYTE),

"DEVICE\_VENDOR" VARCHAR2(30 BYTE),

"DEVICE\_MODEL" VARCHAR2(30 BYTE),

"PARTITION" VARCHAR2(30 BYTE),

"EXTRACT\_DATE" DATE,

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"LAST\_SUCCESSFUL\_SNAPSHOT" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_EQ\_SUMM\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_EQ\_SUMM\_REGION"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"TOTAL\_NE" NUMBER,

"DUPLICATE\_NE" NUMBER,

"MATCHED\_NE" NUMBER,

"PERCENT\_MATCHED" NUMBER,

"PASSED\_AUDIT" NUMBER,

"NAMING\_COMPLIANCE" NUMBER,

"DISCOVERED\_XCONNECTS" NUMBER,

"LIVE\_XCONNECTS" NUMBER,

"OTHER\_XCONNECTS" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_EQ\_SUMM\_REGION\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_EQ\_SUMM\_REGION\_WK"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"TOTAL\_NE" NUMBER,

"DUPLICATE\_NE" NUMBER,

"MATCHED\_NE" NUMBER,

"PERCENT\_MATCHED" NUMBER,

"PASSED\_AUDIT" NUMBER,

"NAMING\_COMPLIANCE" NUMBER,

"DISCOVERED\_XCONNECTS" NUMBER,

"LIVE\_XCONNECTS" NUMBER,

"OTHER\_XCONNECTS" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_FUJITSU\_EQUIPMENT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_FUJITSU\_EQUIPMENT"

( "TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"NE\_LEVEL" NUMBER(9,0),

"NE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS\_TYPE" VARCHAR2(20 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"TYPE" VARCHAR2(50 CHAR),

"STATUS" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"EQ\_CLASS" VARCHAR2(20 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"LOCATION" VARCHAR2(80 CHAR),

"CONTACTS" VARCHAR2(80 CHAR),

"IP\_ADDR" VARCHAR2(40 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"MATCH8" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"LOCATION" VARCHAR2(80 CHAR),

"CONTACTS" VARCHAR2(80 CHAR),

"IP\_ADDR" VARCHAR2(40 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"MATCH8" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"LOCATION" VARCHAR2(80 CHAR),

"CONTACTS" VARCHAR2(80 CHAR),

"IP\_ADDR" VARCHAR2(40 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"MATCH8" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_XCONNECT\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_XCONNECT\_AUDIT"

( "TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"MATCH\_CODE" VARCHAR2(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"XCONNECT\_ASIDE" VARCHAR2(80 CHAR),

"A\_SIDE\_EQUIP\_ID" NUMBER(9,0),

"A\_SIDE\_PATH\_INST\_ID" NUMBER(9,0),

"A\_SIDE\_LEG\_INST\_ID" NUMBER(9,0),

"A\_SIDE\_MEMBER\_NBR" NUMBER(9,0),

"XCONNECT\_ZSIDE" VARCHAR2(80 CHAR),

"Z\_SIDE\_EQUIP\_ID" NUMBER(9,0),

"Z\_SIDE\_PATH\_INST\_ID" NUMBER(9,0),

"Z\_SIDE\_LEG\_INST\_ID" NUMBER(9,0),

"Z\_SIDE\_MEMBER\_NBR" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 CHAR),

"PATH\_STATUS" VARCHAR2(20 CHAR),

"BANDWIDTH" VARCHAR2(30 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_XCONNECT\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_XCONNECT\_AUDIT\_WK"

( "TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"MATCH\_CODE" VARCHAR2(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"XCONNECT\_ASIDE" VARCHAR2(80 CHAR),

"A\_SIDE\_EQUIP\_ID" NUMBER(9,0),

"A\_SIDE\_PATH\_INST\_ID" NUMBER(9,0),

"A\_SIDE\_LEG\_INST\_ID" NUMBER(9,0),

"A\_SIDE\_MEMBER\_NBR" NUMBER(9,0),

"XCONNECT\_ZSIDE" VARCHAR2(80 CHAR),

"Z\_SIDE\_EQUIP\_ID" NUMBER(9,0),

"Z\_SIDE\_PATH\_INST\_ID" NUMBER(9,0),

"Z\_SIDE\_LEG\_INST\_ID" NUMBER(9,0),

"Z\_SIDE\_MEMBER\_NBR" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 CHAR),

"PATH\_STATUS" VARCHAR2(20 CHAR),

"BANDWIDTH" VARCHAR2(30 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_XCONNECT\_EQPT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_XCONNECT\_EQPT"

( "NE\_LEVEL" NUMBER(9,0),

"NE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS\_TYPE" VARCHAR2(20 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"TYPE" VARCHAR2(50 CHAR),

"STATUS" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"EQ\_CLASS" VARCHAR2(20 CHAR)

) SEGMENT CREATION DEFERRED

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE( INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_ADM\_XNG\_PORTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_ADM\_XNG\_PORTS"

( "TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"SLOT" VARCHAR2(50 CHAR),

"PORT\_HUM\_ID" VARCHAR2(50 CHAR),

"PORT\_ACCESS\_ID" VARCHAR2(100 CHAR),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"LEG\_INST\_ID" NUMBER(9,0),

"MEMBER\_NBR" NUMBER(9,0),

"A\_SIDE\_SITE\_ID" NUMBER(9,0),

"A\_SIDE\_SITE\_HUM\_ID" VARCHAR2(100 CHAR),

"Z\_SIDE\_SITE\_ID" NUMBER(9,0),

"Z\_SIDE\_SITE\_HUM\_ID" VARCHAR2(100 CHAR),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 CHAR),

"BANDWIDTH" VARCHAR2(30 CHAR),

"PATH\_STATUS" VARCHAR2(20 CHAR),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"PREV\_PATH\_INST\_ID" NUMBER(9,0),

"NE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"VENDOR" VARCHAR2(50 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"PORT\_INST\_ID" NUMBER(9,0),

"CARD\_INST\_ID" NUMBER(9,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_EQPT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_EQPT"

( "INST\_ID" NUMBER(10,0),

"SNCNE\_ID" NUMBER(10,0),

"COMPONENT\_ID" VARCHAR2(255 CHAR),

"COMPONENT\_TYPE" VARCHAR2(255 CHAR),

"ACTDCM" VARCHAR2(80 CHAR),

"ACTSL" VARCHAR2(80 CHAR),

"ACTTYPE" VARCHAR2(80 CHAR),

"AVAILBW" VARCHAR2(80 CHAR),

"BERSDFIC" VARCHAR2(80 CHAR),

"CACMODE" VARCHAR2(80 CHAR),

"CARDCURRFEED" VARCHAR2(80 CHAR),

"CASCADED" VARCHAR2(80 CHAR),

"CATEGORY" VARCHAR2(80 CHAR),

"CFGMODE" VARCHAR2(80 CHAR),

"CLEI" VARCHAR2(80 CHAR),

"COMPLTYPE" VARCHAR2(80 CHAR),

"CURRENTDRAWNFEED1" VARCHAR2(80 CHAR),

"CURRENTDRAWNFEED2" VARCHAR2(80 CHAR),

"DCCMODE" VARCHAR2(80 CHAR),

"DCMCHK" VARCHAR2(80 CHAR),

"DETWAVE" VARCHAR2(80 CHAR),

"DOM" VARCHAR2(80 CHAR),

"EXPHT2H" VARCHAR2(80 CHAR),

"EXPWAVE" VARCHAR2(80 CHAR),

"EXTOW" VARCHAR2(80 CHAR),

"FFPCONF" VARCHAR2(80 CHAR),

"FICTYPE" VARCHAR2(80 CHAR),

"FREQGRID" VARCHAR2(80 CHAR),

"FREQMAX" VARCHAR2(80 CHAR),

"FREQMIN" VARCHAR2(80 CHAR),

"FUSEFEED1" VARCHAR2(80 CHAR),

"FUSEFEED2" VARCHAR2(80 CHAR),

"GCADD" VARCHAR2(80 CHAR),

"HKIN" VARCHAR2(80 CHAR),

"HLOL" VARCHAR2(80 CHAR),

"LAMBDA" VARCHAR2(80 CHAR),

"LAMBDAINFO" VARCHAR2(80 CHAR),

"LANMODE" VARCHAR2(80 CHAR),

"LANTSLOT" VARCHAR2(80 CHAR),

"LANTYPE" VARCHAR2(80 CHAR),

"LASER" VARCHAR2(80 CHAR),

"LMBDGRID" VARCHAR2(80 CHAR),

"LMBDMAX" VARCHAR2(80 CHAR),

"LMBDMIN" VARCHAR2(80 CHAR),

"LOCATION" VARCHAR2(80 CHAR),

"MAID" VARCHAR2(80 CHAR),

"MANREF" VARCHAR2(80 CHAR),

"MGTSYS" VARCHAR2(80 CHAR),

"NSMODE" VARCHAR2(80 CHAR),

"OOF" VARCHAR2(80 CHAR),

"OWSEL" VARCHAR2(80 CHAR),

"POSTLASER" VARCHAR2(80 CHAR),

"PRELASER" VARCHAR2(80 CHAR),

"RAMAN1" VARCHAR2(80 CHAR),

"RAMAN2" VARCHAR2(80 CHAR),

"REGEN" VARCHAR2(80 CHAR),

"RLOLE1" VARCHAR2(80 CHAR),

"RLOLE2" VARCHAR2(80 CHAR),

"RLOLE3" VARCHAR2(80 CHAR),

"RLOLE4" VARCHAR2(80 CHAR),

"RLOLE5" VARCHAR2(80 CHAR),

"RLOLE6" VARCHAR2(80 CHAR),

"RLOLE7" VARCHAR2(80 CHAR),

"RLOLE8" VARCHAR2(80 CHAR),

"RVFA1" VARCHAR2(80 CHAR),

"RVFA2" VARCHAR2(80 CHAR),

"SCADD" VARCHAR2(80 CHAR),

"SERIALNO" VARCHAR2(80 CHAR),

"SFP1" VARCHAR2(80 CHAR),

"SFP2" VARCHAR2(80 CHAR),

"SFP3" VARCHAR2(80 CHAR),

"SFP4" VARCHAR2(80 CHAR),

"SFP5" VARCHAR2(80 CHAR),

"SFP6" VARCHAR2(80 CHAR),

"SFP7" VARCHAR2(80 CHAR),

"SFP8" VARCHAR2(80 CHAR),

"SHCNFIG1" VARCHAR2(80 CHAR),

"SHCNFIG2" VARCHAR2(80 CHAR),

"SHELFLBL" VARCHAR2(80 CHAR),

"SHFTYPE" VARCHAR2(80 CHAR),

"SMODE" VARCHAR2(80 CHAR),

"SSBITSEL" VARCHAR2(80 CHAR),

"SWPR" VARCHAR2(80 CHAR),

"SYSREF" VARCHAR2(80 CHAR),

"TCAREF" VARCHAR2(80 CHAR),

"TMGLS" VARCHAR2(80 CHAR),

"TMGOUTLS" VARCHAR2(80 CHAR),

"TRBOW" VARCHAR2(80 CHAR),

"TVFA1" VARCHAR2(80 CHAR),

"TVFA2" VARCHAR2(80 CHAR),

"TYPEINFO01" VARCHAR2(256 CHAR),

"UNITNAME" VARCHAR2(80 CHAR),

"USERLBL" VARCHAR2(80 CHAR),

"USI" VARCHAR2(80 CHAR),

"USRREF" VARCHAR2(80 CHAR),

"VENDID" VARCHAR2(80 CHAR),

"VOLTAGE" VARCHAR2(80 CHAR),

"WDMMODE" VARCHAR2(80 CHAR),

"STATUS1" VARCHAR2(80 CHAR),

"STATUS2" VARCHAR2(80 CHAR),

"LAST\_UPDATE" DATE,

"STATUS" VARCHAR2(80 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_EQ\_SUMM\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_EQ\_SUMM\_REGION"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"TOTAL\_NE" NUMBER,

"DUPLICATE\_NE" NUMBER,

"MATCHED\_NE" NUMBER,

"PERCENT\_MATCHED" NUMBER,

"PASSED\_AUDIT" NUMBER,

"NAMING\_COMPLIANCE" NUMBER,

"DISCOVERED\_XCONNECTS" NUMBER,

"LIVE\_XCONNECTS" NUMBER,

"OTHER\_XCONNECTS" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_EQ\_SUMM\_REGION\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_EQ\_SUMM\_REGION\_WK"

( "AREA" VARCHAR2(50 CHAR),

"REGION" VARCHAR2(50 CHAR),

"TOTAL\_NE" NUMBER,

"DUPLICATE\_NE" NUMBER,

"MATCHED\_NE" NUMBER,

"PERCENT\_MATCHED" NUMBER,

"PASSED\_AUDIT" NUMBER,

"NAMING\_COMPLIANCE" NUMBER,

"DISCOVERED\_XCONNECTS" NUMBER,

"LIVE\_XCONNECTS" NUMBER,

"OTHER\_XCONNECTS" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_FUJITSU\_EQUIPMENT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_FUJITSU\_EQUIPMENT"

( "TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"NE\_LEVEL" NUMBER(9,0),

"NE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS\_TYPE" VARCHAR2(20 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"TYPE" VARCHAR2(50 CHAR),

"STATUS" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"EQ\_CLASS" VARCHAR2(20 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_NE\_VS\_XNG\_AUDIT"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"LOCATION" VARCHAR2(80 CHAR),

"CONTACTS" VARCHAR2(80 CHAR),

"IP\_ADDR" VARCHAR2(40 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"MATCH8" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"LOCATION" VARCHAR2(80 CHAR),

"CONTACTS" VARCHAR2(80 CHAR),

"IP\_ADDR" VARCHAR2(40 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"MATCH8" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_NE\_VS\_XNG\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_NE\_VS\_XNG\_AUDIT\_WK"

( "LEAF\_DOMAIN\_NAME" VARCHAR2(80 CHAR),

"TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"LOCATION" VARCHAR2(80 CHAR),

"CONTACTS" VARCHAR2(80 CHAR),

"IP\_ADDR" VARCHAR2(40 CHAR),

"MATCH\_CODE" CHAR(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"NE\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"MATCH8" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_SNCNE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_SNCNE"

( "INST\_ID" NUMBER(9,0),

"TID" VARCHAR2(30 CHAR),

"EQUIP\_INST\_ID" NUMBER(9,0),

"TID\_TYPE" VARCHAR2(50 CHAR),

"AUTOMSG" VARCHAR2(20 CHAR),

"USERLOGIN" VARCHAR2(20 CHAR),

"INSERVICE" VARCHAR2(20 CHAR),

"LAST\_UPDATE" DATE,

"LOCATION" VARCHAR2(80 CHAR),

"CONTACTS" VARCHAR2(80 CHAR),

"IP\_ADDR" VARCHAR2(40 CHAR),

"IP\_SUBNET" VARCHAR2(40 CHAR),

"IP\_NETMASK" VARCHAR2(40 CHAR),

"IP\_DEFAULT\_GATEWAY" VARCHAR2(40 CHAR),

"IP\_PORT" VARCHAR2(20 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON COLUMN "XNG\_REPORTS"."NETSMART\_SNCNE"."AUTOMSG" IS 'Y (Autonomous messages from this NE are forwarded), N (Autonomous messages from this NE are not forwarded)';

COMMENT ON COLUMN "XNG\_REPORTS"."NETSMART\_SNCNE"."USERLOGIN" IS 'ACT (An uplink session is active for this NE), LOFF (No uplink session exists for this NE)';

COMMENT ON COLUMN "XNG\_REPORTS"."NETSMART\_SNCNE"."INSERVICE" IS 'ACT (The NetSmart has an active connection to this NE), LOFF (The NetSmart does not have an active connection to this NE)';

--------------------------------------------------------

-- DDL for Table NETSMART\_SNCNE\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT"

( "INST\_ID" NUMBER(9,0),

"TID" VARCHAR2(30 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"EQUIP\_INST\_ID" NUMBER(9,0),

"MATCH\_CODE" VARCHAR2(30 CHAR),

"MATCH\_STATUS" VARCHAR2(512 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_SNCNE\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT\_WK"

( "INST\_ID" NUMBER(9,0),

"TID" VARCHAR2(30 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"EQUIP\_INST\_ID" NUMBER(9,0),

"MATCH\_CODE" VARCHAR2(30 CHAR),

"MATCH\_STATUS" VARCHAR2(512 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_XCONNECT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_XCONNECT"

( "INST\_ID" NUMBER(10,0),

"SNCNE\_ID" NUMBER(10,0),

"XCONNECT\_ID" VARCHAR2(165 CHAR),

"XCONNECT\_ASIDE" VARCHAR2(80 CHAR),

"XCONNECT\_ZSIDE" VARCHAR2(80 CHAR),

"XCONNECT\_TYPE" VARCHAR2(80 CHAR),

"XCONNECT\_CKTID" VARCHAR2(80 CHAR),

"XCONNECT\_FROMT1" VARCHAR2(80 CHAR),

"XCONNECT\_NUT" VARCHAR2(80 CHAR),

"XCONNECT\_PSWDEF" VARCHAR2(80 CHAR),

"XCONNECT\_RDLNE" VARCHAR2(80 CHAR),

"XCONNECT\_RVRTV" VARCHAR2(80 CHAR),

"XCONNECT\_TAP" VARCHAR2(80 CHAR),

"XCONNECT\_TOT1" VARCHAR2(80 CHAR),

"XCONNECT\_WTR" VARCHAR2(80 CHAR),

"XCONNECT\_SOURCE" VARCHAR2(80 CHAR),

"LAST\_UPDATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_XCONNECT\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_XCONNECT\_AUDIT"

( "TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"MATCH\_CODE" VARCHAR2(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"XCONNECT\_ASIDE" VARCHAR2(80 CHAR),

"A\_SIDE\_EQUIP\_ID" NUMBER(9,0),

"A\_SIDE\_PATH\_INST\_ID" NUMBER(9,0),

"A\_SIDE\_LEG\_INST\_ID" NUMBER(9,0),

"A\_SIDE\_MEMBER\_NBR" NUMBER(9,0),

"XCONNECT\_ZSIDE" VARCHAR2(80 CHAR),

"Z\_SIDE\_EQUIP\_ID" NUMBER(9,0),

"Z\_SIDE\_PATH\_INST\_ID" NUMBER(9,0),

"Z\_SIDE\_LEG\_INST\_ID" NUMBER(9,0),

"Z\_SIDE\_MEMBER\_NBR" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 CHAR),

"PATH\_STATUS" VARCHAR2(20 CHAR),

"BANDWIDTH" VARCHAR2(30 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_XCONNECT\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_XCONNECT\_AUDIT\_WK"

( "TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"MATCH\_CODE" VARCHAR2(20 CHAR),

"MATCH\_STATUS" VARCHAR2(500 CHAR),

"XCONNECT\_ASIDE" VARCHAR2(80 CHAR),

"A\_SIDE\_EQUIP\_ID" NUMBER(9,0),

"A\_SIDE\_PATH\_INST\_ID" NUMBER(9,0),

"A\_SIDE\_LEG\_INST\_ID" NUMBER(9,0),

"A\_SIDE\_MEMBER\_NBR" NUMBER(9,0),

"XCONNECT\_ZSIDE" VARCHAR2(80 CHAR),

"Z\_SIDE\_EQUIP\_ID" NUMBER(9,0),

"Z\_SIDE\_PATH\_INST\_ID" NUMBER(9,0),

"Z\_SIDE\_LEG\_INST\_ID" NUMBER(9,0),

"Z\_SIDE\_MEMBER\_NBR" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 CHAR),

"PATH\_STATUS" VARCHAR2(20 CHAR),

"BANDWIDTH" VARCHAR2(30 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_XCONNECT\_EQUIPMENT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_XCONNECT\_EQUIPMENT"

( "NE\_LEVEL" NUMBER(9,0),

"NE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS\_TYPE" VARCHAR2(20 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"TYPE" VARCHAR2(50 CHAR),

"STATUS" VARCHAR2(50 CHAR),

"VENDOR" VARCHAR2(50 CHAR),

"EQ\_CLASS" VARCHAR2(20 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NETSMART\_XNG\_PORTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NETSMART\_XNG\_PORTS"

( "TID" VARCHAR2(50 CHAR),

"TID\_TYPE" VARCHAR2(50 CHAR),

"SLOT" VARCHAR2(50 CHAR),

"PORT\_HUM\_ID" VARCHAR2(50 CHAR),

"PORT\_ACCESS\_ID" VARCHAR2(100 CHAR),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"LEG\_INST\_ID" NUMBER(9,0),

"MEMBER\_NBR" NUMBER(9,0),

"A\_SIDE\_SITE\_ID" NUMBER(9,0),

"A\_SIDE\_SITE\_HUM\_ID" VARCHAR2(100 CHAR),

"Z\_SIDE\_SITE\_ID" NUMBER(9,0),

"Z\_SIDE\_SITE\_HUM\_ID" VARCHAR2(100 CHAR),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 CHAR),

"BANDWIDTH" VARCHAR2(30 CHAR),

"PATH\_STATUS" VARCHAR2(20 CHAR),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"PREV\_PATH\_INST\_ID" NUMBER(9,0),

"NE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"VENDOR" VARCHAR2(50 CHAR),

"MODEL" VARCHAR2(50 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"PORT\_INST\_ID" NUMBER(9,0),

"CARD\_INST\_ID" NUMBER(9,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NE\_VS\_XNG\_ENB\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_ENB\_AUDIT"

( "NE\_MARKET\_ID" VARCHAR2(3 BYTE),

"NODE" VARCHAR2(100 BYTE),

"FROM\_EMS" VARCHAR2(6 BYTE),

"IP" VARCHAR2(50 BYTE),

"CONNECTION\_STATUS" VARCHAR2(50 BYTE),

"SYNCH\_STATUS" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(500 BYTE),

"XNG\_EQUIP\_NAME" VARCHAR2(100 BYTE),

"XNG\_MARKET\_ID" VARCHAR2(11 BYTE),

"XNG\_ENB\_NUMBER" VARCHAR2(6 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS" CHAR(1 BYTE),

"HAS\_MULTIPLES" VARCHAR2(1 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"NE\_SERVER\_NAME" VARCHAR2(100 BYTE),

"NE\_STATUS" VARCHAR2(50 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"COMMENTS" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NE\_VS\_XNG\_ENB\_AUDIT\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_ENB\_AUDIT\_WRK"

( "NE\_MARKET\_ID" VARCHAR2(3 BYTE),

"NODE" VARCHAR2(100 BYTE),

"FROM\_EMS" VARCHAR2(6 BYTE),

"IP" VARCHAR2(50 BYTE),

"CONNECTION\_STATUS" VARCHAR2(50 BYTE),

"SYNCH\_STATUS" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(500 BYTE),

"XNG\_EQUIP\_NAME" VARCHAR2(100 BYTE),

"XNG\_MARKET\_ID" VARCHAR2(11 BYTE),

"XNG\_ENB\_NUMBER" VARCHAR2(6 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS" CHAR(1 BYTE),

"HAS\_MULTIPLES" VARCHAR2(1 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"NE\_SERVER\_NAME" VARCHAR2(100 BYTE),

"NE\_STATUS" VARCHAR2(50 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"COMMENTS" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NE\_VS\_XNG\_SP\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_AUDIT"

( "DATABASE\_NAME" VARCHAR2(20 BYTE),

"SERVER\_IP" VARCHAR2(20 BYTE),

"GROUP\_NAME" VARCHAR2(100 BYTE),

"SITE\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"ADDRESS" VARCHAR2(4000 BYTE),

"CITY" VARCHAR2(60 BYTE),

"STATE\_PROV" VARCHAR2(40 BYTE),

"POST\_CODE" VARCHAR2(10 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(500 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"DUPLICATE" VARCHAR2(1 CHAR),

"DOMAIN\_INST\_ID" NUMBER(\*,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NE\_VS\_XNG\_SP\_AUDIT\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_AUDIT\_WRK"

( "DATABASE\_NAME" VARCHAR2(20 BYTE),

"SERVER\_IP" VARCHAR2(20 BYTE),

"GROUP\_NAME" VARCHAR2(100 BYTE),

"SITE\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"ADDRESS" VARCHAR2(4000 BYTE),

"CITY" VARCHAR2(60 BYTE),

"STATE\_PROV" VARCHAR2(40 BYTE),

"POST\_CODE" VARCHAR2(10 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(500 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"DUPLICATE" VARCHAR2(1 CHAR),

"DOMAIN\_INST\_ID" NUMBER(\*,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NE\_VS\_XNG\_SP\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY"

( "REGION" VARCHAR2(50 BYTE),

"MATCHED\_GI" NUMBER(6,0),

"NO\_MATCH\_IN\_GI" NUMBER(6,0),

"PER\_MATCHED" NUMBER(5,2) DEFAULT 0,

"TOTAL" NUMBER(6,0) DEFAULT 0,

"AREA" VARCHAR2(25 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NE\_VS\_XNG\_SP\_SUMMARY\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY\_WRK"

( "REGION" VARCHAR2(50 BYTE),

"MATCHED\_GI" NUMBER(6,0),

"NO\_MATCH\_IN\_GI" NUMBER(6,0),

"PER\_MATCHED" NUMBER(5,2) DEFAULT 0,

"TOTAL" NUMBER(6,0) DEFAULT 0,

"AREA" VARCHAR2(25 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGMLS\_DEVICE\_AUDIT\_ISSUES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGMLS\_DEVICE\_AUDIT\_ISSUES"

( "NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_ISSUE\_ID" NUMBER,

"NGMLS\_VENDOR" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK"

( "NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_ISSUE\_ID" NUMBER,

"NGMLS\_VENDOR" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGMLS\_ISSUE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGMLS\_ISSUE"

( "NGMLS\_ISSUE\_ID" NUMBER(\*,0),

"DESCRIPTION" VARCHAR2(100 BYTE),

"IS\_CRITICAL" CHAR(1 BYTE) DEFAULT 'N',

"COMMENTS" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGMLS\_VLAN\_AUDIT\_ISSUES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGMLS\_VLAN\_AUDIT\_ISSUES"

( "NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"NGMLS\_ISSUE\_ID" NUMBER,

"NGMLS\_VENDOR" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGMLS\_VLAN\_AUDIT\_ISSUES\_WK"

( "NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"NGMLS\_ISSUE\_ID" NUMBER,

"NGMLS\_VENDOR" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGT\_QT\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGT\_QT\_NE\_VS\_XNG\_AUDIT"

( "NE\_TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"EQ\_TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"EQUIP\_TYPE" VARCHAR2(5 BYTE),

"MATCH\_CODE" CHAR(10 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"CARD\_INST\_ID" NUMBER,

"CARD\_TEST\_HEAD\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGT\_QT\_NE\_VS\_XNG\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGT\_QT\_NE\_VS\_XNG\_AUDIT\_WK"

( "NE\_TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"EQ\_TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"EQUIP\_TYPE" VARCHAR2(5 BYTE),

"MATCH\_CODE" CHAR(10 BYTE),

"MATCH\_STATUS" VARCHAR2(100 BYTE),

"CARD\_INST\_ID" NUMBER,

"CARD\_TEST\_HEAD\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGT\_QT\_REG\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGT\_QT\_REG\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL" NUMBER,

"GOOD\_MATCH" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NGT\_QT\_REG\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NGT\_QT\_REG\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL" NUMBER,

"GOOD\_MATCH" NUMBER,

"TOTAL\_COMPLIANCE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM"

( "AREA" VARCHAR2(100 CHAR),

"REGION" VARCHAR2(100 CHAR),

"SEG\_EBH\_COUNT" NUMBER(10,0) DEFAULT 0,

"SEG\_NON\_EBH\_COUNT" NUMBER(10,0) DEFAULT 0,

"SEG\_ALL\_COUNT" NUMBER(10,0) DEFAULT 0,

"EQUIP\_EBH\_COUNT" NUMBER(10,0) DEFAULT 0,

"EQUIP\_NON\_EBH\_COUNT" NUMBER(10,0) DEFAULT 0,

"EQUIP\_ALL\_COUNT" NUMBER(10,0) DEFAULT 0,

"PATH\_EBH\_COUNT" NUMBER(10,0) DEFAULT 0,

"PATH\_NON\_EBH\_COUNT" NUMBER(10,0) DEFAULT 0,

"PATH\_ALL\_COUNT" NUMBER(10,0) DEFAULT 0,

"CABLE\_EBH\_COUNT" NUMBER(10,0) DEFAULT 0,

"CABLE\_NON\_EBH\_COUNT" NUMBER(10,0) DEFAULT 0,

"CABLE\_ALL\_COUNT" NUMBER(10,0) DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK"

( "AREA" VARCHAR2(100 CHAR),

"REGION" VARCHAR2(100 CHAR),

"SEG\_EBH\_COUNT" NUMBER(10,0),

"SEG\_NON\_EBH\_COUNT" NUMBER(10,0),

"SEG\_ALL\_COUNT" NUMBER(10,0),

"EQUIP\_EBH\_COUNT" NUMBER(10,0),

"EQUIP\_NON\_EBH\_COUNT" NUMBER(10,0),

"EQUIP\_ALL\_COUNT" NUMBER(10,0),

"PATH\_EBH\_COUNT" NUMBER(10,0),

"PATH\_NON\_EBH\_COUNT" NUMBER(10,0),

"PATH\_ALL\_COUNT" NUMBER(10,0),

"CABLE\_EBH\_COUNT" NUMBER(10,0),

"CABLE\_NON\_EBH\_COUNT" NUMBER(10,0),

"CABLE\_ALL\_COUNT" NUMBER(10,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NL\_OBJ\_IN\_LIVE\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS"

( "CIRC\_PATH\_INST\_ID" NUMBER(10,0),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"SEQUENCE" NUMBER(6,0),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"ELEM\_INST\_ID" NUMBER(10,0),

"PATH\_TYPE" VARCHAR2(30 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"ELEMENT\_NAME" VARCHAR2(100 BYTE),

"DOMAIN\_INST\_ID" NUMBER(10,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NL\_OBJ\_IN\_LIVE\_PATHS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS\_WK"

( "CIRC\_PATH\_INST\_ID" NUMBER(10,0),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"SEQUENCE" NUMBER(6,0),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"ELEM\_INST\_ID" NUMBER(10,0),

"PATH\_TYPE" VARCHAR2(30 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"ELEMENT\_NAME" VARCHAR2(100 BYTE),

"DOMAIN\_INST\_ID" NUMBER(10,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NOCC\_LAST\_TEST\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_DETAILS"

( "DOMAIN\_ID" NUMBER(9,0),

"DOMAIN\_NAME" VARCHAR2(50 BYTE),

"SITE\_INST\_ID" NUMBER(9,0),

"SITE\_NAME" VARCHAR2(100 BYTE),

"SITE\_ID" VARCHAR2(30 BYTE),

"SITE\_TECH\_NAME" VARCHAR2(3100 BYTE),

"INVENTORIED\_CELLS" VARCHAR2(3100 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"NOCC\_MONITORED" VARCHAR2(30 BYTE),

"VZW\_OWNED" VARCHAR2(30 BYTE),

"LIGHTS\_REQD" VARCHAR2(30 BYTE),

"LIGHT\_LAST\_TESTED" VARCHAR2(30 BYTE),

"LIGHT\_NEXT\_TEST" VARCHAR2(30 BYTE),

"LIGHT\_TEST\_OVERDUE" VARCHAR2(30 BYTE),

"ENV\_LAST\_TESTED" VARCHAR2(30 BYTE),

"ENV\_NEXT\_TEST" VARCHAR2(30 BYTE),

"ENV\_TEST\_OVERDUE" VARCHAR2(30 BYTE),

"RUN\_DATE" DATE,

"EQUIP\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NOCC\_LAST\_TEST\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_SUMMARY"

( "AREA" VARCHAR2(40 BYTE),

"REGION" VARCHAR2(80 BYTE),

"DOMAIN\_ID" VARCHAR2(80 BYTE),

"DOMAIN\_NAME" VARCHAR2(100 BYTE),

"SITES" NUMBER(9,0),

"ANTENNAS" NUMBER(9,0),

"VZW\_OWNED" NUMBER(9,0),

"LIGHTS\_REQD" NUMBER(9,0),

"LIGHTS\_TESTED" NUMBER(9,0),

"LIGHTS\_TEST\_OVERDUE" NUMBER(9,0),

"LIGHTS\_TEST\_OK" NUMBER(9,0),

"NOCC\_MONITORED\_SITES" NUMBER(9,0),

"ENV\_TESTED" NUMBER(9,0),

"ENV\_TEST\_OVERDUE" NUMBER(9,0),

"ENV\_TEST\_OK" NUMBER(9,0),

"RUN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NONPRINTABLE\_COLS\_TO\_CLEAN

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NONPRINTABLE\_COLS\_TO\_CLEAN"

( "TABLE\_NAME" VARCHAR2(4000 BYTE),

"COLUMN\_NAME" VARCHAR2(4000 BYTE),

"CLEAN\_TYPE" VARCHAR2(10 BYTE),

"PRIMARY\_KEY" VARCHAR2(4000 BYTE),

"TYPE\_KEY" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NONPRINTABLE\_ERRORS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NONPRINTABLE\_ERRORS"

( "TABLE\_NAME" VARCHAR2(500 BYTE),

"COLUMN\_NAME" VARCHAR2(500 BYTE),

"PRIMARY\_KEY" VARCHAR2(500 BYTE),

"INST\_ID" NUMBER,

"VAL\_ATTR\_INST\_ID" NUMBER,

"ERROR\_MSG" VARCHAR2(1000 BYTE),

"UPD\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NONPRINTABLE\_UPDATES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NONPRINTABLE\_UPDATES"

( "TABLE\_NAME" VARCHAR2(4000 BYTE),

"COLUMN\_NAME" VARCHAR2(4000 BYTE),

"PRIMARY\_KEY" VARCHAR2(4000 BYTE),

"INST\_ID" NUMBER,

"TYPE\_VAL" VARCHAR2(4000 BYTE),

"OLD\_VAL" VARCHAR2(4000 BYTE),

"NEW\_VAL" VARCHAR2(4000 BYTE),

"VAL\_ATTR\_INST\_ID" NUMBER,

"UPD\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_CDMA\_AUDIT\_DEV\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_CDMA\_AUDIT\_DEV\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"LEAF\_DOMAIN" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_MTX" VARCHAR2(100 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER DEFAULT 0.00,

"EBH\_COMP" NUMBER DEFAULT 0.00,

"OVERALL\_COMP" NUMBER DEFAULT 0.00,

"EBH\_CSR\_MATCHED\_LIVE" NUMBER,

"EBH\_CSR\_MATCHED\_NON\_LIVE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_CDMA\_AUDIT\_DEV\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_CDMA\_AUDIT\_DEV\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"LEAF\_DOMAIN" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_MTX" VARCHAR2(100 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"EBH\_CSR\_MATCHED\_LIVE" NUMBER,

"EBH\_CSR\_MATCHED\_NON\_LIVE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_CDMA\_AUDIT\_REG\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_CDMA\_AUDIT\_REG\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"T1\_REPORTED" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_CSR\_MATCHED\_LIVE" NUMBER,

"EBH\_CSR\_MATCHED\_NON\_LIVE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_CDMA\_AUDIT\_REG\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_CDMA\_AUDIT\_REG\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"T1\_DISCOVERED" NUMBER,

"T1\_MATCHED\_LIVE" NUMBER,

"T1\_MATCHED\_NONLIVE" NUMBER,

"EBH\_DISCOVERED" NUMBER,

"EBH\_MATCHED\_LIVE" NUMBER,

"EBH\_MATCHED\_NONLIVE" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"T1\_REPORTED" NUMBER,

"EBH\_REPORTED" NUMBER,

"EBH\_CSR\_MATCHED\_LIVE" NUMBER,

"EBH\_CSR\_MATCHED\_NON\_LIVE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_DOM\_INVENTORY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_DOM\_INVENTORY"

( "SEQ\_NUM" NUMBER,

"EMS\_NAME" VARCHAR2(100 BYTE),

"DOM\_IP" VARCHAR2(100 BYTE),

"BSM\_RAW\_NAME" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONG\_NAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"SLOT\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE,

"SANE\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_DOM\_INVENTORY\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_DOM\_INVENTORY\_WRK"

( "SEQ\_NUM" NUMBER,

"EMS\_NAME" VARCHAR2(100 BYTE),

"DOM\_IP" VARCHAR2(100 BYTE),

"BSM\_RAW\_NAME" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"DCG\_LONG\_NAME" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"SLOT\_NUMBER" NUMBER,

"EXTRACT\_DATE" DATE,

"SANE\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT"

( "VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_EMS" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_MTX" VARCHAR2(100 BYTE),

"NE\_DCG\_NUMBER" NUMBER,

"NE\_SLOT\_NUMBER" NUMBER,

"NE\_SPAN\_NUMBER" NUMBER,

"NE\_DOM\_IP" VARCHAR2(100 BYTE),

"NE\_STATUS" VARCHAR2(50 BYTE),

"TERMINATION\_TYPE" VARCHAR2(8 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_DCG\_NUMBER" NUMBER,

"XNG\_SLOT\_NUMBER" NUMBER,

"XNG\_SPAN\_NUMBER" NUMBER,

"XNG\_DOM\_IP" VARCHAR2(100 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"CIRC\_TYPE" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(1000 BYTE),

"NE\_SPAN\_TYPE" CHAR(10 BYTE),

"EXTRACT\_DATE" DATE,

"AUDIT\_DATE" DATE,

"XNG\_BANDWIDTH" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK"

( "VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_EMS" VARCHAR2(100 BYTE),

"VSM\_DEVICE\_NAME\_MTX" VARCHAR2(100 BYTE),

"NE\_DCG\_NUMBER" NUMBER,

"NE\_SLOT\_NUMBER" NUMBER,

"NE\_SPAN\_NUMBER" NUMBER,

"NE\_DOM\_IP" VARCHAR2(100 BYTE),

"NE\_STATUS" VARCHAR2(50 BYTE),

"TERMINATION\_TYPE" VARCHAR2(8 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_DCG\_NUMBER" NUMBER,

"XNG\_SLOT\_NUMBER" NUMBER,

"XNG\_SPAN\_NUMBER" NUMBER,

"XNG\_DOM\_IP" VARCHAR2(100 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"CIRC\_TYPE" VARCHAR2(100 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(1000 BYTE),

"NE\_SPAN\_TYPE" CHAR(10 BYTE),

"EXTRACT\_DATE" DATE,

"AUDIT\_DATE" DATE,

"XNG\_BANDWIDTH" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_PORT\_INVENTORY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_PORT\_INVENTORY"

( "DOM\_SEQ\_NUM" NUMBER,

"PORT\_TYPE" VARCHAR2(100 BYTE),

"PORT\_NUM" NUMBER,

"STATUS" VARCHAR2(50 BYTE),

"PRIMARY\_IP" VARCHAR2(100 BYTE),

"PRIMARY\_MASK" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NORTEL\_PORT\_INVENTORY\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NORTEL\_PORT\_INVENTORY\_WRK"

( "DOM\_SEQ\_NUM" NUMBER,

"PORT\_TYPE" VARCHAR2(100 BYTE),

"PORT\_NUM" NUMBER,

"STATUS" VARCHAR2(50 BYTE),

"PRIMARY\_IP" VARCHAR2(100 BYTE),

"PRIMARY\_MASK" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NSP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NSP"

( "SITE" VARCHAR2(100 BYTE),

"ASR" VARCHAR2(100 BYTE),

"NWF\_SITE" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NSP\_JUNK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NSP\_JUNK"

( "NWF\_SITE\_ID" VARCHAR2(3100 BYTE),

"SITE\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"ROOM" VARCHAR2(40 BYTE),

"FLOOR" VARCHAR2(40 BYTE),

"ADDRESS" VARCHAR2(4000 BYTE),

"CITY" VARCHAR2(60 BYTE),

"STATE\_PROV" VARCHAR2(40 BYTE),

"POST\_CODE\_1" VARCHAR2(10 BYTE),

"COUNTY" VARCHAR2(40 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NSP\_LIGHT\_REQD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NSP\_LIGHT\_REQD"

( "ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"INST\_ID" NUMBER(9,0),

"VERSION\_CHGD" NUMBER(6,0),

"EDIT\_OPERATION" CHAR(1 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 BYTE),

"CHG\_DESC" VARCHAR2(200 BYTE),

"FIELD\_CHGD" VARCHAR2(550 BYTE),

"OLD\_VALUE" VARCHAR2(4000 BYTE),

"NEW\_VALUE" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NT\_EVDO\_BTS\_TERM\_EMS\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NT\_EVDO\_BTS\_TERM\_EMS\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"VSM\_DEVICE\_NAME\_EMS" VARCHAR2(50 BYTE),

"TOT\_UP\_T1\_SPANS" NUMBER,

"TOT\_UP\_T1\_SPANS\_W\_L\_PATHS" NUMBER,

"TOT\_UP\_T1\_SPANS\_W\_NL\_PATHS" NUMBER,

"TOT\_UP\_EBH\_SPANS" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_L\_PATHS" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_NL\_PATHS" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_L\_PATHSCSR" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_NL\_PATHSCSR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"VSM\_DEVICE\_NAME\_EMS" VARCHAR2(50 BYTE),

"TOT\_UP\_T1\_SPANS" NUMBER,

"TOT\_UP\_T1\_SPANS\_W\_L\_PATHS" NUMBER,

"TOT\_UP\_T1\_SPANS\_W\_NL\_PATHS" NUMBER,

"TOT\_UP\_EBH\_SPANS" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_L\_PATHS" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_NL\_PATHS" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_L\_PATHSCSR" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_NL\_PATHSCSR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NT\_EVDO\_BTS\_TERM\_REG\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NT\_EVDO\_BTS\_TERM\_REG\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_UP\_T1\_SPANS" NUMBER,

"TOT\_UP\_T1\_SPANS\_W\_L\_PATHS" NUMBER,

"TOT\_UP\_T1\_SPANS\_W\_NL\_PATHS" NUMBER,

"TOT\_UP\_EBH\_SPANS" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_L\_PATHS" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_NL\_PATHS" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"T1\_REPORTED" NUMBER,

"EBH\_REPORTED" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_L\_PATHSCSR" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_NL\_PATHSCSR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOT\_UP\_T1\_SPANS" NUMBER,

"TOT\_UP\_T1\_SPANS\_W\_L\_PATHS" NUMBER,

"TOT\_UP\_T1\_SPANS\_W\_NL\_PATHS" NUMBER,

"TOT\_UP\_EBH\_SPANS" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_L\_PATHS" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_NL\_PATHS" NUMBER,

"T1\_COMP" NUMBER,

"EBH\_COMP" NUMBER,

"OVERALL\_COMP" NUMBER,

"T1\_REPORTED" NUMBER,

"EBH\_REPORTED" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_L\_PATHSCSR" NUMBER,

"TOT\_UP\_EBH\_SPANS\_W\_NL\_PATHSCSR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NWF\_BAD\_ENBS\_DEL\_SATYA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA"

( "ENBS" VARCHAR2(100 BYTE),

"ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"INST\_ID" NUMBER(10,0),

"VERSION\_CHGD" NUMBER(6,0),

"EDIT\_OPERATION" CHAR(1 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 BYTE),

"CHG\_DESC" VARCHAR2(200 BYTE),

"DELETED\_NAME" VARCHAR2(50 BYTE),

"EQUIP\_INST\_ID" NUMBER(10,0),

"TYPE" VARCHAR2(30 BYTE),

"VENDOR" VARCHAR2(40 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"REV" VARCHAR2(30 BYTE),

"SW\_REV" VARCHAR2(30 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"PARENT\_EQ\_INST\_ID" NUMBER(10,0),

"STATUS" VARCHAR2(20 BYTE),

"EQ\_CLASS" CHAR(1 BYTE),

"DIM\_HEIGHT" NUMBER(10,3),

"DIM\_WIDTH" NUMBER(10,3),

"DIM\_DEPTH" NUMBER(10,3),

"DIM\_DIST\_TO\_BASE" NUMBER(10,3),

"DIM\_DIST\_TO\_LEFT" NUMBER(10,3),

"DIM\_DIST\_TO\_FRONT" NUMBER(10,3),

"LINEUP" VARCHAR2(12 BYTE),

"FRAME" VARCHAR2(12 BYTE),

"SHELF" VARCHAR2(12 BYTE),

"TARGET\_ID" VARCHAR2(40 BYTE),

"NMS\_EMS" VARCHAR2(30 BYTE),

"ORDER\_NUM" VARCHAR2(50 BYTE),

"ORDERED" DATE,

"DUE" DATE,

"INSTALLED" DATE,

"IN\_SERVICE" DATE,

"SCHED\_DATE" DATE,

"DECOMMISSION" DATE,

"SERIAL\_NO" VARCHAR2(30 BYTE),

"BATCH\_NO" VARCHAR2(20 BYTE),

"BAR\_CODE" VARCHAR2(30 BYTE),

"PURCHASE\_PRICE" NUMBER(10,2),

"PURCHASE\_DATE" DATE,

"ASSET\_LIFE" NUMBER(12,2),

"TG\_MEM\_ACTIVE" CHAR(1 BYTE),

"LAST\_TG\_MEM\_UPD" DATE,

"LAST\_TG\_MEM\_REVIEWED" DATE,

"COMMENTS" VARCHAR2(4000 BYTE),

"LAST\_MOD\_TS" DATE,

"LAST\_MOD\_BY" VARCHAR2(50 BYTE),

"INST\_VER" NUMBER(6,0),

"LOCK\_TS" DATE,

"LOCK\_BY" VARCHAR2(50 BYTE),

"AUTOMATION" VARCHAR2(25 BYTE),

"PLANNING\_ENABLED" CHAR(1 BYTE),

"WARN\_USER\_MOD" CHAR(1 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"CLEI" VARCHAR2(50 BYTE),

"POINT\_CODE" VARCHAR2(12 BYTE),

"EQ\_TPLT\_INST\_ID" NUMBER(10,0),

"CUST\_INST\_ID" NUMBER(10,0),

"NPA" VARCHAR2(50 BYTE),

"ADM\_CO\_INST\_ID" NUMBER(10,0),

"OFFSET\_X" NUMBER(10,3),

"OFFSET\_Y" NUMBER(10,3),

"OFFSET\_Z" NUMBER(10,3),

"ROTATION" NUMBER(10,3),

"AID" VARCHAR2(1000 BYTE),

"HECIG" VARCHAR2(50 BYTE),

"UNIT\_OF\_MEASURE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table NWF\_BAD\_END

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."NWF\_BAD\_END"

( "ENBS" VARCHAR2(100 BYTE),

"ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"INST\_ID" NUMBER(10,0),

"VERSION\_CHGD" NUMBER(6,0),

"EDIT\_OPERATION" CHAR(1 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 BYTE),

"CHG\_DESC" VARCHAR2(200 BYTE),

"DELETED\_NAME" VARCHAR2(50 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER(10,0),

"TYPE" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(10,0),

"LAST\_MOD\_BY" VARCHAR2(50 BYTE),

"LAST\_MOD\_TS" DATE,

"MODEL" VARCHAR2(30 BYTE),

"VENDOR" VARCHAR2(40 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table OBS\_PROD\_DATA\_JUN05\_JC

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."OBS\_PROD\_DATA\_JUN05\_JC"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LIVE" NUMBER,

"RPTD" NUMBER,

"ENB\_TOTAL\_COMPLIANCE" NUMBER,

"MSPP\_TOTAL\_COMPLIANCE" NUMBER,

"E2BTS\_TOTAL\_COMPLIANCE" NUMBER,

"CSR\_TOTAL\_COMPLIANCE" NUMBER,

"T1\_TOTAL\_COMPLIANCE" NUMBER,

"HPOV\_TOTAL\_COMPLIANCE" NUMBER,

"OVERALL\_COMPLIANCE" NUMBER,

"REGIONAL\_MONTHLY\_DELTA" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table OLD\_STATS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."OLD\_STATS"

( "STATID" VARCHAR2(30 BYTE),

"TYPE" CHAR(1 BYTE),

"VERSION" NUMBER,

"FLAGS" NUMBER,

"C1" VARCHAR2(30 BYTE),

"C2" VARCHAR2(30 BYTE),

"C3" VARCHAR2(30 BYTE),

"C4" VARCHAR2(30 BYTE),

"C5" VARCHAR2(30 BYTE),

"N1" NUMBER,

"N2" NUMBER,

"N3" NUMBER,

"N4" NUMBER,

"N5" NUMBER,

"N6" NUMBER,

"N7" NUMBER,

"N8" NUMBER,

"N9" NUMBER,

"N10" NUMBER,

"N11" NUMBER,

"N12" NUMBER,

"D1" DATE,

"R1" RAW(32),

"R2" RAW(32),

"CH1" VARCHAR2(1000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table OPSTRACKER\_ALL\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."OPSTRACKER\_ALL\_DATA"

( "ID" VARCHAR2(30 BYTE),

"SITE\_ID" VARCHAR2(30 BYTE),

"SITE\_NAME" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(30 BYTE),

"MARKET" VARCHAR2(30 BYTE),

"SWITCH" VARCHAR2(30 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"SITE\_FUNCTION" VARCHAR2(30 BYTE),

"BRAND" VARCHAR2(30 BYTE),

"TECH\_NAME" VARCHAR2(30 BYTE),

"ADDRESS" VARCHAR2(30 BYTE),

"CITY" VARCHAR2(30 BYTE),

"ST" VARCHAR2(30 BYTE),

"ZIP" VARCHAR2(30 BYTE),

"COUNTY" VARCHAR2(30 BYTE),

"CLLI" VARCHAR2(30 BYTE),

"MANAGER\_ID" VARCHAR2(30 BYTE),

"NETWORK\_ID" VARCHAR2(30 BYTE),

"XING\_ID" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table OPSTRACKER\_DATA\_SATYA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."OPSTRACKER\_DATA\_SATYA"

( "SITE\_ID" VARCHAR2(100 BYTE),

"SITE\_NAME" VARCHAR2(100 BYTE),

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"MARKET" VARCHAR2(100 BYTE),

"SWITCH" VARCHAR2(100 BYTE),

"SITE\_TYPE" VARCHAR2(100 BYTE),

"ADDRESS" VARCHAR2(200 BYTE),

"CITY" VARCHAR2(100 BYTE),

"ST" VARCHAR2(100 BYTE),

"ZIP" VARCHAR2(100 BYTE),

"COUNTY" VARCHAR2(100 BYTE),

"POWER\_COMPANY" VARCHAR2(100 BYTE),

"POWER\_ACCOUNT" VARCHAR2(100 BYTE),

"POWER\_METER" VARCHAR2(100 BYTE),

"POWER\_PHONE" VARCHAR2(100 BYTE),

"FIRE\_DEPT\_NAME" VARCHAR2(100 BYTE),

"FIRE\_DEPT\_PHONE" VARCHAR2(100 BYTE),

"POLICE\_PHONE" VARCHAR2(100 BYTE),

"SECURITY\_LOCK" VARCHAR2(100 BYTE),

"SECURITY\_LOCK\_NOC\_INT" VARCHAR2(100 BYTE),

"SEC\_LOCK\_STATUS" VARCHAR2(100 BYTE),

"SEC\_LOCK\_STATUS\_DATE" DATE,

"XING\_ID" NUMBER,

"NETWORK\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table OPSTRACKER\_USERS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."OPSTRACKER\_USERS"

( "USERID" VARCHAR2(100 BYTE),

"NAME" VARCHAR2(100 BYTE),

"FNAME" VARCHAR2(100 BYTE),

"LNAME" VARCHAR2(100 BYTE),

"TITLE" VARCHAR2(100 BYTE),

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"MARKET" VARCHAR2(100 BYTE),

"AREA2" VARCHAR2(100 BYTE),

"REGION2" VARCHAR2(100 BYTE),

"MARKET2" VARCHAR2(100 BYTE),

"CONTACT" VARCHAR2(100 BYTE),

"AREA3" VARCHAR2(100 BYTE),

"REGION3" VARCHAR2(100 BYTE),

"MARKET3" VARCHAR2(100 BYTE),

"MANAGER" VARCHAR2(100 BYTE),

"ISTECH" VARCHAR2(100 BYTE),

"ISDIRECTOR" VARCHAR2(100 BYTE),

"ISADMIN" VARCHAR2(100 BYTE),

"ROLL" VARCHAR2(100 BYTE),

"STATUS" VARCHAR2(100 BYTE),

"EMPID" VARCHAR2(100 BYTE),

"EMAIL" VARCHAR2(100 BYTE),

"PERMISSION\_LEVEL" VARCHAR2(100 BYTE),

"VIEWPROFILE" VARCHAR2(100 BYTE),

"PM\_HASBEENCONVERTED" VARCHAR2(100 BYTE),

"MANAGERID" VARCHAR2(100 BYTE),

"META\_UNIVERSALID" VARCHAR2(100 BYTE),

"META\_CREATEDDATE" VARCHAR2(100 BYTE),

"META\_CREATEDBY" VARCHAR2(100 BYTE),

"META\_LASTUPDATEDATE" VARCHAR2(100 BYTE),

"META\_LASTUPDATEBY" VARCHAR2(100 BYTE),

"ISSWITCHTECH" VARCHAR2(100 BYTE),

"READONLYACCESS" VARCHAR2(100 BYTE),

"ISNOC" VARCHAR2(100 BYTE),

"ISSYSPERFENG" VARCHAR2(100 BYTE),

"ISNEPA" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ORDER\_REQD\_FIELDS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ORDER\_REQD\_FIELDS"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"ORDER\_INSTALLED" VARCHAR2(11 BYTE),

"TERM\_END" VARCHAR2(11 BYTE),

"LAST\_RENEWAL\_DATE" VARCHAR2(11 BYTE),

"RENEWAL\_TERM" VARCHAR2(100 BYTE),

"TERM\_START" VARCHAR2(20 BYTE),

"TERM\_PLAN\_EXPR" VARCHAR2(3 BYTE),

"TERM\_REASON" VARCHAR2(200 BYTE),

"RUN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table ORDER\_REQD\_FIELDS\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."ORDER\_REQD\_FIELDS\_WRK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"ORDER\_INSTALLED" VARCHAR2(11 BYTE),

"TERM\_END" VARCHAR2(11 BYTE),

"LAST\_RENEWAL\_DATE" VARCHAR2(11 BYTE),

"RENEWAL\_TERM" VARCHAR2(100 BYTE),

"TERM\_START" VARCHAR2(20 BYTE),

"TERM\_PLAN\_EXPR" VARCHAR2(3 BYTE),

"TERM\_REASON" VARCHAR2(200 BYTE),

"RUN\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table OSS\_RC\_ENB

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."OSS\_RC\_ENB"

( "NODE" VARCHAR2(100 BYTE),

"IP" VARCHAR2(50 BYTE),

"CONNECTION\_STATUS" VARCHAR2(50 BYTE),

"SYNCH\_STATUS" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate,

"SERVER\_NAME" VARCHAR2(100 BYTE),

"S1MME\_IPADDRESS" VARCHAR2(100 BYTE),

"CHASSIS\_TYPE" VARCHAR2(70 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table OSS\_RC\_ENB\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."OSS\_RC\_ENB\_WRK"

( "NODE" VARCHAR2(100 BYTE),

"IP" VARCHAR2(50 BYTE),

"CONNECTION\_STATUS" VARCHAR2(50 BYTE),

"SYNCH\_STATUS" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"SERVER\_NAME" VARCHAR2(100 BYTE),

"S1MME\_IPADDRESS" VARCHAR2(100 BYTE),

"CHASSIS\_TYPE" VARCHAR2(70 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table OT\_CONTACTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."OT\_CONTACTS"

( "SITE\_INST\_ID" NUMBER,

"SITE\_NAME" VARCHAR2(50 BYTE),

"TECH\_NAME" VARCHAR2(50 BYTE),

"MANAGER\_NAME" VARCHAR2(50 BYTE),

"OT\_EXTRACT\_DATE" VARCHAR2(10 BYTE) DEFAULT TO\_CHAR(SYSDATE, 'yyyymmdd')

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PATH\_STANDARD\_REGEX

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PATH\_STANDARD\_REGEX"

( "REGEX\_NAME" VARCHAR2(50 BYTE),

"PATTERN" VARCHAR2(100 BYTE),

"COMMENTS" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PATH\_TYPE\_TO\_SERVICE\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PATH\_TYPE\_TO\_SERVICE\_MAP"

( "PATH\_TYPE" VARCHAR2(20 BYTE),

"SERVICE\_DESC" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PROCESS\_DEPENDENCY\_MATRIX

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PROCESS\_DEPENDENCY\_MATRIX"

( "DS\_PROCESS\_ID" NUMBER,

"PROCESS\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PROD\_STATS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PROD\_STATS"

( "STATID" VARCHAR2(30 BYTE),

"TYPE" CHAR(1 BYTE),

"VERSION" NUMBER,

"FLAGS" NUMBER,

"C1" VARCHAR2(30 BYTE),

"C2" VARCHAR2(30 BYTE),

"C3" VARCHAR2(30 BYTE),

"C4" VARCHAR2(30 BYTE),

"C5" VARCHAR2(30 BYTE),

"N1" NUMBER,

"N2" NUMBER,

"N3" NUMBER,

"N4" NUMBER,

"N5" NUMBER,

"N6" NUMBER,

"N7" NUMBER,

"N8" NUMBER,

"N9" NUMBER,

"N10" NUMBER,

"N11" NUMBER,

"N12" NUMBER,

"D1" DATE,

"R1" RAW(32),

"R2" RAW(32),

"CH1" VARCHAR2(1000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 81920 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SYSAUX" ;

--------------------------------------------------------

-- DDL for Table PROSPECT\_SERVERS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PROSPECT\_SERVERS"

( "SERVER" VARCHAR2(50 BYTE),

"IP\_ADDRESS" VARCHAR2(50 BYTE),

"ENABLED\_YN" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PROSPECT\_SERVERS\_SAVE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PROSPECT\_SERVERS\_SAVE"

( "SERVER" VARCHAR2(50 BYTE),

"IP\_ADDRESS" VARCHAR2(50 BYTE),

"ENABLED\_YN" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PROS\_DS1\_MAP\_TEMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PROS\_DS1\_MAP\_TEMP"

( "SYS\_ID" NUMBER,

"ECP\_SID" NUMBER,

"CELL\_NUMBER" NUMBER,

"URC\_NUM" NUMBER,

"DS1\_NUM" NUMBER,

"BTS\_NAME" VARCHAR2(50 BYTE),

"CELL\_TYPE" VARCHAR2(10 BYTE),

"VERSION" VARCHAR2(50 BYTE),

"CELL\_STAT" VARCHAR2(10 BYTE),

"DS1\_STAT" VARCHAR2(10 BYTE),

"SERVER" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"SERVICE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PROS\_EBH\_MAP\_TEMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PROS\_EBH\_MAP\_TEMP"

( "SYS\_ID" NUMBER,

"ECP\_SID" NUMBER,

"CELL\_NUMBER" NUMBER,

"URC\_NUM" NUMBER,

"BTS\_NAME" VARCHAR2(50 BYTE),

"CELL\_TYPE" VARCHAR2(10 BYTE),

"VERSION" VARCHAR2(50 BYTE),

"CELL\_STAT" VARCHAR2(10 BYTE),

"URC\_STAT" VARCHAR2(10 BYTE),

"SERVER" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"SERVICE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PROS\_ECP\_CELL\_DS1\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PROS\_ECP\_CELL\_DS1\_MAP"

( "SYS\_ID" NUMBER,

"ECP\_SID" NUMBER,

"CELL\_NUMBER" NUMBER,

"URC\_NUM" NUMBER,

"DS1\_NUM" NUMBER,

"BTS\_NAME" VARCHAR2(50 BYTE),

"CELL\_TYPE" VARCHAR2(10 BYTE),

"VERSION" VARCHAR2(50 BYTE),

"CELL\_STAT" VARCHAR2(10 BYTE),

"DS1\_STAT" VARCHAR2(10 BYTE),

"SERVER" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"SERVICE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 16777216 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table PROS\_ECP\_CELL\_EBH\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."PROS\_ECP\_CELL\_EBH\_MAP"

( "SYS\_ID" NUMBER,

"ECP\_SID" NUMBER,

"CELL\_NUMBER" NUMBER,

"URC\_NUM" NUMBER,

"BTS\_NAME" VARCHAR2(50 BYTE),

"CELL\_TYPE" VARCHAR2(10 BYTE),

"VERSION" VARCHAR2(50 BYTE),

"CELL\_STAT" VARCHAR2(10 BYTE),

"URC\_STAT" VARCHAR2(10 BYTE),

"SERVER" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"SERVICE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table RBED\_SEVONE\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DATA"

( "DEVICE\_ID" NUMBER(10,0),

"DEVICE\_NAME" VARCHAR2(100 BYTE),

"OBJECT\_NAME" VARCHAR2(256 BYTE),

"INDICATOR\_NAME" VARCHAR2(50 BYTE),

"VALUE" NUMBER(30,3),

"EPOCH" NUMBER(10,0),

"TIME\_DELTA" NUMBER(10,0),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table RBED\_SEVONE\_DEVICE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DEVICE"

( "ID" NUMBER(10,0),

"NAME" VARCHAR2(100 BYTE),

"SYSNAME" VARCHAR2(100 BYTE),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table RBED\_SEVONE\_OBJECT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."RBED\_SEVONE\_OBJECT"

( "DEVICEID" NUMBER(10,0),

"NAME" VARCHAR2(256 BYTE),

"IFINDEX" VARCHAR2(50 BYTE),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table READ\_ONLY\_OLD\_CK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."READ\_ONLY\_OLD\_CK"

( "EQUIP\_INST\_ID" NUMBER(10,0),

"DESCR" VARCHAR2(100 BYTE),

"INST\_VER" NUMBER(6,0),

"NEW\_VER" NUMBER,

"VAL\_ATTR\_INST\_ID" NUMBER(10,0),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_VALUE" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table READ\_ONLY\_OLD\_UDAS\_CLEANUP\_LOG

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."READ\_ONLY\_OLD\_UDAS\_CLEANUP\_LOG"

( "ELEMENT\_TYPE" CHAR(1 BYTE),

"INST\_ID" NUMBER(9,0),

"INST\_VER" NUMBER(6,0),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_VALUE" VARCHAR2(3100 BYTE),

"DELETED\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table READ\_ONLY\_UDA\_CLEANUP\_LOG

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."READ\_ONLY\_UDA\_CLEANUP\_LOG"

( "ELEMENT\_TYPE" CHAR(1 BYTE),

"INST\_ID" NUMBER(9,0),

"INST\_VER" NUMBER(6,0),

"ATTR\_NAME" VARCHAR2(30 BYTE),

"ATTR\_VALUE" VARCHAR2(3100 BYTE),

"DELETED\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table READ\_ONLY\_UDA\_CLEANUP\_NAMES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."READ\_ONLY\_UDA\_CLEANUP\_NAMES"

( "ELEMENT\_TYPE" CHAR(1 BYTE),

"ATTR\_NAME" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table REMEDY\_EBOND

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."REMEDY\_EBOND"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(100 BYTE),

"LOCATION" VARCHAR2(100 BYTE),

"CIRCUIT\_ID" VARCHAR2(250 BYTE),

"EBOND\_VENDOR" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table REPORT\_CONTROL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."REPORT\_CONTROL"

( "REPORT\_NAME" VARCHAR2(100 BYTE),

"REPORT\_ID" NUMBER(18,0),

"RUN\_DATE" DATE,

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE),

"RUN\_STATUS" VARCHAR2(25 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON COLUMN "XNG\_REPORTS"."REPORT\_CONTROL"."REPORT\_ID" IS 'REPORT\_CONTROL\_1SQ';

--------------------------------------------------------

-- DDL for Table REPORT\_DCOM\_PATH\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL"

( "REPORT\_ID" NUMBER(18,0),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE),

"ROW\_NUM" NUMBER(18,0),

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PATH\_ID" VARCHAR2(200 BYTE),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"NEXT\_PATH\_ID" VARCHAR2(200 BYTE),

"NEXT\_PATH\_STATUS" VARCHAR2(20 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"BPS\_PRESENTED" NUMBER(18,0),

"IN\_SERVICE\_DATE" DATE,

"DECOMMISSION\_DATE" DATE,

"CHANGED\_BY" VARCHAR2(50 BYTE),

"CHANGE\_DATE" DATE,

"ROOT\_STATUS" CHAR(1 BYTE),

"NEXT\_ROOT\_STATUS" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table REPORT\_DCOM\_PATH\_NOTIFICATION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION"

( "USER\_NAME" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE),

"REPORT\_ID" NUMBER(18,0),

"NOTIFICATION\_ID" NUMBER(15,0),

"NOTIFICATION\_DATE" DATE,

"STATUS" VARCHAR2(15 BYTE),

"STATUS\_MESSAGE" CLOB

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00"

LOB ("STATUS\_MESSAGE") STORE AS BASICFILE (

TABLESPACE "SS\_DATA00" ENABLE STORAGE IN ROW CHUNK 16384 RETENTION

NOCACHE LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)) ;

--------------------------------------------------------

-- DDL for Table REPORT\_DCOM\_PATH\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_SUMMARY"

( "REPORT\_ID" NUMBER(18,0),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE),

"ROW\_NUM" NUMBER(18,0),

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"TOTAL\_COUNT" NUMBER(18,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_CSR\_VLAN\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_VENDOR" VARCHAR2(50 BYTE),

"NE\_HOSTNAME" VARCHAR2(100 BYTE),

"DISPLAYED\_NAME" VARCHAR2(50 BYTE),

"NE\_VLAN\_NUMBER" NUMBER,

"NODE\_ID" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"XNG\_VLAN\_NUMBER" NUMBER,

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"CLLI" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"PORT\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_CSR\_VLAN\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT\_WK"

( "CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_VENDOR" VARCHAR2(50 BYTE),

"NE\_HOSTNAME" VARCHAR2(100 BYTE),

"DISPLAYED\_NAME" VARCHAR2(50 BYTE),

"NE\_VLAN\_NUMBER" NUMBER,

"NODE\_ID" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"XNG\_VLAN\_NUMBER" NUMBER,

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"CLLI" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"PORT\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_CSR\_VLAN\_CLLI\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_CLLI\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_CSR\_VLAN\_CLLI\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_CLLI\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_CSR\_VLAN\_DEVICE\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_DEVICE\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_VENDOR" VARCHAR2(2 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_CSR\_VLAN\_DEVICE\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_DEVICE\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"CSR\_VENDOR" VARCHAR2(2 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_CSR\_VLAN\_REGION\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_REGION\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_CSR\_VLAN\_REGION\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_REGION\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NE\_TOTAL\_CSR\_VLAN" NUMBER,

"MATCHED\_L\_CSR\_VLAN" NUMBER,

"MATCHED\_NL\_CSR\_VLAN" NUMBER,

"MISMATCHED\_CSR\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EBH\_HEALTH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EBH\_HEALTH"

( "LAST\_RESYN\_START" DATE,

"LAST\_RESYN\_END" DATE,

"MAC\_ADDRESS" VARCHAR2(50 BYTE),

"REACHABILITY" VARCHAR2(50 BYTE),

"CHASSIS\_TYPE" VARCHAR2(50 BYTE),

"SITE\_NAME" VARCHAR2(50 BYTE),

"RESYNC\_STATE" VARCHAR2(50 BYTE),

"RESYNC\_STATUS" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate,

"SERVER\_NAME" VARCHAR2(50 BYTE),

"INST\_ID" NUMBER(10,0),

"DEVICE\_TYPE" VARCHAR2(30 BYTE),

"PARSE\_STATUS" VARCHAR2(200 BYTE),

"OUT\_OF\_BAND\_ADDR" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EBH\_HEALTH\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EBH\_HEALTH\_WK"

( "LAST\_RESYN\_START" DATE,

"LAST\_RESYN\_END" DATE,

"MAC\_ADDRESS" VARCHAR2(50 BYTE),

"REACHABILITY" VARCHAR2(50 BYTE),

"CHASSIS\_TYPE" VARCHAR2(50 BYTE),

"SITE\_NAME" VARCHAR2(50 BYTE),

"RESYNC\_STATE" VARCHAR2(50 BYTE),

"RESYNC\_STATUS" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate,

"SERVER\_NAME" VARCHAR2(50 BYTE),

"INST\_ID" NUMBER(10,0),

"DEVICE\_TYPE" VARCHAR2(30 BYTE),

"PARSE\_STATUS" VARCHAR2(200 BYTE),

"OUT\_OF\_BAND\_ADDR" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

COMMENT ON COLUMN "XNG\_REPORTS"."SAM\_EBH\_HEALTH\_WK"."PARSE\_STATUS" IS 'If the discovered device does not match the MTCE standard then mark it here.';

--------------------------------------------------------

-- DDL for Table SAM\_EBH\_PORT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EBH\_PORT"

( "CARD\_TYPE\_BIT" VARCHAR2(50 BYTE),

"EBH\_PORT\_MODE" VARCHAR2(50 BYTE),

"MAC\_ADDRESS" VARCHAR2(50 BYTE),

"ENCAP\_TYPE" VARCHAR2(50 BYTE),

"PORT\_NAME" VARCHAR2(50 BYTE),

"SITE\_ID" VARCHAR2(50 BYTE),

"SITE\_NAME" VARCHAR2(50 BYTE),

"OPR\_STATE" VARCHAR2(50 BYTE),

"ADM\_STATE" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE,

"SERVER\_NAME" VARCHAR2(50 BYTE),

"CFM\_LB\_MODE" VARCHAR2(50 BYTE),

"INST\_ID" NUMBER(10,0),

"FUTURE\_USE" VARCHAR2(50 BYTE),

"DEVICE\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EBH\_PORT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EBH\_PORT\_WK"

( "CARD\_TYPE\_BIT" VARCHAR2(50 BYTE),

"EBH\_PORT\_MODE" VARCHAR2(50 BYTE),

"MAC\_ADDRESS" VARCHAR2(50 BYTE),

"ENCAP\_TYPE" VARCHAR2(50 BYTE),

"PORT\_NAME" VARCHAR2(50 BYTE),

"SITE\_ID" VARCHAR2(50 BYTE),

"SITE\_NAME" VARCHAR2(50 BYTE),

"OPR\_STATE" VARCHAR2(50 BYTE),

"ADM\_STATE" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate,

"SERVER\_NAME" VARCHAR2(50 BYTE),

"CFM\_LB\_MODE" VARCHAR2(50 BYTE),

"INST\_ID" NUMBER(10,0),

"FUTURE\_USE" VARCHAR2(50 BYTE),

"DEVICE\_TYPE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EBH\_RTR

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EBH\_RTR"

( "PHYSICAL\_ADDRESS" VARCHAR2(50 BYTE),

"IP" VARCHAR2(50 BYTE),

"OUT\_ENCAP\_VAL" NUMBER,

"PORT\_NAME" VARCHAR2(100 BYTE),

"ENCAP\_TYPE" VARCHAR2(50 BYTE),

"NODE\_NAME" VARCHAR2(50 BYTE),

"NODE\_ID" VARCHAR2(50 BYTE),

"DISPLAYED\_NAME" VARCHAR2(50 BYTE),

"DESCRIPTION" VARCHAR2(200 BYTE),

"INST\_ID" NUMBER(10,0),

"EXTRACT\_DATE" DATE,

"SERVER\_NAME" VARCHAR2(50 BYTE),

"DEVICE\_TYPE" VARCHAR2(30 BYTE),

"PARSE\_STATUS" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EBH\_RTR\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EBH\_RTR\_WK"

( "PHYSICAL\_ADDRESS" VARCHAR2(50 BYTE),

"IP" VARCHAR2(50 BYTE),

"OUT\_ENCAP\_VAL" NUMBER,

"PORT\_NAME" VARCHAR2(100 BYTE),

"ENCAP\_TYPE" VARCHAR2(50 BYTE),

"NODE\_NAME" VARCHAR2(50 BYTE),

"NODE\_ID" VARCHAR2(50 BYTE),

"DISPLAYED\_NAME" VARCHAR2(50 BYTE),

"DESCRIPTION" VARCHAR2(200 BYTE),

"INST\_ID" NUMBER(10,0),

"EXTRACT\_DATE" DATE,

"SERVER\_NAME" VARCHAR2(50 BYTE),

"DEVICE\_TYPE" VARCHAR2(30 BYTE),

"PARSE\_STATUS" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EH\_GI\_CSR\_CLLI\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EH\_GI\_CSR\_CLLI\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CLLI\_6" VARCHAR2(50 BYTE),

"EH\_CSRS" NUMBER,

"GI\_L\_CSRS" NUMBER,

"SAM\_CSRS" NUMBER,

"SAM\_GI\_L\_MATCH" NUMBER,

"SAM\_GI\_PER" NUMBER,

"EH\_GI\_L\_MATCH" NUMBER,

"EH\_GI\_PER" NUMBER,

"SAM\_EH\_GI\_ISSUES" NUMBER,

"SAM\_GI\_NL\_MATCH" NUMBER,

"SAM\_GI\_MATCH" NUMBER,

"EH\_GI\_NL\_MATCH" NUMBER,

"EH\_GI\_CSRS" NUMBER,

"EH\_GI\_MATCH" NUMBER,

"GI\_NL\_CSRS" NUMBER,

"GI\_CSRS" NUMBER,

"SAM\_GI\_MIS\_MATCH" NUMBER,

"EH\_GI\_MIS\_MATCH" NUMBER,

"BRIX\_CSRS" NUMBER DEFAULT 0,

"BRIX\_GI\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_L\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_NL\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_MIS\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_PER" NUMBER DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EH\_GI\_CSR\_CLLI\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EH\_GI\_CSR\_CLLI\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CLLI\_6" VARCHAR2(50 BYTE),

"EH\_CSRS" NUMBER,

"GI\_L\_CSRS" NUMBER,

"SAM\_CSRS" NUMBER,

"SAM\_GI\_L\_MATCH" NUMBER,

"SAM\_GI\_PER" NUMBER,

"EH\_GI\_L\_MATCH" NUMBER,

"EH\_GI\_PER" NUMBER,

"SAM\_EH\_GI\_ISSUES" NUMBER,

"SAM\_GI\_NL\_MATCH" NUMBER,

"SAM\_GI\_MATCH" NUMBER,

"EH\_GI\_NL\_MATCH" NUMBER,

"EH\_GI\_CSRS" NUMBER,

"EH\_GI\_MATCH" NUMBER,

"GI\_NL\_CSRS" NUMBER,

"GI\_CSRS" NUMBER,

"SAM\_GI\_MIS\_MATCH" NUMBER,

"EH\_GI\_MIS\_MATCH" NUMBER,

"BRIX\_CSRS" NUMBER DEFAULT 0,

"BRIX\_GI\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_L\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_NL\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_MIS\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_PER" NUMBER DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EH\_GI\_CSR\_REG\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EH\_GI\_CSR\_REG\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"GI\_CSRS" NUMBER,

"GI\_L\_CSRS" NUMBER,

"GI\_NL\_CSRS" NUMBER,

"EH\_CSRS" NUMBER,

"EH\_GI\_MATCH" NUMBER,

"EH\_GI\_L\_MATCH" NUMBER,

"EH\_GI\_NL\_MATCH" NUMBER,

"EH\_GI\_PER" NUMBER,

"SAM\_CSRS" NUMBER,

"SAM\_GI\_MATCH" NUMBER,

"SAM\_GI\_L\_MATCH" NUMBER,

"SAM\_GI\_NL\_MATCH" NUMBER,

"SAM\_GI\_PER" NUMBER,

"SAM\_EH\_GI\_ISSUES" NUMBER,

"EH\_GI\_MIS\_MATCH" NUMBER,

"SAM\_GI\_MIS\_MATCH" NUMBER,

"BRIX\_CSRS" NUMBER DEFAULT 0,

"BRIX\_GI\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_L\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_NL\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_MIS\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_PER" NUMBER DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_EH\_GI\_CSR\_REG\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_EH\_GI\_CSR\_REG\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"GI\_CSRS" NUMBER,

"GI\_L\_CSRS" NUMBER,

"GI\_NL\_CSRS" NUMBER,

"EH\_CSRS" NUMBER,

"EH\_GI\_MATCH" NUMBER,

"EH\_GI\_L\_MATCH" NUMBER,

"EH\_GI\_NL\_MATCH" NUMBER,

"EH\_GI\_PER" NUMBER,

"SAM\_CSRS" NUMBER,

"SAM\_GI\_MATCH" NUMBER,

"SAM\_GI\_L\_MATCH" NUMBER,

"SAM\_GI\_NL\_MATCH" NUMBER,

"SAM\_GI\_PER" NUMBER,

"SAM\_EH\_GI\_ISSUES" NUMBER,

"EH\_GI\_MIS\_MATCH" NUMBER,

"SAM\_GI\_MIS\_MATCH" NUMBER,

"BRIX\_CSRS" NUMBER DEFAULT 0,

"BRIX\_GI\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_L\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_NL\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_MIS\_MATCH" NUMBER DEFAULT 0,

"BRIX\_GI\_PER" NUMBER DEFAULT 0

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_ENB

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_ENB"

( "NODE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" VARCHAR2(50 BYTE),

"SERVER\_NAME" VARCHAR2(100 BYTE),

"ENB\_STATUS" VARCHAR2(50 BYTE),

"ENB\_LONG" NUMBER,

"ENB\_LAT" NUMBER,

"REMOTE\_IP" VARCHAR2(50 BYTE),

"TELECOM\_IP" VARCHAR2(50 BYTE),

"IP4" VARCHAR2(50 BYTE),

"IP6" VARCHAR2(50 BYTE),

"CHASSIS\_TYPE" VARCHAR2(50 BYTE),

"CONTROLLER\_CARD\_TYPE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_ENB\_010816

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_ENB\_010816"

( "NODE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" VARCHAR2(50 BYTE),

"SERVER\_NAME" VARCHAR2(100 BYTE),

"ENB\_STATUS" VARCHAR2(50 BYTE),

"ENB\_LONG" NUMBER,

"ENB\_LAT" NUMBER,

"REMOTE\_IP" VARCHAR2(50 BYTE),

"TELECOM\_IP" VARCHAR2(50 BYTE),

"IP4" VARCHAR2(50 BYTE),

"IP6" VARCHAR2(50 BYTE),

"CHASSIS\_TYPE" VARCHAR2(50 BYTE),

"CONTROLLER\_CARD\_TYPE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_ENB\_SAVE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_ENB\_SAVE"

( "NODE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" VARCHAR2(50 BYTE),

"SERVER\_NAME" VARCHAR2(100 BYTE),

"ENB\_STATUS" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_ENB\_SERVER\_PAIRS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_ENB\_SERVER\_PAIRS"

( "PAIR\_ID" NUMBER,

"SERVER\_NAME" VARCHAR2(100 BYTE),

"SERVER\_IP" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_ENB\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_ENB\_WK"

( "NODE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" VARCHAR2(50 BYTE),

"SERVER\_NAME" VARCHAR2(100 BYTE),

"ENB\_STATUS" VARCHAR2(50 BYTE),

"ENB\_LONG" NUMBER,

"ENB\_LAT" NUMBER,

"REMOTE\_IP" VARCHAR2(50 BYTE),

"TELECOM\_IP" VARCHAR2(50 BYTE),

"IP4" VARCHAR2(50 BYTE),

"IP6" VARCHAR2(50 BYTE),

"CHASSIS\_TYPE" VARCHAR2(50 BYTE),

"CONTROLLER\_CARD\_TYPE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_ENB\_WK\_010816

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_ENB\_WK\_010816"

( "NODE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" VARCHAR2(50 BYTE),

"SERVER\_NAME" VARCHAR2(100 BYTE),

"ENB\_STATUS" VARCHAR2(50 BYTE),

"ENB\_LONG" NUMBER,

"ENB\_LAT" NUMBER,

"REMOTE\_IP" VARCHAR2(50 BYTE),

"TELECOM\_IP" VARCHAR2(50 BYTE),

"IP4" VARCHAR2(50 BYTE),

"IP6" VARCHAR2(50 BYTE),

"CHASSIS\_TYPE" VARCHAR2(50 BYTE),

"CONTROLLER\_CARD\_TYPE" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_GI\_AL\_CSR\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_GI\_AL\_CSR\_AUDIT"

( "CSR\_VENDOR" VARCHAR2(2 BYTE),

"CSR\_DEVICE\_NAME" VARCHAR2(200 BYTE),

"SAM\_CSR\_DEVICE\_NAME" VARCHAR2(50 BYTE),

"SAM\_REACHABILITY" VARCHAR2(50 BYTE),

"SAM\_RESYNC\_STATUS" VARCHAR2(50 BYTE),

"SAM\_RESYNC\_STATE" VARCHAR2(50 BYTE),

"SAM\_SERVER\_NAME" VARCHAR2(50 BYTE),

"SAM\_LAST\_RESYNC\_START" DATE,

"SAM\_LAST\_RESYNC\_END" DATE,

"GI\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"EQ\_STATUS" VARCHAR2(20 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"EH\_CSR\_HOSTNAME" VARCHAR2(50 BYTE),

"EH\_CSR\_IPADDRESS" VARCHAR2(50 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CLLI" VARCHAR2(50 BYTE),

"BRIX\_CSR\_SITE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_GI\_AL\_CSR\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_GI\_AL\_CSR\_AUDIT\_WK"

( "CSR\_VENDOR" VARCHAR2(2 BYTE),

"CSR\_DEVICE\_NAME" VARCHAR2(200 BYTE),

"SAM\_CSR\_DEVICE\_NAME" VARCHAR2(50 BYTE),

"SAM\_REACHABILITY" VARCHAR2(50 BYTE),

"SAM\_RESYNC\_STATUS" VARCHAR2(50 BYTE),

"SAM\_RESYNC\_STATE" VARCHAR2(50 BYTE),

"SAM\_SERVER\_NAME" VARCHAR2(50 BYTE),

"SAM\_LAST\_RESYNC\_START" DATE,

"SAM\_LAST\_RESYNC\_END" DATE,

"GI\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"EQ\_STATUS" VARCHAR2(20 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"EH\_CSR\_HOSTNAME" VARCHAR2(50 BYTE),

"EH\_CSR\_IPADDRESS" VARCHAR2(50 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"CLLI" VARCHAR2(50 BYTE),

"BRIX\_CSR\_SITE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_GI\_AL\_NGMLS\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_GI\_AL\_NGMLS\_AUDIT"

( "NGMLS\_VENDOR" CHAR(2 BYTE),

"NGMLS\_DEVICE\_NAME" VARCHAR2(200 BYTE),

"SAM\_NGMLS\_DEVICE\_NAME" VARCHAR2(50 BYTE),

"SAM\_REACHABILITY" VARCHAR2(50 BYTE),

"SAM\_RESYNC\_STATUS" VARCHAR2(50 BYTE),

"SAM\_RESYNC\_STATE" VARCHAR2(50 BYTE),

"SAM\_SERVER\_NAME" VARCHAR2(50 BYTE),

"SAM\_LAST\_RESYNC\_START" DATE,

"SAM\_LAST\_RESYNC\_END" DATE,

"GI\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NE\_INST\_ID" NUMBER(9,0),

"LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"EQ\_STATUS" VARCHAR2(20 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_GI\_AL\_NGMLS\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_GI\_AL\_NGMLS\_AUDIT\_WK"

( "NGMLS\_VENDOR" CHAR(2 BYTE),

"NGMLS\_DEVICE\_NAME" VARCHAR2(200 BYTE),

"SAM\_NGMLS\_DEVICE\_NAME" VARCHAR2(50 BYTE),

"SAM\_REACHABILITY" VARCHAR2(50 BYTE),

"SAM\_RESYNC\_STATUS" VARCHAR2(50 BYTE),

"SAM\_RESYNC\_STATE" VARCHAR2(50 BYTE),

"SAM\_SERVER\_NAME" VARCHAR2(50 BYTE),

"SAM\_LAST\_RESYNC\_START" DATE,

"SAM\_LAST\_RESYNC\_END" DATE,

"GI\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NE\_INST\_ID" NUMBER(9,0),

"LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"EQ\_STATUS" VARCHAR2(20 BYTE),

"MATCH\_CODE" VARCHAR2(8 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_GI\_AL\_NGMLS\_REGION\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_GI\_AL\_NGMLS\_REGION\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"SAM\_NGMLSS" NUMBER,

"GI\_NGMLSS" NUMBER,

"GI\_L\_NGMLSS" NUMBER,

"GI\_NL\_NGMLSS" NUMBER,

"SAM\_GI\_MATCH" NUMBER,

"SAM\_GI\_L\_MATCH" NUMBER,

"SAM\_GI\_NL\_MATCH" NUMBER,

"SAM\_GI\_MISMATCH" NUMBER,

"SAM\_GI\_PER" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_GI\_AL\_NGMLS\_REGION\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_GI\_AL\_NGMLS\_REGION\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"SAM\_NGMLSS" NUMBER,

"GI\_NGMLSS" NUMBER,

"GI\_L\_NGMLSS" NUMBER,

"GI\_NL\_NGMLSS" NUMBER,

"SAM\_GI\_MATCH" NUMBER,

"SAM\_GI\_L\_MATCH" NUMBER,

"SAM\_GI\_NL\_MATCH" NUMBER,

"SAM\_GI\_MISMATCH" NUMBER,

"SAM\_GI\_PER" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_NGMLS\_VLAN\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT"

( "NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_VENDOR" VARCHAR2(50 BYTE),

"NE\_HOSTNAME" VARCHAR2(100 BYTE),

"DISPLAYED\_NAME" VARCHAR2(50 BYTE),

"NE\_VLAN\_NUMBER" NUMBER,

"NODE\_ID" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"NE\_INST\_ID" NUMBER,

"XNG\_VLAN\_NUMBER" NUMBER,

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"PORT\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_NGMLS\_VLAN\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT\_WK"

( "NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_VENDOR" VARCHAR2(50 BYTE),

"NE\_HOSTNAME" VARCHAR2(100 BYTE),

"DISPLAYED\_NAME" VARCHAR2(50 BYTE),

"NE\_VLAN\_NUMBER" NUMBER,

"NODE\_ID" VARCHAR2(50 BYTE),

"MATCH\_CODE" VARCHAR2(10 BYTE),

"MATCH\_STATUS" VARCHAR2(250 BYTE),

"NE\_INST\_ID" NUMBER,

"XNG\_VLAN\_NUMBER" NUMBER,

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"AUDIT\_DATE" DATE DEFAULT sysdate,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"PORT\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_NGMLS\_VLAN\_DEVICE\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_DEVICE\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_VENDOR" VARCHAR2(2 BYTE),

"NE\_TOTAL\_NGMLS\_VLAN" NUMBER,

"MATCHED\_L\_NGMLS\_VLAN" NUMBER,

"MATCHED\_NL\_NGMLS\_VLAN" NUMBER,

"MISMATCHED\_NGMLS\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER,

"NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"NGMLS\_VENDOR" VARCHAR2(2 BYTE),

"NE\_TOTAL\_NGMLS\_VLAN" NUMBER,

"MATCHED\_L\_NGMLS\_VLAN" NUMBER,

"MATCHED\_NL\_NGMLS\_VLAN" NUMBER,

"MISMATCHED\_NGMLS\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_NGMLS\_VLAN\_REGION\_SUMM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_REGION\_SUMM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NE\_TOTAL\_NGMLS\_VLAN" NUMBER,

"MATCHED\_L\_NGMLS\_VLAN" NUMBER,

"MATCHED\_NL\_NGMLS\_VLAN" NUMBER,

"MISMATCHED\_NGMLS\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SAM\_NGMLS\_VLAN\_REGION\_SUMM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_REGION\_SUMM\_WK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NE\_TOTAL\_NGMLS\_VLAN" NUMBER,

"MATCHED\_L\_NGMLS\_VLAN" NUMBER,

"MATCHED\_NL\_NGMLS\_VLAN" NUMBER,

"MISMATCHED\_NGMLS\_VLAN" NUMBER,

"VLAN\_COMPLIANCE" NUMBER,

"GI\_NL\_2\_TOTAL\_VLAN\_PCT" NUMBER,

"AUDIT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEGMENTS\_MISSING\_EST\_COST

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST"

( "CIRC\_INST\_ID" NUMBER(\*,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"INSERTED\_ON" DATE,

"IS\_COST\_MISSING" NUMBER(\*,0) DEFAULT 0,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEGMENTS\_MODIFIED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEGMENTS\_MODIFIED"

( "CIRC\_INST\_ID" NUMBER(\*,0),

"NEW\_STATUS" VARCHAR2(20 BYTE),

"CHG\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEGMENTS\_MODIFIED\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEGMENTS\_MODIFIED\_WK"

( "CIRC\_INST\_ID" NUMBER(\*,0),

"NEW\_STATUS" VARCHAR2(20 BYTE),

"CHG\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEGMENTS\_TERM\_EXPIRATION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION"

( "CIRC\_INST\_ID" NUMBER(\*,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"IS\_TERM\_EXPIRED" NUMBER(\*,0) DEFAULT 0,

"TERM\_END" DATE,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN" VARCHAR2(50 BYTE),

"RECUR\_PERIOD" CHAR(1 BYTE),

"ORDERED\_DATE" DATE,

"RECUR\_COSTS" NUMBER(10,2),

"TERM\_REASON" VARCHAR2(200 BYTE),

"TERM\_START" VARCHAR2(20 BYTE),

"TERM\_PLAN\_EXP" VARCHAR2(20 BYTE),

"LAST\_BILL\_DATE" DATE,

"TERM\_DURATION" NUMBER(13,3),

"ORDER\_NUM" VARCHAR2(50 BYTE),

"M2M\_YN" CHAR(1 BYTE),

"DATE\_BEGIN" DATE,

"RUN\_DATE" DATE,

"A\_SITE\_ID" NUMBER(10,0),

"A\_SITE\_NAME" VARCHAR2(100 BYTE),

"Z\_SITE\_ID" NUMBER(10,0),

"Z\_SITE\_NAME" VARCHAR2(100 BYTE),

"VENDOR" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEGMENTS\_TERM\_EXPIRATION\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION\_WK"

( "CIRC\_INST\_ID" NUMBER(\*,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"IS\_TERM\_EXPIRED" NUMBER(\*,0) DEFAULT 0,

"TERM\_END" DATE,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN" VARCHAR2(50 BYTE),

"RECUR\_PERIOD" CHAR(1 BYTE),

"ORDERED\_DATE" DATE,

"RECUR\_COSTS" NUMBER(10,2),

"TERM\_REASON" VARCHAR2(200 BYTE),

"TERM\_START" VARCHAR2(20 BYTE),

"TERM\_PLAN\_EXP" VARCHAR2(20 BYTE),

"LAST\_BILL\_DATE" DATE,

"TERM\_DURATION" NUMBER(13,3),

"ORDER\_NUM" VARCHAR2(50 BYTE),

"M2M\_YN" CHAR(1 BYTE),

"DATE\_BEGIN" DATE,

"RUN\_DATE" DATE,

"A\_SITE\_ID" NUMBER(10,0),

"A\_SITE\_NAME" VARCHAR2(100 BYTE),

"Z\_SITE\_ID" NUMBER(10,0),

"Z\_SITE\_NAME" VARCHAR2(100 BYTE),

"VENDOR" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEGMENT\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEGMENT\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"PATHS" NUMBER,

"SEGMENTS" NUMBER,

"SEGMENTS\_W\_PATH" NUMBER,

"SEGMENTS\_WO\_PATH" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEGS\_UPGRADED\_EBH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEGS\_UPGRADED\_EBH"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"SRC" CHAR(13 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEG\_INS\_EBH\_UPGRADED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEG\_INS\_EBH\_UPGRADED"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"SRC" CHAR(13 BYTE),

"CHG\_TS" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEG\_LIVE\_IN\_2010

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEG\_LIVE\_IN\_2010"

( "CIRC\_INST\_ID" NUMBER(9,0),

"A\_SITE\_ID" NUMBER(9,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEVONE\_CSR

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEVONE\_CSR"

( "DEVICE\_NAME" VARCHAR2(50 BYTE),

"SOURCE" VARCHAR2(50 BYTE),

"DEVICE\_IP" VARCHAR2(50 BYTE),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE),

"ERROR\_CODE" NUMBER,

"VENDOR" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEVONE\_CSR\_PARSED\_DATA\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEVONE\_CSR\_PARSED\_DATA\_WK"

( "DEVICE\_ID" NUMBER,

"DEVICE\_NAME" VARCHAR2(100 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"DEVICE\_IP" VARCHAR2(50 BYTE),

"ELEMENT\_TYPE" VARCHAR2(30 BYTE),

"TR\_TOKEN" VARCHAR2(20 BYTE) DEFAULT 0,

"EXTRACT\_DATE" DATE,

"PARSE\_STATUS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEVONE\_PARSED\_DATA\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEVONE\_PARSED\_DATA\_WK"

( "DEVICE\_ID" NUMBER,

"DEVICE\_NAME" VARCHAR2(100 BYTE),

"BANDWIDTH" VARCHAR2(30 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"DEVICE\_IP" VARCHAR2(50 BYTE),

"ELEMENT\_TYPE" VARCHAR2(30 BYTE),

"TR\_TOKEN" VARCHAR2(20 BYTE) DEFAULT 0,

"EXTRACT\_DATE" DATE,

"PARSE\_STATUS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SEVONE\_PERFORMANCE\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SEVONE\_PERFORMANCE\_DATA"

( "DEVICE\_ID" NUMBER(10,0),

"DEVICE\_NAME" VARCHAR2(50 BYTE),

"OBJECT\_NAME" VARCHAR2(256 BYTE),

"DEVICE\_IP" VARCHAR2(50 BYTE),

"INDICATOR\_NAME" VARCHAR2(50 BYTE),

"INDICATOR\_TYPE" VARCHAR2(50 BYTE),

"TIMESTAMP" VARCHAR2(50 BYTE),

"VALUE" FLOAT(126),

"DATA\_UNIT" VARCHAR2(25 BYTE),

"SYS\_CREATION\_DATE" DATE,

"SYS\_UPDATE\_DATE" DATE,

"OPERATOR\_ID" VARCHAR2(15 BYTE),

"APPLICATION\_ID" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SITE\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SITE\_AUDIT"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"ATTR\_NAME" VARCHAR2(100 BYTE),

"POPULATED\_COUNT" NUMBER,

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SITE\_INST\_GEOCODE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SITE\_INST\_GEOCODE"

( "SITE\_INST\_ID" NUMBER,

"LATITUDE" NUMBER,

"LONGITUDE" NUMBER,

"NEAREST\_AIRPORT\_CODE" CHAR(3 BYTE),

"NEAREST\_AIRPORT\_NAME" VARCHAR2(100 BYTE),

"NEAREST\_AIRPORT\_DIST\_IN\_MILES" NUMBER,

"UPDATED\_TIME" TIMESTAMP (6),

"QUALITY" NUMBER(2,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SITE\_INST\_GEOCODE\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SITE\_INST\_GEOCODE\_WK"

( "SITE\_INST\_ID" NUMBER,

"LATITUDE" NUMBER,

"LONGITUDE" NUMBER,

"NEAREST\_AIRPORT\_CODE" CHAR(3 BYTE),

"NEAREST\_AIRPORT\_NAME" VARCHAR2(100 BYTE),

"NEAREST\_AIRPORT\_DIST\_IN\_MILES" NUMBER,

"UPDATED\_TIME" TIMESTAMP (6),

"QUALITY" NUMBER(2,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SITE\_PORTAL\_DOMAIN\_MAPPING

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SITE\_PORTAL\_DOMAIN\_MAPPING"

( "SITE\_PORTAL\_DOMAIN\_CODE" VARCHAR2(10 CHAR),

"DOMAIN\_INST\_ID" NUMBER(10,0),

"SITE\_PORTAL\_DOMAIN\_MAPPING\_ID" NUMBER(10,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SITE\_PORTAL\_EXTRACT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SITE\_PORTAL\_EXTRACT"

( "REGION" VARCHAR2(20 BYTE),

"DATABASE\_NAME" VARCHAR2(20 BYTE),

"SERVER\_IP" VARCHAR2(60 BYTE),

"GROUP\_NAME" VARCHAR2(100 BYTE),

"VENDOR" VARCHAR2(100 BYTE),

"MODEL" VARCHAR2(100 BYTE),

"DEVICE\_NAME" VARCHAR2(100 BYTE),

"IP\_ADDRESS" VARCHAR2(60 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SITE\_PORTAL\_NO\_DUPS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SITE\_PORTAL\_NO\_DUPS"

( "REGION" VARCHAR2(20 BYTE),

"DATABASE\_NAME" VARCHAR2(20 BYTE),

"SERVER\_IP" VARCHAR2(60 BYTE),

"GROUP\_NAME" VARCHAR2(50 BYTE),

"VENDOR" VARCHAR2(50 BYTE),

"MODEL" VARCHAR2(50 BYTE),

"DEVICE\_NAME" VARCHAR2(100 BYTE),

"IP\_ADDRESS" VARCHAR2(60 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SITE\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SITE\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"SITE\_COUNT" NUMBER,

"EXTRACT\_DATE" DATE,

"MTSO\_COUNT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SS7\_AUDIT\_RAW\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SS7\_AUDIT\_RAW\_DATA"

( "STP\_PAIR" VARCHAR2(30 BYTE),

"STP" VARCHAR2(30 BYTE),

"STP\_PC" VARCHAR2(30 BYTE),

"STP\_CLLI" VARCHAR2(30 BYTE),

"EQUIP\_TYPE" VARCHAR2(30 BYTE),

"FAR\_NODE" VARCHAR2(30 BYTE),

"LINKSET" VARCHAR2(30 BYTE),

"LSTYPE" VARCHAR2(30 BYTE),

"NUM\_OF\_LINKS" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(30 BYTE),

"MARKET" VARCHAR2(30 BYTE),

"POINT\_CODE" VARCHAR2(30 BYTE),

"CLLI" VARCHAR2(30 BYTE),

"LINK\_NUM" VARCHAR2(30 BYTE),

"UTILIZATION" VARCHAR2(30 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table STRIPPED\_XNG\_CIRCUIT\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."STRIPPED\_XNG\_CIRCUIT\_DATA"

( "XNG\_CIRC\_INST\_ID" NUMBER(9,0),

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_CIRC\_STRIP" VARCHAR2(60 BYTE),

"INST\_VER" NUMBER(6,0),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_VENDOR\_STRIP" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_BAN" VARCHAR2(30 BYTE),

"XNG\_BAN\_STRIP" VARCHAR2(30 BYTE),

"XNG\_RECUR\_COSTS" NUMBER,

"XNG\_RECUR\_COSTS\_STRIP" NUMBER,

"XNG\_CIRC\_UNQ" CHAR(1 BYTE),

"XNG\_CIRC\_VENDOR\_UNQ" CHAR(1 BYTE),

"XNG\_CIRC\_VENDOR\_BAN\_UNQ" CHAR(1 BYTE),

"XNG\_BDATE\_VAL\_ATTR\_ID" NUMBER(9,0),

"XNG\_BILL\_DATE" DATE,

"XNG\_BILL\_DATE\_CHAR" VARCHAR2(20 BYTE),

"XNG\_TYPE" VARCHAR2(30 BYTE),

"XNG\_BANDWIDTH" VARCHAR2(30 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"XNG\_NEXT\_PATH\_INST\_ID" NUMBER(9,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 125829120 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SWITCH\_PATH\_ISSUE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SWITCH\_PATH\_ISSUE"

( "SWITCH\_PATH\_ISSUE\_ID" NUMBER(\*,0),

"DESCRIPTION" VARCHAR2(50 BYTE),

"IS\_CRITICAL" CHAR(1 BYTE) DEFAULT 'N',

"COMMENTS" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SWITCH\_PATH\_T1\_DACSPORTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_DACSPORTS"

( "SWITCH\_PATH\_T1\_TESTABILITY\_ID" NUMBER,

"CIRC\_PATH\_INST\_ID" NUMBER,

"EQUIP\_INST\_ID" NUMBER,

"PORT\_INST\_ID" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"IS\_VALID\_PORT\_NAME" CHAR(1 BYTE),

"IS\_VALID\_TEST\_HEAD\_NAME" CHAR(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SWITCH\_PATH\_T1\_ENDPOINTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_ENDPOINTS"

( "SWITCH\_PATH\_T1\_TESTABILITY\_ID" NUMBER,

"CIRC\_PATH\_INST\_ID" NUMBER,

"SWRTMSC\_EQ\_INST\_ID" NUMBER,

"SWRTMSC\_PORT\_INST\_ID" NUMBER,

"CELL\_EQ\_INST\_ID" NUMBER,

"CELL\_PORT\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SWITCH\_PATH\_T1\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"ACTIVE\_NE\_SPANS" NUMBER,

"ACTIVE\_SPANS\_LIVE\_IN\_XNG" NUMBER,

"ACTIVE\_SPANS\_NOT\_FOUND\_IN\_XNG" NUMBER,

"ACTIVE\_SPANS\_WITH\_EXTRA\_XNG" NUMBER,

"ACTIVE\_LIVE\_SPANS\_EXEMPT" NUMBER,

"ACTIVE\_LIVE\_SPANS\_NOT\_EXEMPT" NUMBER,

"ACTIVE\_LIVE\_SPANS\_TESTABLE" NUMBER,

"ACTIVE\_LIVE\_SPANS\_UNTESTABLE" NUMBER,

"PERCENTAGE\_T1\_TESTABLE" NUMBER,

"ACTIVE\_REPORTED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SWITCH\_PATH\_T1\_TESTABILITY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY"

( "SWITCH\_PATH\_T1\_TESTABILITY\_ID" NUMBER(\*,0),

"DOMAIN\_INST\_ID" VARCHAR2(50 BYTE),

"VENDOR" VARCHAR2(10 BYTE),

"SPAN\_TECHNOLOGY" VARCHAR2(20 BYTE),

"NETWORK\_ELEMENT\_ID" VARCHAR2(50 BYTE),

"DEVICE\_NAME" VARCHAR2(40 BYTE),

"IPBSCDO\_NUMBER" NUMBER(\*,0),

"IS\_ACTIVE\_IN\_NE" CHAR(1 BYTE),

"IS\_FOUND\_IN\_XNG" CHAR(1 BYTE),

"IS\_ACTIVE\_IN\_XNG" CHAR(1 BYTE),

"MATCHING\_LIVE\_XNG\_PATHS" NUMBER(\*,0) DEFAULT 0,

"CIRC\_PATH\_INST\_ID" NUMBER(\*,0),

"PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"A\_SIDE\_SITE\_INST\_ID" NUMBER(\*,0),

"SITE\_NAME" VARCHAR2(100 BYTE),

"DOM\_IP" VARCHAR2(20 BYTE),

"IS\_EXEMPT" CHAR(1 BYTE),

"IS\_TESTABLE" CHAR(1 BYTE),

"GUI\_LINK" VARCHAR2(200 BYTE),

"MATCH\_CODE" VARCHAR2(20 BYTE),

"EVALUATION\_DATE" DATE DEFAULT TRUNC(SYSDATE),

"EXEMPTION\_REASON" VARCHAR2(100 BYTE),

"IS\_BACKUP\_PATH" CHAR(1 BYTE) DEFAULT 'N'

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SWITCH\_PATH\_T1\_TESTABILITY\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK"

( "SWITCH\_PATH\_T1\_TESTABILITY\_ID" NUMBER(\*,0),

"DOMAIN\_INST\_ID" VARCHAR2(50 BYTE),

"VENDOR" VARCHAR2(10 BYTE),

"SPAN\_TECHNOLOGY" VARCHAR2(20 BYTE),

"NETWORK\_ELEMENT\_ID" VARCHAR2(50 BYTE),

"DEVICE\_NAME" VARCHAR2(40 BYTE),

"IPBSCDO\_NUMBER" NUMBER(\*,0),

"IS\_ACTIVE\_IN\_NE" CHAR(1 BYTE),

"IS\_FOUND\_IN\_XNG" CHAR(1 BYTE),

"IS\_ACTIVE\_IN\_XNG" CHAR(1 BYTE),

"MATCHING\_LIVE\_XNG\_PATHS" NUMBER(\*,0) DEFAULT 0,

"CIRC\_PATH\_INST\_ID" NUMBER(\*,0),

"PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"A\_SIDE\_SITE\_INST\_ID" NUMBER(\*,0),

"SITE\_NAME" VARCHAR2(100 BYTE),

"DOM\_IP" VARCHAR2(20 BYTE),

"IS\_EXEMPT" CHAR(1 BYTE),

"IS\_TESTABLE" CHAR(1 BYTE),

"GUI\_LINK" VARCHAR2(200 BYTE),

"MATCH\_CODE" VARCHAR2(20 BYTE),

"EVALUATION\_DATE" DATE DEFAULT TRUNC(SYSDATE),

"EXEMPTION\_REASON" VARCHAR2(100 BYTE),

"IS\_BACKUP\_PATH" CHAR(1 BYTE) DEFAULT 'N'

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table SWITCH\_PATH\_TESTABILITY\_ISSUE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."SWITCH\_PATH\_TESTABILITY\_ISSUE"

( "SWITCH\_PATH\_T1\_TESTABILITY\_ID" NUMBER(\*,0),

"SWITCH\_PATH\_ISSUE\_ID" NUMBER(\*,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table T1\_EXEMPTIONS\_ID\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."T1\_EXEMPTIONS\_ID\_DEL"

( "T1\_EXEMPTION\_ID" NUMBER,

"DESCRIPTION" VARCHAR2(100 BYTE),

"IS\_CRITICAL" CHAR(1 BYTE) DEFAULT 'N',

"COMMENTS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table T1\_ISSUES\_ID\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."T1\_ISSUES\_ID\_DEL"

( "T1\_ISSUE\_ID" NUMBER,

"DESCRIPTION" VARCHAR2(100 BYTE),

"IS\_CRITICAL" CHAR(1 BYTE) DEFAULT 'N',

"COMMENTS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TDM\_CKTS\_TO\_EBH\_SITES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TDM\_CKTS\_TO\_EBH\_SITES"

( "DOMAIN\_INST\_ID" NUMBER(9,0),

"CIRC\_INST\_ID" NUMBER(9,0),

"Z\_SITE\_ID" NUMBER(9,0),

"STATUS" VARCHAR2(20 BYTE),

"MRC" NUMBER(10,2),

"EBH\_IN\_SERVICE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TDM\_DISCONNECT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TDM\_DISCONNECT"

( "CIRC\_INST\_ID" NUMBER(\*,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"DISCONNECT\_DATE" DATE,

"DURATION" NUMBER(\*,0),

"PATH\_Z\_SITE\_ID" NUMBER(\*,0),

"DOMAIN\_NAME" VARCHAR2(50 BYTE),

"HAS\_BILLING" CHAR(1 BYTE) DEFAULT 'N',

"BTP\_IN\_SERVICE\_DATE" DATE,

"GIGE\_BTP\_IN\_SERVICE\_DATE" DATE,

"GIGE\_PATH\_INST\_ID" NUMBER(\*,0),

"GIGE\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"GIGE\_PATH\_Z\_SITE\_ID" NUMBER(\*,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TDM\_DISCONNECT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TDM\_DISCONNECT\_WK"

( "CIRC\_INST\_ID" NUMBER(\*,0),

"CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"DISCONNECT\_DATE" DATE,

"DURATION" NUMBER(\*,0),

"PATH\_Z\_SITE\_ID" NUMBER(\*,0),

"DOMAIN\_NAME" VARCHAR2(50 BYTE),

"HAS\_BILLING" CHAR(1 BYTE) DEFAULT 'N',

"BTP\_IN\_SERVICE\_DATE" DATE,

"GIGE\_BTP\_IN\_SERVICE\_DATE" DATE,

"GIGE\_PATH\_INST\_ID" NUMBER(\*,0),

"GIGE\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"GIGE\_PATH\_Z\_SITE\_ID" NUMBER(\*,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TE1ST

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TE1ST"

( "INST\_ID" NUMBER,

"UDA\_VALUE" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_BTP\_GI\_MATCH\_RPT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_BTP\_GI\_MATCH\_RPT"

( "XNG\_AREA" VARCHAR2(50 BYTE),

"XNG\_REGION" VARCHAR2(50 BYTE),

"STRIPPED\_EC\_CIRCUIT\_ID" VARCHAR2(60 BYTE),

"INVENTORY\_VENDOR\_CODE" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_1" VARCHAR2(50 BYTE),

"BAN\_CUSTOM\_FIELD\_4" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_1" VARCHAR2(50 BYTE),

"BAN\_CATEGORY\_2" VARCHAR2(50 BYTE),

"CHARGE\_AMOUNT" NUMBER,

"MASTER\_BAN" VARCHAR2(50 BYTE),

"BAN" VARCHAR2(50 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"MANUAL\_ENTRY" VARCHAR2(10 BYTE),

"XNG\_CIRC\_HUM\_ID" VARCHAR2(60 BYTE),

"XNG\_VENDOR" VARCHAR2(30 BYTE),

"XNG\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CIRC\_INST\_ID" NUMBER,

"UNIQUELY\_IDENTIFIED" VARCHAR2(1 BYTE),

"IN\_SERVICE\_DATE" VARCHAR2(10 BYTE),

"MAX\_CIRC\_DATE" VARCHAR2(10 BYTE),

"SUMM\_TOT" NUMBER,

"SUMM\_AMT" NUMBER,

"MAX\_AUDIT\_DATE" VARCHAR2(10 BYTE),

"STATUS\_DATE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_S2

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_S2"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_DESCRIPTION" VARCHAR2(150 BYTE),

"NAURU\_ID" VARCHAR2(25 BYTE),

"ACCESS\_RESTRICTIONS" VARCHAR2(4000 BYTE),

"MSC\_CSR\_PORT" VARCHAR2(4000 BYTE),

"CSR\_MAC\_ADDRESS" VARCHAR2(4000 BYTE),

"DRIVE\_DIRECTIONS" VARCHAR2(4000 BYTE),

"LOCAL\_FIRE\_PHONE" VARCHAR2(4000 BYTE),

"LOCAL\_FIRE\_PHONE\_COMMENTS" VARCHAR2(4000 BYTE),

"LOCAL\_POLICE\_PHONE" VARCHAR2(4000 BYTE),

"LOCAL\_POLICE\_PHONE\_COMMENTS" VARCHAR2(4000 BYTE),

"ENV\_ALARM\_TEST\_DATE" VARCHAR2(4000 BYTE),

"HVAC\_CONTACT\_PHONE" VARCHAR2(4000 BYTE),

"HVAC\_MAINT\_VENDOR" VARCHAR2(4000 BYTE),

"POWER\_ACCOUNT\_NUMBER" VARCHAR2(4000 BYTE),

"POWER\_PHONE" VARCHAR2(4000 BYTE),

"POWER\_COMPANY" VARCHAR2(4000 BYTE),

"POWER\_METER\_NUM" VARCHAR2(4000 BYTE),

"BUILDING\_TYPE" VARCHAR2(4000 BYTE),

"FIRE\_SUPRESSION\_TYPE" VARCHAR2(4000 BYTE),

"XNG\_PATH\_HYPERLINK" VARCHAR2(4000 BYTE),

"EXTERNAL\_TELCO\_NIU\_FLAG" VARCHAR2(4000 BYTE),

"GAS\_COMPANY" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_ACCT\_NUM" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_METER\_NUM" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_PHONE" VARCHAR2(4000 BYTE),

"HIGH\_VOLTAGE\_PROTECTION" VARCHAR2(4000 BYTE),

"POTS\_COMPANY" VARCHAR2(4000 BYTE),

"POSTS\_COMPANY\_ACCT\_NUM" VARCHAR2(4000 BYTE),

"POTS\_PHONE\_NUM" VARCHAR2(4000 BYTE),

"POTS\_COMPANY\_PHONE\_NUMBER" VARCHAR2(4000 BYTE),

"ENODEB\_IPV4" VARCHAR2(4000 BYTE),

"ENODEB\_IPV6" VARCHAR2(4000 BYTE),

"COPPER\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"FIBER\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"MICROWAVE\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"EBH\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"T1\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"XNG\_COMMENTS" VARCHAR2(4000 BYTE),

"CSR\_DEVICE\_IP" VARCHAR2(4000 BYTE),

"CSR\_SYSTEM\_IP" VARCHAR2(4000 BYTE),

"CSR\_MGMT\_IP" VARCHAR2(4000 BYTE),

"CSR\_LTE\_LOOPBACK\_IPV4" VARCHAR2(4000 BYTE),

"CSR\_LTE\_LOOPBACK\_IPV6" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_SAT\_A1

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_SAT\_A1"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"SITE\_DESCRIPTION" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_SAT\_S1

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_SAT\_S1"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"SITE\_DESCRIPTION" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_SAT\_S2

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_SAT\_S2"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"SITE\_DESCRIPTION" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_XNG\_CSR

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_XNG\_CSR"

( "SITE\_INST\_ID" NUMBER(9,0),

"CSR\_ID" VARCHAR2(100 BYTE),

"CSR\_DEVICE\_IP" VARCHAR2(3100 BYTE),

"CSR\_SYSTEM\_IP" VARCHAR2(3100 BYTE),

"CSR\_MGMT\_IP" VARCHAR2(3100 BYTE),

"CSR\_LTE\_LOOPBACK\_IPV4" VARCHAR2(3100 BYTE),

"CSR\_LTE\_LOOPBACK\_IPV6" VARCHAR2(3100 BYTE),

"CSR\_MAC\_ADDRESS" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_XNG\_CSR\_PORT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_XNG\_CSR\_PORT"

( "SITE\_INST\_ID" NUMBER(9,0),

"MSC\_CSR\_PORT" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_XNG\_ENODE\_B

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_DESCRIPTION" VARCHAR2(150 BYTE),

"EQUIP\_INST\_ID" NUMBER(10,0),

"DESCR" VARCHAR2(100 BYTE),

"NODE" VARCHAR2(100 BYTE),

"IP" VARCHAR2(50 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"XNG\_EQUIP\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_XNG\_ENODE\_B\_SATYA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B\_SATYA"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_DESCRIPTION" VARCHAR2(150 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"DESCR" VARCHAR2(100 BYTE),

"NODE" VARCHAR2(100 BYTE),

"IP" VARCHAR2(50 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"XNG\_EQUIP\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEMP\_XOR\_20131015

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEMP\_XOR\_20131015"

( "REPORT\_ID" NUMBER(5,0),

"REPORT\_NAME" VARCHAR2(50 BYTE),

"PROJECT\_NAME" VARCHAR2(50 BYTE),

"HOSTED\_APPLICATION" VARCHAR2(50 BYTE),

"HOSTED\_SERVER" VARCHAR2(50 BYTE),

"RELATIVE\_URL" VARCHAR2(200 BYTE),

"DESCRIPTION" VARCHAR2(1024 BYTE),

"REL\_ORDER" NUMBER(3,0),

"OPEN\_OPTION" NUMBER(1,0),

"REP\_LEVEL" VARCHAR2(10 BYTE),

"NEXT\_PAGE" VARCHAR2(100 BYTE),

"PERMANENT\_REWRITE\_NAME" VARCHAR2(100 BYTE),

"HOSTED\_PORT" VARCHAR2(10 BYTE),

"CREATE\_HTACCESS\_REWRITE" CHAR(1 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEST\_HEAD\_QT\_XREF

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF"

( "TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"STATUS" VARCHAR2(10 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(30 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEST\_HEAD\_QT\_XREF\_BAK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF\_BAK"

( "TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"STATUS" VARCHAR2(10 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(30 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEST\_HEAD\_XREF

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEST\_HEAD\_XREF"

( "TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"STATUS" VARCHAR2(10 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"AREA" VARCHAR2(30 BYTE),

"REGION" VARCHAR2(30 BYTE),

"ALLTEL\_DO\_NOT\_REMOVE" VARCHAR2(1 BYTE),

"ALLTEL\_CENTEST\_TID" VARCHAR2(50 BYTE),

"ALLTEL\_CENTEST\_IP" VARCHAR2(15 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEST\_SSS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEST\_SSS"

( "CSR\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"CSR\_STATUS" VARCHAR2(20 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"VLAN\_INST\_ID" NUMBER,

"VLAN\_HUM\_ID" VARCHAR2(100 BYTE),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"GI\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"BRIX\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"GI\_VLAN" NUMBER,

"BRIX\_VLAN" NUMBER,

"BRIX\_FD" VARCHAR2(100 BYTE),

"GI\_FD" VARCHAR2(3100 BYTE),

"GI\_TEST\_TYPE" VARCHAR2(3100 BYTE),

"GI\_BC" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEST\_SSSS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEST\_SSSS"

( "CSR\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"CSR\_STATUS" VARCHAR2(20 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"VLAN\_INST\_ID" NUMBER,

"VLAN\_HUM\_ID" VARCHAR2(100 BYTE),

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"GI\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"BRIX\_CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"GI\_VLAN" NUMBER,

"BRIX\_VLAN" NUMBER,

"BRIX\_FD" VARCHAR2(100 BYTE),

"GI\_FD" VARCHAR2(3100 BYTE),

"GI\_TEST\_TYPE" VARCHAR2(3100 BYTE),

"GI\_BC" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEST\_T1\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEST\_T1\_DEL"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"ACTIVE\_NE\_SPANS" NUMBER,

"ACTIVE\_SPANS\_LIVE\_IN\_XNG" NUMBER,

"ACTIVE\_SPANS\_NOT\_FOUND\_IN\_XNG" NUMBER,

"ACTIVE\_SPANS\_WITH\_EXTRA\_XNG" NUMBER,

"ACTIVE\_LIVE\_SPANS\_EXEMPT" NUMBER,

"ACTIVE\_LIVE\_SPANS\_NOT\_EXEMPT" NUMBER,

"ACTIVE\_LIVE\_SPANS\_TESTABLE" NUMBER,

"ACTIVE\_LIVE\_SPANS\_UNTESTABLE" NUMBER,

"PERCENTAGE\_T1\_TESTABLE" NUMBER,

"ACTIVE\_REPORTED" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TEST\_TABLE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TEST\_TABLE"

( "T\_ID" NUMBER

) SEGMENT CREATION DEFERRED

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TMP\_XOR\_MENUS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TMP\_XOR\_MENUS"

( "MENU\_ID" NUMBER(5,0),

"MENU\_NAME" VARCHAR2(50 BYTE),

"PARENT\_MENU\_ID" NUMBER(5,0),

"REL\_ORDER" NUMBER(3,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TMP\_XOR\_MENUS\_REPORTS\_LINK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TMP\_XOR\_MENUS\_REPORTS\_LINK"

( "MENU\_ID" NUMBER(5,0),

"REPORT\_ID" NUMBER(5,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TMP\_XOR\_REPORTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TMP\_XOR\_REPORTS"

( "REPORT\_ID" NUMBER(5,0),

"REPORT\_NAME" VARCHAR2(50 BYTE),

"PROJECT\_NAME" VARCHAR2(50 BYTE),

"HOSTED\_APPLICATION" VARCHAR2(50 BYTE),

"HOSTED\_SERVER" VARCHAR2(50 BYTE),

"RELATIVE\_URL" VARCHAR2(200 BYTE),

"DESCRIPTION" VARCHAR2(1024 BYTE),

"REL\_ORDER" NUMBER(3,0),

"OPEN\_OPTION" NUMBER(1,0),

"REP\_LEVEL" VARCHAR2(10 BYTE),

"NEXT\_PAGE" VARCHAR2(100 BYTE),

"PERMANENT\_REWRITE\_NAME" VARCHAR2(100 BYTE),

"HOSTED\_PORT" VARCHAR2(10 BYTE),

"CREATE\_HTACCESS\_REWRITE" CHAR(1 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TMP\_XOR\_TABS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TMP\_XOR\_TABS"

( "TAB\_ID" NUMBER(5,0),

"TAB\_NAME" VARCHAR2(30 BYTE),

"REL\_ORDER" NUMBER(3,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TMP\_XOR\_TABS\_MENUS\_LINK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TMP\_XOR\_TABS\_MENUS\_LINK"

( "TAB\_ID" NUMBER(5,0),

"MENU\_ID" NUMBER(5,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table TOAD\_PLAN\_TABLE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."TOAD\_PLAN\_TABLE"

( "STATEMENT\_ID" VARCHAR2(30 BYTE),

"PLAN\_ID" NUMBER,

"TIMESTAMP" DATE,

"REMARKS" VARCHAR2(4000 BYTE),

"OPERATION" VARCHAR2(30 BYTE),

"OPTIONS" VARCHAR2(255 BYTE),

"OBJECT\_NODE" VARCHAR2(128 BYTE),

"OBJECT\_OWNER" VARCHAR2(30 BYTE),

"OBJECT\_NAME" VARCHAR2(30 BYTE),

"OBJECT\_ALIAS" VARCHAR2(65 BYTE),

"OBJECT\_INSTANCE" NUMBER(\*,0),

"OBJECT\_TYPE" VARCHAR2(30 BYTE),

"OPTIMIZER" VARCHAR2(255 BYTE),

"SEARCH\_COLUMNS" NUMBER,

"ID" NUMBER(\*,0),

"PARENT\_ID" NUMBER(\*,0),

"DEPTH" NUMBER(\*,0),

"POSITION" NUMBER(\*,0),

"COST" NUMBER(\*,0),

"CARDINALITY" NUMBER(\*,0),

"BYTES" NUMBER(\*,0),

"OTHER\_TAG" VARCHAR2(255 BYTE),

"PARTITION\_START" VARCHAR2(255 BYTE),

"PARTITION\_STOP" VARCHAR2(255 BYTE),

"PARTITION\_ID" NUMBER(\*,0),

"OTHER" LONG,

"DISTRIBUTION" VARCHAR2(30 BYTE),

"CPU\_COST" NUMBER(\*,0),

"IO\_COST" NUMBER(\*,0),

"TEMP\_SPACE" NUMBER(\*,0),

"ACCESS\_PREDICATES" VARCHAR2(4000 BYTE),

"FILTER\_PREDICATES" VARCHAR2(4000 BYTE),

"PROJECTION" VARCHAR2(4000 BYTE),

"TIME" NUMBER(\*,0),

"QBLOCK\_NAME" VARCHAR2(30 BYTE),

"OTHER\_XML" CLOB

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00"

LOB ("OTHER\_XML") STORE AS BASICFILE (

TABLESPACE "SS\_DATA00" ENABLE STORAGE IN ROW CHUNK 16384 RETENTION

NOCACHE LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)) ;

--------------------------------------------------------

-- DDL for Table UNQ\_XNG\_CONTACTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."UNQ\_XNG\_CONTACTS"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"ADDRESS" VARCHAR2(4000 BYTE),

"CITY" VARCHAR2(60 BYTE),

"STATE\_PROV" VARCHAR2(40 BYTE),

"POST\_CODE\_1" VARCHAR2(10 BYTE),

"SITE\_MANAGER" VARCHAR2(255 BYTE),

"SITE\_MGR\_PHONE" VARCHAR2(3100 BYTE),

"SITE\_TECHNICIAN" VARCHAR2(255 BYTE),

"SITE\_TECH\_PHONE" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table URL\_STATS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."URL\_STATS"

( "URL" VARCHAR2(1000 BYTE),

"PAGES" NUMBER,

"PAGES\_PERCENT" VARCHAR2(25 BYTE),

"HITS" NUMBER,

"HITS\_PERCENT" VARCHAR2(25 BYTE),

"MONTH" VARCHAR2(3 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table USER\_CIRC\_DOMAIN\_PRIMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."USER\_CIRC\_DOMAIN\_PRIMARY"

( "CIRC\_INST\_ID" NUMBER(9,0),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"DOMAIN\_INST\_ID" NUMBER(9,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table USER\_PATH\_DOMAIN\_PRIMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."USER\_PATH\_DOMAIN\_PRIMARY"

( "CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"DOMAIN\_INST\_ID" NUMBER(9,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VPI\_DATA

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VPI\_DATA"

( "TERRITORY" VARCHAR2(500 BYTE),

"MARKET" VARCHAR2(500 BYTE),

"SUB\_MARKET" VARCHAR2(500 BYTE),

"MORPHOLOGY" VARCHAR2(500 BYTE),

"CMA\_NO" VARCHAR2(500 BYTE),

"CMA\_NAME" VARCHAR2(500 BYTE),

"UACE" VARCHAR2(500 BYTE),

"UACE\_NAME" VARCHAR2(500 BYTE),

"NWID" VARCHAR2(500 BYTE),

"ENODEB\_GRP" VARCHAR2(500 BYTE),

"SITE\_NAME" VARCHAR2(500 BYTE),

"COUNTY" VARCHAR2(500 BYTE),

"STATE" VARCHAR2(500 BYTE),

"LATITUDE" VARCHAR2(500 BYTE),

"LONGITUDE" VARCHAR2(500 BYTE),

"MARKET\_NAME" VARCHAR2(500 BYTE),

"MKT\_ID" VARCHAR2(500 BYTE),

"PSLC" VARCHAR2(500 BYTE),

"S\_C\_COUNT" VARCHAR2(500 BYTE),

"TOP\_CMA\_UACE" VARCHAR2(500 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VPI\_DATA\_20160203

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VPI\_DATA\_20160203"

( "TERRITORY" VARCHAR2(500 BYTE),

"MARKET" VARCHAR2(500 BYTE),

"SUB\_MARKET" VARCHAR2(500 BYTE),

"MORPHOLOGY" VARCHAR2(500 BYTE),

"CMA\_NO" VARCHAR2(500 BYTE),

"CMA\_NAME" VARCHAR2(500 BYTE),

"UACE" VARCHAR2(500 BYTE),

"UACE\_NAME" VARCHAR2(500 BYTE),

"NWID" VARCHAR2(500 BYTE),

"ENODEB\_GRP" VARCHAR2(500 BYTE),

"SITE\_NAME" VARCHAR2(500 BYTE),

"COUNTY" VARCHAR2(500 BYTE),

"STATE" VARCHAR2(500 BYTE),

"LATITUDE" VARCHAR2(500 BYTE),

"LONGITUDE" VARCHAR2(500 BYTE),

"MARKET\_NAME" VARCHAR2(500 BYTE),

"MKT\_ID" VARCHAR2(500 BYTE),

"PSLC" VARCHAR2(500 BYTE),

"S\_C\_COUNT" VARCHAR2(500 BYTE),

"TOP\_CMA\_UACE" VARCHAR2(500 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VPI\_GI\_ALT\_VALS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VPI\_GI\_ALT\_VALS"

( "SITE\_INST\_ID" NUMBER(10,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SITE\_HUM\_ID1" VARCHAR2(100 BYTE),

"SITE\_HUM\_ID2" VARCHAR2(100 BYTE),

"LATITUDE" VARCHAR2(20 BYTE),

"LONGITUDE" VARCHAR2(20 BYTE),

"LATITUDE2" VARCHAR2(4000 BYTE),

"LONGITUDE2" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VPI\_GI\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VPI\_GI\_AUDIT"

( "TERRITORY" VARCHAR2(500 BYTE),

"MARKET" VARCHAR2(500 BYTE),

"SUB\_MARKET" VARCHAR2(500 BYTE),

"MORPHOLOGY" VARCHAR2(500 BYTE),

"CMA\_NO" VARCHAR2(500 BYTE),

"CMA\_NAME" VARCHAR2(500 BYTE),

"UACE" VARCHAR2(500 BYTE),

"UACE\_NAME" VARCHAR2(500 BYTE),

"NWID" VARCHAR2(500 BYTE),

"ENODEB\_GRP" VARCHAR2(500 BYTE),

"SITE\_NAME" VARCHAR2(500 BYTE),

"COUNTY" VARCHAR2(500 BYTE),

"STATE" VARCHAR2(500 BYTE),

"LATITUDE" VARCHAR2(500 BYTE),

"LONGITUDE" VARCHAR2(500 BYTE),

"MARKET\_NAME" VARCHAR2(500 BYTE),

"MKT\_ID" VARCHAR2(500 BYTE),

"PSLC" VARCHAR2(500 BYTE),

"S\_C\_COUNT" VARCHAR2(500 BYTE),

"TOP\_CMA\_UACE" VARCHAR2(500 BYTE),

"VPI\_EXTRACT\_DATE" DATE,

"GI\_SITE\_ID" NUMBER,

"GI\_SITE\_NAME" VARCHAR2(100 BYTE),

"GI\_SITE\_TYPE" VARCHAR2(30 BYTE),

"GI\_SITE\_STATUS" VARCHAR2(20 BYTE),

"GI\_PSLC" VARCHAR2(3100 BYTE),

"GI\_COUNTY" VARCHAR2(40 BYTE),

"GI\_STATE\_PROV" VARCHAR2(40 BYTE),

"GI\_LAT" VARCHAR2(20 BYTE),

"GI\_LONG" VARCHAR2(20 BYTE),

"GI\_NAURU\_ID" VARCHAR2(40 BYTE),

"GI\_LOC\_CAT" VARCHAR2(3100 BYTE),

"GI\_EXTRACT\_DATE" DATE,

"GI\_DF\_VENDOR" VARCHAR2(30 BYTE),

"GI\_DF\_STATUS" VARCHAR2(30 BYTE),

"GI\_MW\_VENDOR" VARCHAR2(30 BYTE),

"GI\_MW\_STATUS" VARCHAR2(30 BYTE),

"GI\_LS\_VENDOR" VARCHAR2(30 BYTE),

"GI\_LS\_STATUS" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VPI\_GI\_AUDIT\_FIXES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VPI\_GI\_AUDIT\_FIXES"

( "TERRITORY" VARCHAR2(500 BYTE),

"MARKET" VARCHAR2(500 BYTE),

"SUB\_MARKET" VARCHAR2(500 BYTE),

"MORPHOLOGY" VARCHAR2(500 BYTE),

"CMA\_NO" VARCHAR2(500 BYTE),

"CMA\_NAME" VARCHAR2(500 BYTE),

"UACE" VARCHAR2(500 BYTE),

"UACE\_NAME" VARCHAR2(500 BYTE),

"NWID" VARCHAR2(500 BYTE),

"ENODEB\_GRP" VARCHAR2(500 BYTE),

"SITE\_NAME" VARCHAR2(500 BYTE),

"COUNTY" VARCHAR2(500 BYTE),

"STATE" VARCHAR2(500 BYTE),

"LATITUDE" VARCHAR2(500 BYTE),

"LONGITUDE" VARCHAR2(500 BYTE),

"MARKET\_NAME" VARCHAR2(500 BYTE),

"MKT\_ID" VARCHAR2(500 BYTE),

"PSLC" VARCHAR2(500 BYTE),

"S\_C\_COUNT" VARCHAR2(500 BYTE),

"TOP\_CMA\_UACE" VARCHAR2(500 BYTE),

"GI\_SITE\_INST\_ID" NUMBER(10,0),

"GI\_SITE\_NAME" VARCHAR2(100 BYTE),

"GI\_SITE\_ID" VARCHAR2(30 BYTE),

"GI\_SITE\_TYPE" VARCHAR2(30 BYTE),

"GI\_STATUS" VARCHAR2(20 BYTE),

"GI\_PSLC" VARCHAR2(3100 BYTE),

"GI\_COUNTY" VARCHAR2(40 BYTE),

"GI\_STATE\_PROV" VARCHAR2(40 BYTE),

"GI\_LAT" VARCHAR2(20 BYTE),

"GI\_LONG" VARCHAR2(20 BYTE),

"GI\_NAURU\_ID" VARCHAR2(40 BYTE),

"GI\_SEG\_VENDORS" VARCHAR2(318 BYTE),

"GI\_LOC\_CAT" VARCHAR2(3100 BYTE),

"MATCH\_TYPE" VARCHAR2(28 BYTE),

"MATCH\_ORD" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VPI\_GI\_AUDIT\_UNQ

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VPI\_GI\_AUDIT\_UNQ"

( "TERRITORIES" VARCHAR2(1000 BYTE),

"MARKETS" VARCHAR2(1000 BYTE),

"SUB\_MARKETS" VARCHAR2(1000 BYTE),

"MORPHOLOGY" VARCHAR2(500 BYTE),

"CMA\_NO" VARCHAR2(500 BYTE),

"CMA\_NAME" VARCHAR2(500 BYTE),

"UACE" VARCHAR2(500 BYTE),

"UACE\_NAME" VARCHAR2(500 BYTE),

"NWID" VARCHAR2(500 BYTE),

"ENODEB\_GRP" VARCHAR2(500 BYTE),

"SITE\_NAME" VARCHAR2(500 BYTE),

"COUNTY" VARCHAR2(500 BYTE),

"STATE" VARCHAR2(500 BYTE),

"LATITUDE" VARCHAR2(500 BYTE),

"LONGITUDE" VARCHAR2(500 BYTE),

"MARKET\_NAME" VARCHAR2(500 BYTE),

"MKT\_ID" VARCHAR2(500 BYTE),

"PSLC" VARCHAR2(500 BYTE),

"S\_C\_COUNT" VARCHAR2(500 BYTE),

"TOP\_CMA\_UACE" VARCHAR2(500 BYTE),

"VPI\_EXTRACT\_DATE" DATE,

"GI\_SITE\_ID" NUMBER,

"GI\_SITE\_NAME" VARCHAR2(100 BYTE),

"GI\_SITE\_TYPE" VARCHAR2(30 BYTE),

"GI\_SITE\_STATUS" VARCHAR2(20 BYTE),

"GI\_LOC\_CAT" VARCHAR2(3100 BYTE),

"GI\_EXTRACT\_DATE" DATE,

"GI\_DF\_VENDORS" VARCHAR2(1000 BYTE),

"GI\_DF\_STATUS" VARCHAR2(30 BYTE),

"GI\_MW\_VENDORS" VARCHAR2(1000 BYTE),

"GI\_MW\_STATUS" VARCHAR2(30 BYTE),

"GI\_LS\_VENDORS" VARCHAR2(1000 BYTE),

"GI\_LS\_STATUS" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VPI\_GI\_AUDIT\_UNQ\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VPI\_GI\_AUDIT\_UNQ\_WK"

( "TERRITORIES" VARCHAR2(1000 BYTE),

"MARKETS" VARCHAR2(1000 BYTE),

"SUB\_MARKETS" VARCHAR2(1000 BYTE),

"MORPHOLOGY" VARCHAR2(500 BYTE),

"CMA\_NO" VARCHAR2(500 BYTE),

"CMA\_NAME" VARCHAR2(500 BYTE),

"UACE" VARCHAR2(500 BYTE),

"UACE\_NAME" VARCHAR2(500 BYTE),

"NWID" VARCHAR2(500 BYTE),

"ENODEB\_GRP" VARCHAR2(500 BYTE),

"SITE\_NAME" VARCHAR2(500 BYTE),

"COUNTY" VARCHAR2(500 BYTE),

"STATE" VARCHAR2(500 BYTE),

"LATITUDE" VARCHAR2(500 BYTE),

"LONGITUDE" VARCHAR2(500 BYTE),

"MARKET\_NAME" VARCHAR2(500 BYTE),

"MKT\_ID" VARCHAR2(500 BYTE),

"PSLC" VARCHAR2(500 BYTE),

"S\_C\_COUNT" VARCHAR2(500 BYTE),

"TOP\_CMA\_UACE" VARCHAR2(500 BYTE),

"VPI\_EXTRACT\_DATE" DATE,

"GI\_SITE\_ID" NUMBER,

"GI\_SITE\_NAME" VARCHAR2(100 BYTE),

"GI\_SITE\_TYPE" VARCHAR2(30 BYTE),

"GI\_SITE\_STATUS" VARCHAR2(20 BYTE),

"GI\_LOC\_CAT" VARCHAR2(3100 BYTE),

"GI\_EXTRACT\_DATE" DATE,

"GI\_DF\_VENDORS" VARCHAR2(1000 BYTE),

"GI\_DF\_STATUS" VARCHAR2(30 BYTE),

"GI\_MW\_VENDORS" VARCHAR2(1000 BYTE),

"GI\_MW\_STATUS" VARCHAR2(30 BYTE),

"GI\_LS\_VENDORS" VARCHAR2(1000 BYTE),

"GI\_LS\_STATUS" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VPI\_GI\_AUDIT\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VPI\_GI\_AUDIT\_WK"

( "TERRITORY" VARCHAR2(500 BYTE),

"MARKET" VARCHAR2(500 BYTE),

"SUB\_MARKET" VARCHAR2(500 BYTE),

"MORPHOLOGY" VARCHAR2(500 BYTE),

"CMA\_NO" VARCHAR2(500 BYTE),

"CMA\_NAME" VARCHAR2(500 BYTE),

"UACE" VARCHAR2(500 BYTE),

"UACE\_NAME" VARCHAR2(500 BYTE),

"NWID" VARCHAR2(500 BYTE),

"ENODEB\_GRP" VARCHAR2(500 BYTE),

"SITE\_NAME" VARCHAR2(500 BYTE),

"COUNTY" VARCHAR2(500 BYTE),

"STATE" VARCHAR2(500 BYTE),

"LATITUDE" VARCHAR2(500 BYTE),

"LONGITUDE" VARCHAR2(500 BYTE),

"MARKET\_NAME" VARCHAR2(500 BYTE),

"MKT\_ID" VARCHAR2(500 BYTE),

"PSLC" VARCHAR2(500 BYTE),

"S\_C\_COUNT" VARCHAR2(500 BYTE),

"TOP\_CMA\_UACE" VARCHAR2(500 BYTE),

"VPI\_EXTRACT\_DATE" DATE,

"GI\_SITE\_ID" NUMBER,

"GI\_SITE\_NAME" VARCHAR2(100 BYTE),

"GI\_SITE\_TYPE" VARCHAR2(30 BYTE),

"GI\_SITE\_STATUS" VARCHAR2(20 BYTE),

"GI\_PSLC" VARCHAR2(3100 BYTE),

"GI\_COUNTY" VARCHAR2(40 BYTE),

"GI\_STATE\_PROV" VARCHAR2(40 BYTE),

"GI\_LAT" VARCHAR2(20 BYTE),

"GI\_LONG" VARCHAR2(20 BYTE),

"GI\_NAURU\_ID" VARCHAR2(40 BYTE),

"GI\_LOC\_CAT" VARCHAR2(3100 BYTE),

"GI\_EXTRACT\_DATE" DATE,

"GI\_DF\_VENDOR" VARCHAR2(30 BYTE),

"GI\_DF\_STATUS" VARCHAR2(30 BYTE),

"GI\_MW\_VENDOR" VARCHAR2(30 BYTE),

"GI\_MW\_STATUS" VARCHAR2(30 BYTE),

"GI\_LS\_VENDOR" VARCHAR2(30 BYTE),

"GI\_LS\_STATUS" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_CONNECTIONS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_CONNECTIONS"

( "VSM\_INSTANCE\_NAME" VARCHAR2(50 BYTE),

"VSM\_DB\_SERVER" VARCHAR2(128 BYTE),

"VSM\_DB\_IPADDRESS" VARCHAR2(15 BYTE),

"VSM\_DB\_USER" VARCHAR2(30 BYTE),

"VSM\_DB\_PASSWORD" VARCHAR2(128 BYTE),

"VSM\_DB\_PORT" NUMBER,

"VSM\_DB\_SID" VARCHAR2(30 BYTE),

"VSM\_DB\_VERSION" VARCHAR2(10 BYTE),

"CONNECTION\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_CONNECTIONS\_BAK\_NSP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_CONNECTIONS\_BAK\_NSP"

( "VSM\_INSTANCE\_NAME" VARCHAR2(50 BYTE),

"VSM\_DB\_SERVER" VARCHAR2(128 BYTE),

"VSM\_DB\_IPADDRESS" VARCHAR2(15 BYTE),

"VSM\_DB\_USER" VARCHAR2(30 BYTE),

"VSM\_DB\_PASSWORD" VARCHAR2(128 BYTE),

"VSM\_DB\_PORT" NUMBER,

"VSM\_DB\_SID" VARCHAR2(30 BYTE),

"VSM\_DB\_VERSION" VARCHAR2(10 BYTE),

"CONNECTION\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_DEVICES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_DEVICES"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE),

"LOAD\_DATE" DATE,

"VSM\_GATEWAY\_NAME" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_DEVICES\_BAK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_DEVICES\_BAK"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE),

"LOAD\_DATE" DATE,

"VSM\_GATEWAY\_NAME" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_DEVICES\_BLOCKED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_DEVICES\_BLOCKED"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"COMMENTS" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_DEVICES\_DOMAIN\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_DEVICES\_DOMAIN\_MAP"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE),

"AS\_OF\_DATE" DATE,

"ALLTEL\_DO\_NOT\_REMOVE" VARCHAR2(1 BYTE),

"LEAF\_DOMAIN\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_DEVICES\_IPADDRESSES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_DEVICES\_IPADDRESSES"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_IP" VARCHAR2(254 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_DEVICES\_IPADDRESSES\_BAK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_DEVICES\_IPADDRESSES\_BAK"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_IP" VARCHAR2(254 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_DEVICES\_IPADDRESSES\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_DEVICES\_IPADDRESSES\_WRK"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_IP" VARCHAR2(254 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_DEVICES\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_DEVICES\_WRK"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE),

"LOAD\_DATE" DATE,

"VSM\_GATEWAY\_NAME" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_SANE\_NAME\_EMS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_SANE\_NAME\_EMS"

( "VSM\_DEVICE\_NAME\_EMS" VARCHAR2(100 BYTE),

"SANE\_NAME\_EMS" VARCHAR2(100 BYTE),

"EMS\_PATH" VARCHAR2(100 BYTE),

"EMS\_STATUS" VARCHAR2(100 BYTE),

"CREATE\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VSM\_SANE\_RAW\_NAME\_BSM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VSM\_SANE\_RAW\_NAME\_BSM"

( "VSM\_DEVICE\_NAME\_BSM" VARCHAR2(100 BYTE),

"SANE\_NAME\_BSM" VARCHAR2(100 BYTE),

"RAW\_NAME\_BSM" VARCHAR2(100 BYTE),

"BSM\_PATH" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_EHEALTH\_TRAFFIC

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_EHEALTH\_TRAFFIC"

( "HOSTNAME" VARCHAR2(30 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"HAS\_TRAFFIC" CHAR(1 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_EHEALTH\_TRAFFIC\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_EHEALTH\_TRAFFIC\_WK"

( "HOSTNAME" VARCHAR2(30 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"HAS\_TRAFFIC" CHAR(1 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_INST\_AUDIT\_WINDOW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW"

( "ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"INST\_ID" NUMBER(9,0),

"VERSION\_CHGD" NUMBER(6,0),

"EDIT\_OPERATION" CHAR(1 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 BYTE),

"CHG\_DESC" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_INST\_AUDIT\_WINDOW\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW\_WRK"

( "ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"INST\_ID" NUMBER(9,0),

"VERSION\_CHGD" NUMBER(6,0),

"EDIT\_OPERATION" CHAR(1 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(50 BYTE),

"CHG\_DESC" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_LUCENT\_CELL\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_LUCENT\_CELL\_PATHS"

( "LOCALE" VARCHAR2(8 BYTE),

"PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"SWITCH\_ID" VARCHAR2(8 BYTE),

"CELL\_NAME" VARCHAR2(100 BYTE),

"CELL\_NUMBER" NUMBER,

"URC\_CRC" NUMBER,

"DS1\_NUMBER" NUMBER,

"XNG\_PATH\_STATUS" VARCHAR2(20 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(10 BYTE),

"PARSE\_STATUS" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_MOTO\_EVDO\_CELL\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_MOTO\_EVDO\_CELL\_PATHS"

( "PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"IPBSCDO\_NUMBER" NUMBER,

"MCCDO\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER,

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(100 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"BANDWIDTH" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_MOTO\_UBS\_CELL\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_MOTO\_UBS\_CELL\_PATHS"

( "PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"LOCALE" VARCHAR2(10 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"IPBSCDO\_NUMBER" NUMBER,

"BTS\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER,

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(10 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"BANDWIDTH" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_MOTO\_VOICE\_CELL\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_MOTO\_VOICE\_CELL\_PATHS"

( "PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"LOCALE" VARCHAR2(10 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"BTS\_NUMBER" NUMBER,

"CLUSTER\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER,

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(100 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"BANDWIDTH" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_NETWORK\_ORG

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG"

( "INST\_ID" NUMBER,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"REGION\_DOMAIN\_NAME" VARCHAR2(50 BYTE),

"MARKET\_DOMAIN\_NAME" VARCHAR2(50 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"TERRITORY" VARCHAR2(100 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_NETWORK\_ORG\_NEW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_NEW"

( "INST\_ID" NUMBER,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"REGION\_DOMAIN\_NAME" VARCHAR2(50 BYTE),

"MARKET\_DOMAIN\_NAME" VARCHAR2(50 BYTE),

"MARKET\_TERRITORY" VARCHAR2(50 BYTE),

"TERRITORY" VARCHAR2(100 BYTE),

"SUB\_MARKET" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_NETWORK\_ORG\_OLD

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_OLD"

( "INST\_ID" NUMBER,

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"MARKET" VARCHAR2(50 BYTE),

"REGION\_DOMAIN\_NAME" VARCHAR2(50 BYTE),

"MARKET\_DOMAIN\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_NORTEL\_CELL\_PATHS\_ADHOC

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_NORTEL\_CELL\_PATHS\_ADHOC"

( "VSM\_NAME\_MTX" VARCHAR2(100 BYTE),

"COMMON\_STRING" VARCHAR2(50 BYTE),

"DCG\_NUMBER" NUMBER,

"SLOT\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER,

"DOM\_NODE\_IP" VARCHAR2(50 BYTE),

"PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(100 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"LOCALE" VARCHAR2(10 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_NT\_EVDO\_CELL\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_NT\_EVDO\_CELL\_PATHS"

( "PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"LOCALE" VARCHAR2(10 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"SLOT\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER,

"DOM\_NODE\_IP" VARCHAR2(100 BYTE),

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(100 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"BANDWIDTH" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_NT\_VOICE\_CELL\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_NT\_VOICE\_CELL\_PATHS"

( "PATH\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"LOCALE" VARCHAR2(10 BYTE),

"SWITCH\_ID" VARCHAR2(100 BYTE),

"DCG\_NUMBER" NUMBER,

"SPAN\_NUMBER" NUMBER,

"XNG\_PATH\_STATUS" VARCHAR2(100 BYTE),

"XNG\_PATH\_TYPE" VARCHAR2(100 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"BANDWIDTH" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_SEVONE\_TRAFFIC

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_SEVONE\_TRAFFIC"

( "HOSTNAME" VARCHAR2(30 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"HAS\_TRAFFIC" CHAR(1 BYTE),

"EXTRACT\_DATE" DATE,

"ELEMENT\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_SEVONE\_TRAFFIC\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_SEVONE\_TRAFFIC\_WK"

( "HOSTNAME" VARCHAR2(30 BYTE),

"PORT\_NAME" VARCHAR2(100 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"HAS\_TRAFFIC" CHAR(1 BYTE),

"EXTRACT\_DATE" DATE,

"ELEMENT\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_VSM\_XNG\_XREF\_12142010

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_VSM\_XNG\_XREF\_12142010"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE),

"ALLTEL\_DO\_NOT\_REMOVE" VARCHAR2(1 BYTE),

"AS\_OF\_DATE" DATE,

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"CITY" VARCHAR2(40 BYTE),

"COUNTY" VARCHAR2(40 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"DOMAIN\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"MARKET" VARCHAR2(100 BYTE),

"MARKET\_INST\_ID" NUMBER,

"OLD\_SITE\_INST\_ID" NUMBER,

"SITE\_INST\_ID" NUMBER,

"SITE\_HUM\_ID" VARCHAR2(50 BYTE),

"STATE" VARCHAR2(40 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"VSM\_CITY" VARCHAR2(100 BYTE),

"VSM\_STATE" VARCHAR2(2 BYTE),

"VSM\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"XNG\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"XNG\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"SCHEMA" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_VSM\_XNG\_XREF\_BK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_VSM\_XNG\_XREF\_BK"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE),

"ALLTEL\_DO\_NOT\_REMOVE" VARCHAR2(1 BYTE),

"AS\_OF\_DATE" DATE,

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"CITY" VARCHAR2(40 BYTE),

"COUNTY" VARCHAR2(40 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"DOMAIN\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"MARKET" VARCHAR2(100 BYTE),

"MARKET\_INST\_ID" NUMBER,

"OLD\_SITE\_INST\_ID" NUMBER,

"SITE\_INST\_ID" NUMBER,

"SITE\_HUM\_ID" VARCHAR2(50 BYTE),

"STATE" VARCHAR2(40 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"VSM\_CITY" VARCHAR2(100 BYTE),

"VSM\_STATE" VARCHAR2(2 BYTE),

"VSM\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"XNG\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"XNG\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"SCHEMA" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_VSM\_XNG\_XREF\_FEB\_10\_2011

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_VSM\_XNG\_XREF\_FEB\_10\_2011"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE),

"ALLTEL\_DO\_NOT\_REMOVE" VARCHAR2(1 BYTE),

"AS\_OF\_DATE" DATE,

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"CITY" VARCHAR2(40 BYTE),

"COUNTY" VARCHAR2(40 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"DOMAIN\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"MARKET" VARCHAR2(100 BYTE),

"MARKET\_INST\_ID" NUMBER,

"OLD\_SITE\_INST\_ID" NUMBER,

"SITE\_INST\_ID" NUMBER,

"SITE\_HUM\_ID" VARCHAR2(50 BYTE),

"STATE" VARCHAR2(40 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"VSM\_CITY" VARCHAR2(100 BYTE),

"VSM\_STATE" VARCHAR2(2 BYTE),

"VSM\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"XNG\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"XNG\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"SCHEMA" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table VZW\_VSM\_XNG\_XREF\_FEB\_16\_2011

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."VZW\_VSM\_XNG\_XREF\_FEB\_16\_2011"

( "VSM\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VSM\_CLASS" VARCHAR2(100 BYTE),

"ALLTEL\_DO\_NOT\_REMOVE" VARCHAR2(1 BYTE),

"AS\_OF\_DATE" DATE,

"AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"CITY" VARCHAR2(40 BYTE),

"COUNTY" VARCHAR2(40 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"DOMAIN\_NAME" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"MARKET" VARCHAR2(100 BYTE),

"MARKET\_INST\_ID" NUMBER,

"OLD\_SITE\_INST\_ID" NUMBER,

"SITE\_INST\_ID" NUMBER,

"SITE\_HUM\_ID" VARCHAR2(50 BYTE),

"STATE" VARCHAR2(40 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"VSM\_CITY" VARCHAR2(100 BYTE),

"VSM\_STATE" VARCHAR2(2 BYTE),

"VSM\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"XNG\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"XNG\_MOD\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"SCHEMA" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WATCHDOG\_HIST

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WATCHDOG\_HIST"

( "PROCESS\_ID" NUMBER,

"PROCESS\_NAME" VARCHAR2(50 BYTE),

"PROCESS\_TYPE" VARCHAR2(1 BYTE),

"BASE\_EXEC\_TIME" NUMBER,

"START\_TIME" DATE,

"END\_TIME" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WD\_ERROR\_CODE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WD\_ERROR\_CODE"

( "ERROR\_CODE" VARCHAR2(50 BYTE),

"DESCRIPTION" VARCHAR2(1000 BYTE),

"ERROR\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WD\_ERROR\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WD\_ERROR\_DETAILS"

( "PROCESS\_ID" NUMBER,

"ERROR\_ID" NUMBER,

"ERROR\_MSG" VARCHAR2(2000 BYTE),

"LOG\_DATE" DATE DEFAULT sysdate,

"NOTIFY" CHAR(1 BYTE) DEFAULT 'N',

"EMAIL\_SENT" CHAR(1 BYTE) DEFAULT 'N',

"EMAIL\_SENT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_BY\_QUEUE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE"

( "WO\_INST\_ID" NUMBER(9,0),

"WO\_NAME" VARCHAR2(20 BYTE),

"DESCRIPTION" VARCHAR2(80 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"START\_AFTER" DATE,

"COMPLETE\_BY" DATE,

"ACTUAL\_COMPL" DATE,

"QUEUE\_INST\_ID" NUMBER(9,0),

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"TASK\_NAME" VARCHAR2(60 BYTE),

"TASK\_INST\_ID" NUMBER(9,0),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"ELEMENT\_CATEGORY" VARCHAR2(30 BYTE),

"TASK\_STATUS" VARCHAR2(20 BYTE),

"TASK\_START\_AFTER" DATE,

"TASK\_COMPLETE\_BY" DATE,

"DAYS\_IN\_QUEUE" NUMBER,

"DAYS\_AT\_STATUS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_BY\_QUEUE\_BACKUP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_BACKUP"

( "WO\_INST\_ID" NUMBER(9,0),

"WO\_NAME" VARCHAR2(20 BYTE),

"DESCRIPTION" VARCHAR2(80 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"START\_AFTER" DATE,

"COMPLETE\_BY" DATE,

"ACTUAL\_COMPL" DATE,

"QUEUE\_INST\_ID" NUMBER(9,0),

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"TASK\_NAME" VARCHAR2(60 BYTE),

"TASK\_INST\_ID" NUMBER(9,0),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"ELEMENT\_CATEGORY" VARCHAR2(30 BYTE),

"TASK\_STATUS" VARCHAR2(20 BYTE),

"TASK\_START\_AFTER" DATE,

"TASK\_COMPLETE\_BY" DATE,

"DAYS\_IN\_QUEUE" NUMBER,

"DAYS\_AT\_STATUS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_BY\_QUEUE\_TEMP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_TEMP"

( "WO\_INST\_ID" NUMBER(9,0),

"WO\_NAME" VARCHAR2(20 BYTE),

"DESCRIPTION" VARCHAR2(80 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"START\_AFTER" DATE,

"COMPLETE\_BY" DATE,

"ACTUAL\_COMPL" DATE,

"QUEUE\_INST\_ID" NUMBER(9,0),

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"TASK\_NAME" VARCHAR2(60 BYTE),

"TASK\_INST\_ID" NUMBER(9,0),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"ELEMENT\_CATEGORY" VARCHAR2(30 BYTE),

"TASK\_STATUS" VARCHAR2(20 BYTE),

"TASK\_START\_AFTER" DATE,

"TASK\_COMPLETE\_BY" DATE,

"DAYS\_IN\_QUEUE" NUMBER,

"DAYS\_AT\_STATUS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_BY\_Q\_TEMP\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_BY\_Q\_TEMP\_AUDIT"

( "ELEMENT\_TYPE" VARCHAR2(2 BYTE),

"INST\_ID" NUMBER(9,0),

"VERSION\_CHGD" NUMBER(6,0),

"EDIT\_OPERATION" CHAR(1 BYTE),

"CHG\_TS" DATE,

"CHG\_BY" VARCHAR2(20 BYTE),

"CHG\_DESC" VARCHAR2(200 BYTE),

"SUBCLASS\_TYPE" VARCHAR2(2 BYTE),

"SUBCLASS\_INST\_ID" NUMBER(9,0),

"SUBCLASS\_OPER" CHAR(1 BYTE),

"FIELD\_CHGD" VARCHAR2(550 BYTE),

"OLD\_VALUE" VARCHAR2(4000 BYTE),

"NEW\_VALUE" VARCHAR2(4000 BYTE),

"DAYS\_IN\_QUEUE" NUMBER,

"DAYS\_AT\_STATUS" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_TASK\_AGING\_DATES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_TASK\_AGING\_DATES"

( "WO\_INST\_ID" NUMBER(9,0),

"TASK\_INST\_ID" NUMBER(9,0),

"READY\_DATE" DATE,

"READY\_BY" VARCHAR2(20 BYTE),

"IN\_PROCESS\_DATE" DATE,

"IN\_PROCESS\_BY" VARCHAR2(20 BYTE),

"COMPLETED\_DATE" DATE,

"COMPLETED\_BY" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 201326592 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_TASK\_AGING\_DATES\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_TASK\_AGING\_DATES\_WRK"

( "INST\_ID" NUMBER(9,0),

"TASK\_ID" NUMBER(9,0),

"VERSION\_CHGD" NUMBER(6,0),

"STATUS" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_TASK\_AGING\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL"

( "MARKET" VARCHAR2(30 BYTE),

"WO\_NAME" VARCHAR2(20 BYTE),

"WO\_INST\_ID" NUMBER,

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"QUEUE\_INST\_ID" NUMBER,

"WO\_STATUS" CHAR(1 BYTE),

"TASK\_INST\_ID" NUMBER,

"TASK\_NAME" VARCHAR2(60 BYTE),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"ELEMENT\_CATEGORY" VARCHAR2(30 BYTE),

"ELEMENT\_NAME" VARCHAR2(300 BYTE),

"TASK\_OPERATION" VARCHAR2(20 BYTE),

"WTI\_STATUS\_CODE" CHAR(1 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"ORIGINATOR" VARCHAR2(20 BYTE),

"CHG\_TS" DATE,

"WO\_CREATE\_DATE" DATE,

"LOAD\_DATE" DATE,

"WO\_LOCALE" VARCHAR2(10 BYTE),

"QUEUE\_LOCALE" VARCHAR2(10 BYTE),

"TASK\_CHG\_BY" VARCHAR2(20 BYTE),

"PROJECT\_ID" VARCHAR2(200 BYTE),

"ACTUAL\_START" DATE,

"ACTUAL\_COMPL" DATE,

"READY\_DATE" DATE,

"READY\_DY" NUMBER,

"READY\_HR" NUMBER,

"READY\_MI" NUMBER,

"ACTUAL\_DY" NUMBER,

"ACTUAL\_HR" NUMBER,

"ACTUAL\_MI" NUMBER,

"TOTAL\_DY" NUMBER,

"TOTAL\_HR" NUMBER,

"TOTAL\_MI" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_TASK\_AGING\_DETAIL\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL\_AUDIT"

( "MARKET" VARCHAR2(30 BYTE),

"WO\_NAME" VARCHAR2(20 BYTE),

"WO\_INST\_ID" NUMBER,

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"QUEUE\_INST\_ID" NUMBER,

"WO\_STATUS" CHAR(1 BYTE),

"TASK\_INST\_ID" NUMBER,

"TASK\_NAME" VARCHAR2(60 BYTE),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"ELEMENT\_CATEGORY" VARCHAR2(30 BYTE),

"ELEMENT\_NAME" VARCHAR2(300 BYTE),

"TASK\_OPERATION" VARCHAR2(20 BYTE),

"WTI\_STATUS\_CODE" CHAR(1 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"ORIGINATOR" VARCHAR2(20 BYTE),

"CHG\_TS" DATE,

"WO\_CREATE\_DATE" DATE,

"LOAD\_DATE" DATE,

"WO\_LOCALE" VARCHAR2(10 BYTE),

"QUEUE\_LOCALE" VARCHAR2(10 BYTE),

"TASK\_CHG\_BY" VARCHAR2(20 BYTE),

"PROJECT\_ID" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_TASK\_AGING\_DETAIL\_BACKUP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL\_BACKUP"

( "MARKET" VARCHAR2(30 BYTE),

"WO\_NAME" VARCHAR2(20 BYTE),

"WO\_INST\_ID" NUMBER,

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"QUEUE\_INST\_ID" NUMBER,

"WO\_STATUS" CHAR(1 BYTE),

"TASK\_INST\_ID" NUMBER,

"TASK\_NAME" VARCHAR2(60 BYTE),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"ELEMENT\_CATEGORY" VARCHAR2(30 BYTE),

"ELEMENT\_NAME" VARCHAR2(300 BYTE),

"TASK\_OPERATION" VARCHAR2(20 BYTE),

"WTI\_STATUS\_CODE" CHAR(1 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"ORIGINATOR" VARCHAR2(20 BYTE),

"CHG\_TS" DATE,

"WO\_CREATE\_DATE" DATE,

"LOAD\_DATE" DATE,

"WO\_LOCALE" VARCHAR2(10 BYTE),

"QUEUE\_LOCALE" VARCHAR2(10 BYTE),

"TASK\_CHG\_BY" VARCHAR2(20 BYTE),

"PROJECT\_ID" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table WO\_TASK\_AGING\_DETAIL\_WRK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL\_WRK"

( "MARKET" VARCHAR2(30 BYTE),

"WO\_NAME" VARCHAR2(20 BYTE),

"WO\_INST\_ID" NUMBER,

"QUEUE\_NAME" VARCHAR2(30 BYTE),

"QUEUE\_INST\_ID" NUMBER,

"WO\_STATUS" CHAR(1 BYTE),

"TASK\_INST\_ID" NUMBER,

"TASK\_NAME" VARCHAR2(60 BYTE),

"ELEMENT\_TYPE" CHAR(1 BYTE),

"ELEMENT\_CATEGORY" VARCHAR2(30 BYTE),

"ELEMENT\_NAME" VARCHAR2(300 BYTE),

"TASK\_OPERATION" VARCHAR2(20 BYTE),

"WTI\_STATUS\_CODE" CHAR(1 BYTE),

"STATUS\_NAME" VARCHAR2(20 BYTE),

"ORIGINATOR" VARCHAR2(20 BYTE),

"CHG\_TS" DATE,

"WO\_CREATE\_DATE" DATE,

"LOAD\_DATE" DATE,

"WO\_LOCALE" VARCHAR2(10 BYTE),

"QUEUE\_LOCALE" VARCHAR2(10 BYTE),

"TASK\_CHG\_BY" VARCHAR2(20 BYTE),

"PROJECT\_ID" VARCHAR2(200 BYTE),

"ACTUAL\_START" DATE,

"ACTUAL\_COMPL" DATE,

"READY\_DATE" DATE,

"READY\_DY" NUMBER,

"READY\_HR" NUMBER,

"READY\_MI" NUMBER,

"ACTUAL\_DY" NUMBER,

"ACTUAL\_HR" NUMBER,

"ACTUAL\_MI" NUMBER,

"TOTAL\_DY" NUMBER,

"TOTAL\_HR" NUMBER,

"TOTAL\_MI" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 201326592 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XCONNECT\_SUMMARY

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XCONNECT\_SUMMARY"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"AUDITABLE" NUMBER,

"NOT\_AUDITABLE" NUMBER,

"TOTAL\_XCONN\_COUNT" NUMBER,

"MATCHED\_XCONN\_COUNT" NUMBER,

"COMPLIANCE\_PCT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XCONN\_AUDIT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XCONN\_AUDIT"

( "XNG\_SITE\_INST\_ID" NUMBER(9,0),

"XNG\_SITE\_NAME" VARCHAR2(50 BYTE),

"XNG\_EQUIP\_INST\_ID" NUMBER(9,0),

"XNG\_EQUIP\_CONTAINER" VARCHAR2(128 BYTE),

"XNG\_DEVICE\_TYPE" VARCHAR2(20 BYTE),

"XNG\_TARGET\_ID" VARCHAR2(30 BYTE),

"XNG\_DOMAIN\_INST\_ID" NUMBER(9,0),

"AUDIT\_STATUS" VARCHAR2(20 BYTE),

"AUDIT\_DATE" DATE,

"LAST\_SUCCESSFUL\_AUDIT\_DATE" DATE,

"PERCENT\_CORRECT\_IN\_XNG" VARCHAR2(10 BYTE),

"MATCHED\_XCONN\_COUNT" NUMBER(9,0),

"DISCREPANCY\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_XCONN\_COUNT" NUMBER(9,0),

"TOTAL\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"MATCHED\_CIRC\_PATH\_COUNT" NUMBER(9,0),

"PERCENT\_CORRECT\_FROM\_NETWORK" VARCHAR2(10 BYTE),

"DESCRIPTION" VARCHAR2(300 BYTE),

"NE\_CLLI" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(60 BYTE),

"STATUS" VARCHAR2(30 BYTE),

"METROWATCH\_DB" VARCHAR2(50 BYTE),

"METROWATCH\_NE\_NAME" VARCHAR2(40 BYTE),

"AREA" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XCONN\_AUDIT\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XCONN\_AUDIT\_DETAILS"

( "XNG\_EQUIP\_INST\_ID" NUMBER(9,0),

"AUDIT\_STATUS" VARCHAR2(15 BYTE),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"DEVICE\_XCONN\_ELEMENTS" VARCHAR2(50 BYTE),

"DETAILS\_STATUS" VARCHAR2(20 BYTE),

"MEMBER\_NBR" NUMBER(5,0),

"DISCREPANCY\_DETAILS" VARCHAR2(256 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XCONN\_CIRC\_PATH\_DETAILS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XCONN\_CIRC\_PATH\_DETAILS"

( "XNG\_EQUIP\_INST\_ID" NUMBER(9,0),

"XNG\_CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"XNG\_PATH\_NAME" VARCHAR2(150 BYTE),

"XNG\_PATH\_ELEMENTS" VARCHAR2(1200 BYTE),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"PREV\_PATH\_INST\_ID" NUMBER(9,0),

"PATH\_STATUS" VARCHAR2(20 BYTE),

"PATH\_DETAILS\_STATUS" VARCHAR2(20 BYTE),

"XNG\_CONTAINER\_ID" NUMBER(9,0),

"MEMBER\_NBR" NUMBER(5,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XMS\_ENB\_DEL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XMS\_ENB\_DEL"

( "NODE" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" VARCHAR2(50 BYTE),

"SERVER\_NAME" VARCHAR2(100 BYTE),

"ENB\_STATUS" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_AUDIT\_PROPERTIES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_AUDIT\_PROPERTIES"

( "NAME" VARCHAR2(100 BYTE),

"VALUE" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_AUDIT\_SUMMARY\_REPORT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_AUDIT\_SUMMARY\_REPORT"

( "AREA" VARCHAR2(100 BYTE),

"REGION" VARCHAR2(100 BYTE),

"SITE\_FAULT\_PCT" FLOAT(126),

"PATH\_FAULT\_PCT" FLOAT(126),

"EQUIP\_FAULT\_PCT" FLOAT(126),

"OVERALL\_FAULT\_PCT" FLOAT(126),

"CREATION\_DATE" DATE DEFAULT SYSDATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_DOMAIN\_REGION

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_DOMAIN\_REGION"

( "DOMAIN\_NAME" VARCHAR2(30 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_EQUIP\_CNT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_EQUIP\_CNT"

( "VW\_TYPE" VARCHAR2(20 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"TOTAL\_REC" NUMBER,

"TOTAL\_FAULT\_REC" NUMBER,

"TOTAL\_FAULT\_CNT" NUMBER,

"DESCR\_FAULT\_CNT" NUMBER,

"EQ\_CLASS\_FAULT\_CNT" NUMBER,

"CREATION\_DATE" DATE DEFAULT SYSDATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_EQUIP\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_EQUIP\_DETAIL"

( "VW\_TYPE" VARCHAR2(20 BYTE),

"DOMAIN\_NAME" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER,

"SITE\_INST\_ID" NUMBER,

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SITE\_STATE" VARCHAR2(40 BYTE),

"CATEGORY" VARCHAR2(30 BYTE),

"DESCR\_FAULT" NUMBER,

"EQ\_CLASS\_FAULT" NUMBER,

"FAULT\_CNT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_PATH\_CNT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_PATH\_CNT"

( "VW\_TYPE" VARCHAR2(20 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"PATH\_NUM\_ID\_FAULT\_CNT" NUMBER,

"ASIDE\_FAULT\_CNT" NUMBER,

"ZSIDE\_FAULT\_CNT" NUMBER,

"DOM\_IP\_FAULT\_CNT" NUMBER,

"TOTAL\_FAULT\_REC" NUMBER,

"TOTAL\_FAULT\_CNT" NUMBER,

"TOTAL\_REC" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_PATH\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_PATH\_DETAIL"

( "VW\_TYPE" VARCHAR2(20 BYTE),

"DOMAIN\_NAME" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_PATH\_HUM\_ID" VARCHAR2(200 BYTE),

"CIRC\_PATH\_REV\_NBR" NUMBER,

"CATEGORY" VARCHAR2(50 BYTE),

"DOMAIN\_INST\_ID" NUMBER,

"BANDWIDTH" VARCHAR2(50 BYTE),

"STATUS" VARCHAR2(50 BYTE),

"A\_SIDE\_SITE\_ID" VARCHAR2(50 BYTE),

"Z\_SIDE\_SITE\_ID" VARCHAR2(50 BYTE),

"NBR\_CHANNELS" NUMBER,

"NBR\_CHAN\_ASSIGNED" NUMBER,

"TOPOLOGY" VARCHAR2(5 BYTE),

"IS\_TRUNK\_GROUP" VARCHAR2(5 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER,

"A\_SIDE\_SITE\_STATE" VARCHAR2(5 BYTE),

"DOM\_IP" VARCHAR2(50 BYTE),

"PATH\_NUM\_ID\_FAULT" NUMBER,

"ASIDE\_FAULT" NUMBER,

"ZSIDE\_FAULT" NUMBER,

"DOM\_IP\_FAULT" NUMBER,

"TOTAL\_FAULT" NUMBER,

"SUBCATEGORY" VARCHAR2(50 BYTE),

"PATTERN" VARCHAR2(200 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_SEGMENT\_CNT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SEGMENT\_CNT"

( "VW\_TYPE" VARCHAR2(20 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"PON\_FAULT\_CNT" NUMBER,

"ORDER\_DATE\_FAULT\_CNT" NUMBER,

"ORDER\_NUMBER\_FAULT\_CNT" NUMBER,

"DATE\_FAULT\_CNT" NUMBER,

"TOTAL\_FAULT\_REC" NUMBER,

"TOTAL\_FAULT\_CNT" NUMBER,

"TOTAL\_REC" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_SEGMENT\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SEGMENT\_DETAIL"

( "VW\_TYPE" VARCHAR2(20 BYTE),

"DOMAIN\_NAME" VARCHAR2(50 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"CIRC\_HUM\_ID" VARCHAR2(200 BYTE),

"ASIDESITESTATE" VARCHAR2(20 BYTE),

"CIRC\_INST\_ID" NUMBER,

"XNG\_TYPE" VARCHAR2(30 BYTE),

"BILL\_DATE\_MONTH" VARCHAR2(20 BYTE),

"BILL\_DATE\_YEAR" VARCHAR2(20 BYTE),

"DISCONNECT\_PON" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_DATE" VARCHAR2(3100 BYTE),

"DISCONNECT\_ORDER\_NUMBER" VARCHAR2(3100 BYTE),

"DISCONNECT\_DATE" VARCHAR2(3100 BYTE),

"PON\_FAULT" NUMBER,

"ORDER\_DATE\_FAULT" NUMBER,

"ORDER\_NUMBER\_FAULT" NUMBER,

"DATE\_FAULT" NUMBER,

"TOTAL\_FAULT" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_SITE\_CNT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SITE\_CNT"

( "VW\_TYPE" VARCHAR2(20 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"NAME\_NULL\_CNT" NUMBER,

"SITEID\_NULL\_CNT" NUMBER,

"LATITUDE\_NULL\_CNT" NUMBER,

"LONGITUDE\_NULL\_CNT" NUMBER,

"STREET\_NULL\_CNT" NUMBER,

"CITY\_NULL\_CNT" NUMBER,

"STATE\_NULL\_CNT" NUMBER,

"POSTALCODE\_NULL\_CNT" NUMBER,

"FLOOR\_NULL\_CNT" NUMBER,

"COUNTY\_NULL\_CNT" NUMBER,

"COUNTRY\_NULL\_CNT" NUMBER,

"DIRECTIONS\_NULL\_CNT" NUMBER,

"FIREPHONE\_FAULT\_CNT" NUMBER,

"POLICE\_FAULT\_CNT" NUMBER,

"POWERPHONE\_FAULT\_CNT" NUMBER,

"POWERACCT\_NULL\_CNT" NUMBER,

"POWERCOMPANY\_NULL\_CNT" NUMBER,

"TOTAL\_FAULT\_REC" NUMBER,

"TOTAL\_FAULT\_CNT" NUMBER,

"TOTAL\_REC" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_SITE\_DETAIL

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SITE\_DETAIL"

( "VW\_TYPE" VARCHAR2(20 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"SITE\_INST\_ID" NUMBER,

"SITE\_NAME" VARCHAR2(100 BYTE),

"DOMAIN\_NAME" VARCHAR2(30 BYTE),

"CATEGORY" VARCHAR2(30 BYTE),

"STATE" VARCHAR2(40 BYTE),

"NAME\_NULL" NUMBER,

"SITEID\_NULL" NUMBER,

"LATITUDE\_NULL" NUMBER,

"LONGITUDE\_NULL" NUMBER,

"STREET\_NULL" NUMBER,

"CITY\_NULL" NUMBER,

"STATE\_NULL" NUMBER,

"POSTALCODE\_NULL" NUMBER,

"FLOOR\_NULL" NUMBER,

"COUNTY\_NULL" NUMBER,

"COUNTRY\_NULL" NUMBER,

"DIRECTIONS\_NULL" NUMBER,

"FIREPHONE\_FAULT" NUMBER,

"POLICEPHONE\_FAULT" NUMBER,

"POWERPHONE\_FAULT" NUMBER,

"POWERACCT\_NULL" NUMBER,

"POWERCOMPANY\_NULL" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_SUM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SUM"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"SITE\_FAULT\_PCT" FLOAT(126),

"EQUIP\_FAULT\_PCT" FLOAT(126),

"PATH\_FAULT\_PCT" FLOAT(126),

"SEG\_FAULT\_PCT" FLOAT(126),

"OVERALL\_FAULT\_PCT" FLOAT(126),

"CREATION\_DATE" DATE DEFAULT SYSDATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_FAULT\_SUM\_BK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SUM\_BK"

( "AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"SITE\_FAULT\_PCT" FLOAT(126),

"EQUIP\_FAULT\_PCT" FLOAT(126),

"PATH\_FAULT\_PCT" FLOAT(126),

"SEG\_FAULT\_PCT" FLOAT(126),

"OVERALL\_FAULT\_PCT" FLOAT(126),

"CREATION\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNGSTD\_SWITCH\_VENDORS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNGSTD\_SWITCH\_VENDORS"

( "VENDOR" VARCHAR2(20 BYTE),

"SWITCH\_ID" VARCHAR2(20 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_ADM\_FW\_EQUIP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_ADM\_FW\_EQUIP"

( "NE\_INST\_ID" NUMBER,

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS\_TYPE" VARCHAR2(5 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(30 CHAR),

"TYPE" VARCHAR2(30 CHAR),

"STATUS" VARCHAR2(20 CHAR),

"VENDOR" VARCHAR2(40 CHAR),

"EQ\_CLASS" CHAR(1 CHAR),

"PARSE\_STATUS" VARCHAR2(250 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_CONTACTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_CONTACTS"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"SITE\_TYPE" VARCHAR2(30 BYTE),

"ADDRESS" VARCHAR2(4000 BYTE),

"CITY" VARCHAR2(60 BYTE),

"STATE\_PROV" VARCHAR2(40 BYTE),

"POST\_CODE\_1" VARCHAR2(10 BYTE),

"COUNTY" VARCHAR2(40 BYTE),

"LATITUDE" VARCHAR2(20 BYTE),

"LONGITUDE" VARCHAR2(20 BYTE),

"MGR\_NAME" VARCHAR2(255 BYTE),

"MGR\_PH" VARCHAR2(3100 BYTE),

"TECH\_NAME" VARCHAR2(255 BYTE),

"TECH\_PH" VARCHAR2(3100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_CSR\_BTS\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_CSR\_BTS\_PATHS"

( "MATCHED\_EQUIP\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"CSR\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"PATH\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_CSR\_BTS\_PATHS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_CSR\_BTS\_PATHS\_WK"

( "MATCHED\_EQUIP\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"CSR\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"PATH\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_CSR\_GIGE\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_CSR\_GIGE\_PATHS"

( "MATCHED\_EQUIP\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER,

"PORT\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"CSR\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"GIGE\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_CSR\_GIGE\_PATHS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_CSR\_GIGE\_PATHS\_WK"

( "MATCHED\_EQUIP\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER,

"PORT\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"CSR\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"GIGE\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_CSR\_PARSED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_CSR\_PARSED"

( "EQUIP\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_CSR\_PARSED\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_CSR\_PARSED\_WK"

( "EQUIP\_INST\_ID" NUMBER,

"DESCR" VARCHAR2(100 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"CSR\_DEVICE\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_CSR\_VLAN\_PATHS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_CSR\_VLAN\_PATHS\_WK"

( "EQUIP\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_BANDWIDTH" VARCHAR2(30 BYTE),

"PATH\_TYPE" VARCHAR2(30 BYTE),

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_HUM\_ID" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate,

"CARD\_INST\_ID" NUMBER,

"CARD\_TYPE" VARCHAR2(50 BYTE),

"TEST\_HEAD\_NAME" VARCHAR2(50 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_ENB\_PARSED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_ENB\_PARSED"

( "SITE\_HUM\_ID" VARCHAR2(100 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"ENB\_NUMBER" VARCHAR2(6 BYTE),

"EQ\_CLASS" CHAR(1 BYTE),

"AREA" VARCHAR2(50 BYTE),

"REGION" VARCHAR2(50 BYTE),

"LEAF\_DOMAIN\_NAME" VARCHAR2(30 BYTE),

"SITE\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"LEAF\_DOMAIN\_INST\_ID" NUMBER(9,0),

"PARSE\_STATUS" VARCHAR2(50 BYTE),

"HAS\_MULTIPLES" VARCHAR2(1 BYTE),

"VENDOR" VARCHAR2(30 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"TYPE" VARCHAR2(30 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_FW\_EQUIP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_FW\_EQUIP"

( "NE\_INST\_ID" NUMBER,

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS\_TYPE" VARCHAR2(5 CHAR),

"DESCR" VARCHAR2(100 CHAR),

"MODEL" VARCHAR2(30 CHAR),

"TYPE" VARCHAR2(30 CHAR),

"STATUS" VARCHAR2(20 CHAR),

"VENDOR" VARCHAR2(40 CHAR),

"EQ\_CLASS" CHAR(1 CHAR),

"PARSE\_STATUS" VARCHAR2(250 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 1048576 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_GIGE\_DELIVERY\_TABLE

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_GIGE\_DELIVERY\_TABLE"

( "GIGE\_PATH\_INST\_ID" NUMBER,

"SEGMENT\_INST\_ID" NUMBER(9,0),

"DELIVERED\_ON" VARCHAR2(100 BYTE),

"SEGMENT\_SEQ\_NBR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_GIGE\_DELIVERY\_TABLE\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_GIGE\_DELIVERY\_TABLE\_WK"

( "GIGE\_PATH\_INST\_ID" NUMBER,

"SEGMENT\_INST\_ID" NUMBER(9,0),

"DELIVERED\_ON" VARCHAR2(100 BYTE),

"SEGMENT\_SEQ\_NBR" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_GIGE\_PARENTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_GIGE\_PARENTS"

( "GIGE\_INST\_ID" NUMBER(9,0),

"LEG\_INST\_ID" NUMBER(9,0),

"MIN\_REL\_ORDER" NUMBER(9,0),

"PARENT\_PATH\_INST\_ID" NUMBER(9,0),

"SYS\_CONN\_BY\_PATH" VARCHAR2(1000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_GIGE\_PARENTS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_GIGE\_PARENTS\_WK"

( "GIGE\_INST\_ID" NUMBER(9,0),

"LEG\_INST\_ID" NUMBER(9,0),

"MIN\_REL\_ORDER" NUMBER(9,0),

"PARENT\_PATH\_INST\_ID" NUMBER(9,0),

"SYS\_CONN\_BY\_PATH" VARCHAR2(1000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_GIGE\_PARENT\_MW

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_GIGE\_PARENT\_MW"

( "GIGE\_PATH\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_GIGE\_PARENT\_MW\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_GIGE\_PARENT\_MW\_WK"

( "GIGE\_PATH\_INST\_ID" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_GIGE\_VLAN\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_GIGE\_VLAN\_MAP"

( "GIGE\_PATH\_INST\_ID" NUMBER(9,0),

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"VLAN\_NUMBER" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_GIGE\_VLAN\_MAP\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_GIGE\_VLAN\_MAP\_WK"

( "GIGE\_PATH\_INST\_ID" NUMBER(10,0),

"VLAN\_INST\_ID" NUMBER(10,0),

"VLAN\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"VLAN\_NUMBER" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_JOB\_STATUS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_JOB\_STATUS"

( "JOB\_ID" VARCHAR2(30 BYTE),

"LAST\_COMPLETED\_ON" DATE,

"RUN\_TIME\_IN\_MINUTES" NUMBER DEFAULT 0,

"SUCCESS" NUMBER(\*,0),

"MESSAGE" VARCHAR2(255 BYTE),

"DATE\_RANGE\_BEGIN" DATE,

"DATE\_RANGE\_END" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_MPLS\_RTR\_PORTS\_PARSED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_MPLS\_RTR\_PORTS\_PARSED"

( "ROUTER\_NAME" VARCHAR2(11 BYTE),

"PORT\_INST\_ID" NUMBER(9,0),

"PORT\_HUM\_ID" VARCHAR2(50 BYTE),

"PORT\_ACCESS\_ID" VARCHAR2(100 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"P1" VARCHAR2(50 BYTE),

"P2" VARCHAR2(50 BYTE),

"P3" VARCHAR2(50 BYTE),

"P4" VARCHAR2(50 BYTE),

"P5" VARCHAR2(50 BYTE),

"P6" VARCHAR2(50 BYTE),

"PX" VARCHAR2(50 BYTE),

"PORT\_BW" VARCHAR2(30 BYTE),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"PATH\_LIVE\_IN\_XNG" CHAR(1 BYTE),

"EXTRACT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_MSPP\_DEVICES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES"

( "EQ\_CLASS\_TYPE" VARCHAR2(5 BYTE),

"EQ\_CLASS" CHAR(1 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"ACCESS\_ID" VARCHAR2(40 BYTE),

"NE\_ACCESS\_ID" VARCHAR2(4000 BYTE),

"INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"NE\_INST\_ID" NUMBER,

"VENDOR" VARCHAR2(40 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"SITE\_INST\_ID" NUMBER(9,0),

"ISSUE\_CODE" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_MSPP\_DEVICES\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES\_WK"

( "EQ\_CLASS\_TYPE" VARCHAR2(5 BYTE),

"EQ\_CLASS" CHAR(1 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"ACCESS\_ID" VARCHAR2(40 BYTE),

"NE\_ACCESS\_ID" VARCHAR2(4000 BYTE),

"INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"NE\_INST\_ID" NUMBER,

"VENDOR" VARCHAR2(40 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"CLLI" VARCHAR2(50 BYTE),

"SITE\_INST\_ID" NUMBER(9,0),

"ISSUE\_CODE" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_MWR\_PARSED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_MWR\_PARSED"

( "EQUIP\_INST\_ID" NUMBER(10,0),

"DESCR" VARCHAR2(100 BYTE),

"MWR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VENDOR" VARCHAR2(40 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE)

) SEGMENT CREATION DEFERRED

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE( INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_MWR\_PARSED\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_MWR\_PARSED\_WK"

( "EQUIP\_INST\_ID" NUMBER(10,0),

"DESCR" VARCHAR2(100 BYTE),

"MWR\_DEVICE\_NAME" VARCHAR2(100 BYTE),

"VENDOR" VARCHAR2(40 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_MWR\_PATHS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_MWR\_PATHS"

( "EQUIP\_INST\_ID" NUMBER,

"PORT\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"MWR\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"PATH\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"IS\_GIGE" VARCHAR2(1 BYTE)

) SEGMENT CREATION DEFERRED

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE( INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_MWR\_PATHS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_MWR\_PATHS\_WK"

( "EQUIP\_INST\_ID" NUMBER,

"PORT\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"NEXT\_PATH\_INST\_ID" NUMBER(9,0),

"MWR\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"PATH\_LIVE\_IN\_XNG" VARCHAR2(1 BYTE),

"IS\_GIGE" VARCHAR2(1 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_NEWS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_NEWS"

( "ID" NUMBER,

"TITLE" VARCHAR2(100 BYTE),

"PUB\_DATE" DATE,

"EXP\_DATE" DATE,

"CONTENT" VARCHAR2(4000 BYTE),

"PRIORITY" NUMBER,

"INSERT\_DATE" DATE

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_NGMLS\_DEVICES

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_NGMLS\_DEVICES"

( "EQUIP\_INST\_ID" NUMBER(9,0),

"ISSUE\_CODE" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_NGMLS\_DEVICES\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_NGMLS\_DEVICES\_WK"

( "EQUIP\_INST\_ID" NUMBER(9,0),

"ISSUE\_CODE" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_NGMLS\_EQUIP\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK"

( "NE\_INST\_ID" NUMBER,

"EQUIP\_INST\_ID" NUMBER(9,0),

"SITE\_INST\_ID" NUMBER(9,0),

"PARENT\_EQ\_INST\_ID" NUMBER(9,0),

"EQ\_CLASS\_TYPE" VARCHAR2(5 BYTE),

"DESCR" VARCHAR2(100 BYTE),

"MODEL" VARCHAR2(30 BYTE),

"TYPE" VARCHAR2(30 BYTE),

"STATUS" VARCHAR2(20 BYTE),

"VENDOR" VARCHAR2(40 BYTE),

"EQ\_CLASS" CHAR(1 BYTE),

"PARSE\_STATUS" VARCHAR2(250 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate,

"NGMLS\_VENDOR" VARCHAR2(10 BYTE),

"NGMLS\_DEVICE\_NAME" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_NGMLS\_VLAN\_PATHS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_NGMLS\_VLAN\_PATHS\_WK"

( "NE\_INST\_ID" NUMBER,

"EQUIP\_INST\_ID" NUMBER(9,0),

"CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_BANDWIDTH" VARCHAR2(30 BYTE),

"PATH\_TYPE" VARCHAR2(30 BYTE),

"VLAN\_INST\_ID" NUMBER(9,0),

"VLAN\_HUM\_ID" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"VLAN\_STATUS" VARCHAR2(50 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate,

"CARD\_INST\_ID" NUMBER,

"CARD\_TYPE" VARCHAR2(50 BYTE),

"TEST\_HEAD\_NAME" VARCHAR2(50 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_SS7\_PATHS\_PARSED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_SS7\_PATHS\_PARSED"

( "CIRC\_PATH\_INST\_ID" NUMBER,

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"PATH\_TYPE" VARCHAR2(50 BYTE),

"LINK\_1" VARCHAR2(20 BYTE),

"A\_CLLI" VARCHAR2(15 BYTE),

"LINK\_2" VARCHAR2(20 BYTE),

"Z\_CLLI" VARCHAR2(15 BYTE),

"EQUIP\_TYPE" VARCHAR2(50 BYTE),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"EXTRACT\_DATE" DATE DEFAULT sysdate

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_SUMMIT\_ANT\_EXTRACT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_DESCRIPTION" VARCHAR2(150 BYTE),

"EQUIP\_INST\_ID" NUMBER(9,0),

"ANTENNA\_ID" VARCHAR2(250 BYTE),

"NAURU\_ID" VARCHAR2(25 BYTE),

"TOWER\_OWNER" VARCHAR2(4000 BYTE),

"TOWER\_OWNER\_CONTACT\_INFO" VARCHAR2(4000 BYTE),

"NEAREST\_AIRPORT\_DIRECTION" VARCHAR2(4000 BYTE),

"NEAREST\_AIRPORT\_DISTANCE" VARCHAR2(4000 BYTE),

"NEAREST\_AIRPORT\_NAME" VARCHAR2(4000 BYTE),

"NEAREST\_AIRPORT\_CONTACT\_PHONE" VARCHAR2(4000 BYTE),

"ANTENNA\_SUPPORT\_FINISH" VARCHAR2(4000 BYTE),

"ANTENNA\_SUPPORT\_MAINTAINED\_BY" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_SUMMIT\_EXTRACT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_EXTRACT"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_DESCRIPTION" VARCHAR2(4000 BYTE),

"NAURU" VARCHAR2(4000 BYTE),

"ACCESS\_RESTRICTIONS" VARCHAR2(4000 BYTE),

"ACCESS\_WINDOW" VARCHAR2(4000 BYTE),

"MSC\_CSR\_PORT" VARCHAR2(4000 BYTE),

"CSR\_IP" VARCHAR2(4000 BYTE),

"CSR\_IPV6" VARCHAR2(4000 BYTE),

"CSR\_IPV4" VARCHAR2(4000 BYTE),

"CSR\_MAC\_ADDRESS" VARCHAR2(4000 BYTE),

"SITE\_BANDWIDTH" VARCHAR2(4000 BYTE),

"CIRCUIT\_ID" CLOB,

"DACS\_CROSSCONNECT" VARCHAR2(4000 BYTE),

"DRIVE\_DIRECTIONS" VARCHAR2(4000 BYTE),

"LOCAL\_FIRE\_PHONE" VARCHAR2(4000 BYTE),

"LOCAL\_POLICE\_PHONE" VARCHAR2(4000 BYTE),

"SITE\_ESCORT\_REQ" VARCHAR2(4000 BYTE),

"ENV\_ALARM\_TEST\_DATE" VARCHAR2(4000 BYTE),

"HVAC\_CONTACT" VARCHAR2(4000 BYTE),

"HVAC\_MAINT\_VENDOR" VARCHAR2(4000 BYTE),

"MICROWAVE\_LINK\_COUNT" VARCHAR2(4000 BYTE),

"NIU\_LOCATION" VARCHAR2(4000 BYTE),

"POWER\_ACCOUNT\_NUMBER" VARCHAR2(4000 BYTE),

"POWER\_PHONE" VARCHAR2(4000 BYTE),

"POWER\_COMPANY" VARCHAR2(4000 BYTE),

"POWER\_METER\_NUM" VARCHAR2(4000 BYTE),

"SITE\_TYPE" VARCHAR2(4000 BYTE),

"TELCO\_CONTACT\_PHONE" CLOB,

"TELCO\_MW" VARCHAR2(4000 BYTE),

"TELCO\_PROVIDER" CLOB,

"TELCO\_PUNCHDOWN\_BLOCK\_LOC" VARCHAR2(4000 BYTE),

"FIRE\_SUPRESSION\_TYPE" VARCHAR2(4000 BYTE),

"XNG\_PATH\_HYPERLINK" VARCHAR2(4000 BYTE),

"EXT\_TELCO\_NIU\_FLAG" VARCHAR2(4000 BYTE),

"GAS\_COMPANY" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_ACCT\_NUM" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_METER\_NUM" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_PHONE" VARCHAR2(4000 BYTE),

"HIGH\_VOLTAGE\_PROTECTION" VARCHAR2(4000 BYTE),

"POTS\_COMPANY" VARCHAR2(4000 BYTE),

"POTS\_COMPANY\_ACCT\_NUM" VARCHAR2(4000 BYTE),

"POTS\_PHONE\_NUM" VARCHAR2(4000 BYTE),

"AIRPORT\_DIRECTION" VARCHAR2(4000 BYTE),

"AIRPORT\_DISTANCE" VARCHAR2(4000 BYTE),

"AIRPORT\_NAME" VARCHAR2(4000 BYTE),

"AIRPORT\_CONTACT\_NAME" VARCHAR2(4000 BYTE),

"AIRPORT\_CONTACT\_PHONE" VARCHAR2(4000 BYTE),

"ANTENNA\_SUPPORT\_FINISH" VARCHAR2(4000 BYTE),

"ANTENNA\_SUPPORT\_MAINT\_BY" VARCHAR2(4000 BYTE),

"ACCESS\_ESCORT\_REQ\_FOR\_VENDOR" VARCHAR2(4000 BYTE),

"ACCESS\_INSTRUCTIONS" VARCHAR2(4000 BYTE),

"BACKHAUL\_TYPE" VARCHAR2(4000 BYTE),

"ENODEB\_IPV4" VARCHAR2(4000 BYTE),

"ENODEB\_IPV6" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00"

LOB ("CIRCUIT\_ID") STORE AS BASICFILE (

TABLESPACE "SS\_DATA00" ENABLE STORAGE IN ROW CHUNK 16384 RETENTION

NOCACHE LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT))

LOB ("TELCO\_CONTACT\_PHONE") STORE AS BASICFILE (

TABLESPACE "SS\_DATA00" ENABLE STORAGE IN ROW CHUNK 16384 RETENTION

NOCACHE LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT))

LOB ("TELCO\_PROVIDER") STORE AS BASICFILE (

TABLESPACE "SS\_DATA00" ENABLE STORAGE IN ROW CHUNK 16384 RETENTION

NOCACHE LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)) ;

--------------------------------------------------------

-- DDL for Table XNG\_SUMMIT\_SEG\_EXTRACT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_DESCRIPTION" VARCHAR2(150 BYTE),

"CIRC\_INST\_ID" NUMBER(9,0),

"NAURU\_ID" VARCHAR2(25 BYTE),

"CIRCUIT\_ID" VARCHAR2(150 BYTE),

"SEGMENT\_TYPE" VARCHAR2(50 BYTE),

"SEGMENT\_BANDWIDTH" VARCHAR2(50 BYTE),

"SEGMENT\_BPS" NUMBER(18,0),

"TELCO\_PROVIDER" VARCHAR2(4000 BYTE),

"TELCO\_VENDOR\_COMMENTS" VARCHAR2(4000 BYTE),

"TELCO\_VENDOR\_TECHNICAL\_CONTACT" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_SUMMIT\_SITE\_EXTRACT

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT"

( "SITE\_INST\_ID" NUMBER(9,0),

"SITE\_DESCRIPTION" VARCHAR2(150 BYTE),

"NAURU\_ID" VARCHAR2(25 BYTE),

"ACCESS\_RESTRICTIONS" VARCHAR2(4000 BYTE),

"MSC\_CSR\_PORT" VARCHAR2(4000 BYTE),

"CSR\_MAC\_ADDRESS" VARCHAR2(4000 BYTE),

"DRIVE\_DIRECTIONS" VARCHAR2(4000 BYTE),

"LOCAL\_FIRE\_PHONE" VARCHAR2(4000 BYTE),

"LOCAL\_FIRE\_PHONE\_COMMENTS" VARCHAR2(4000 BYTE),

"LOCAL\_POLICE\_PHONE" VARCHAR2(4000 BYTE),

"LOCAL\_POLICE\_PHONE\_COMMENTS" VARCHAR2(4000 BYTE),

"ENV\_ALARM\_TEST\_DATE" VARCHAR2(4000 BYTE),

"HVAC\_CONTACT\_PHONE" VARCHAR2(4000 BYTE),

"HVAC\_MAINT\_VENDOR" VARCHAR2(4000 BYTE),

"POWER\_ACCOUNT\_NUMBER" VARCHAR2(4000 BYTE),

"POWER\_PHONE" VARCHAR2(4000 BYTE),

"POWER\_COMPANY" VARCHAR2(4000 BYTE),

"POWER\_METER\_NUM" VARCHAR2(4000 BYTE),

"BUILDING\_TYPE" VARCHAR2(4000 BYTE),

"FIRE\_SUPRESSION\_TYPE" VARCHAR2(4000 BYTE),

"XNG\_PATH\_HYPERLINK" VARCHAR2(4000 BYTE),

"EXTERNAL\_TELCO\_NIU\_FLAG" VARCHAR2(4000 BYTE),

"GAS\_COMPANY" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_ACCT\_NUM" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_METER\_NUM" VARCHAR2(4000 BYTE),

"GAS\_COMPANY\_PHONE" VARCHAR2(4000 BYTE),

"HIGH\_VOLTAGE\_PROTECTION" VARCHAR2(4000 BYTE),

"POTS\_COMPANY" VARCHAR2(4000 BYTE),

"POSTS\_COMPANY\_ACCT\_NUM" VARCHAR2(4000 BYTE),

"POTS\_PHONE\_NUM" VARCHAR2(4000 BYTE),

"POTS\_COMPANY\_PHONE\_NUMBER" VARCHAR2(4000 BYTE),

"ENODEB\_IPV4" VARCHAR2(4000 BYTE),

"ENODEB\_IPV6" VARCHAR2(4000 BYTE),

"COPPER\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"FIBER\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"MICROWAVE\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"EBH\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"T1\_DEMARC\_LOCATION" VARCHAR2(4000 BYTE),

"XNG\_COMMENTS" VARCHAR2(4000 BYTE),

"CSR\_DEVICE\_IP" VARCHAR2(4000 BYTE),

"CSR\_SYSTEM\_IP" VARCHAR2(4000 BYTE),

"CSR\_MGMT\_IP" VARCHAR2(4000 BYTE),

"CSR\_LTE\_LOOPBACK\_IPV4" VARCHAR2(4000 BYTE),

"CSR\_LTE\_LOOPBACK\_IPV6" VARCHAR2(4000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_MSPP\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_MAP"

( "CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"NE\_INST\_ID" NUMBER,

"CARD\_INST\_ID" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"CARD\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_MSPP\_MAP\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_MAP\_WK"

( "CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"NE\_INST\_ID" NUMBER,

"CARD\_INST\_ID" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"CARD\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_MSPP\_TAM

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_TAM"

( "VLAN\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"CARD\_INST\_ID" NUMBER,

"PARENT\_PATH\_INST\_ID" NUMBER(9,0),

"TEST\_HEAD\_ON\_CARD" VARCHAR2(100 BYTE),

"TEST\_HEAD\_ON\_EQ" VARCHAR2(100 BYTE),

"PORT\_HUM\_ID" VARCHAR2(50 BYTE),

"SLOT" VARCHAR2(30 BYTE),

"VENDOR" VARCHAR2(50 CHAR),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"TAM\_STRING" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_MSPP\_TAM\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_TAM\_WK"

( "VLAN\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"CARD\_INST\_ID" NUMBER,

"PARENT\_PATH\_INST\_ID" NUMBER(9,0),

"TEST\_HEAD\_ON\_CARD" VARCHAR2(100 BYTE),

"TEST\_HEAD\_ON\_EQ" VARCHAR2(100 BYTE),

"PORT\_HUM\_ID" VARCHAR2(50 BYTE),

"SLOT" VARCHAR2(30 BYTE),

"VENDOR" VARCHAR2(50 CHAR),

"PARSE\_STATUS" VARCHAR2(100 BYTE),

"TAM\_STRING" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_NGMLS\_MAP

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_NGMLS\_MAP"

( "CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"CARD\_INST\_ID" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"CARD\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_NGMLS\_MAP\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_NGMLS\_MAP\_WK"

( "CIRC\_PATH\_INST\_ID" NUMBER(9,0),

"PORT\_INST\_ID" NUMBER(9,0),

"EQUIP\_INST\_ID" NUMBER(9,0),

"CARD\_INST\_ID" NUMBER,

"TEST\_HEAD\_NAME" VARCHAR2(100 BYTE),

"CARD\_TYPE" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_PARENTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_PARENTS"

( "VLAN\_INST\_ID" NUMBER(9,0),

"LEG\_INST\_ID" NUMBER(9,0),

"MIN\_REL\_ORDER" NUMBER(9,0),

"PARENT\_PATH\_INST\_ID" NUMBER(9,0),

"TERMINATES\_ON\_MTSO" NUMBER,

"HAS\_MSPP" NUMBER,

"HAS\_NGMLS" NUMBER,

"SPOKE\_TO\_HUB" NUMBER,

"SYS\_CONN\_BY\_PATH" VARCHAR2(1000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_PARENTS\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_PARENTS\_WK"

( "VLAN\_INST\_ID" NUMBER(9,0),

"LEG\_INST\_ID" NUMBER(9,0),

"MIN\_REL\_ORDER" NUMBER(9,0),

"PARENT\_PATH\_INST\_ID" NUMBER(9,0),

"TERMINATES\_ON\_MTSO" NUMBER,

"HAS\_MSPP" NUMBER,

"HAS\_NGMLS" NUMBER,

"SPOKE\_TO\_HUB" NUMBER,

"SYS\_CONN\_BY\_PATH" VARCHAR2(1000 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_PARSED

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_PARSED"

( "CIRC\_PATH\_INST\_ID" NUMBER,

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"PARSE\_STATUS" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XNG\_VLAN\_PARSED\_WK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XNG\_VLAN\_PARSED\_WK"

( "CIRC\_PATH\_INST\_ID" NUMBER,

"CIRC\_PATH\_HUM\_ID" VARCHAR2(100 BYTE),

"VLAN\_NUMBER" NUMBER,

"PARSE\_STATUS" VARCHAR2(50 BYTE)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XOR\_MENUS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XOR\_MENUS"

( "MENU\_ID" NUMBER(5,0),

"MENU\_NAME" VARCHAR2(50 BYTE),

"PARENT\_MENU\_ID" NUMBER(5,0),

"REL\_ORDER" NUMBER(3,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XOR\_MENUS\_REPORTS\_LINK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XOR\_MENUS\_REPORTS\_LINK"

( "MENU\_ID" NUMBER(5,0),

"REPORT\_ID" NUMBER(5,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XOR\_REPORTS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XOR\_REPORTS"

( "REPORT\_ID" NUMBER(5,0),

"REPORT\_NAME" VARCHAR2(50 BYTE),

"PROJECT\_NAME" VARCHAR2(50 BYTE),

"HOSTED\_APPLICATION" VARCHAR2(50 BYTE),

"HOSTED\_SERVER" VARCHAR2(50 BYTE),

"RELATIVE\_URL" VARCHAR2(200 BYTE),

"DESCRIPTION" VARCHAR2(1024 BYTE),

"REL\_ORDER" NUMBER(3,0),

"OPEN\_OPTION" NUMBER(1,0) DEFAULT 0,

"REP\_LEVEL" VARCHAR2(10 BYTE),

"NEXT\_PAGE" VARCHAR2(100 BYTE),

"PERMANENT\_REWRITE\_NAME" VARCHAR2(200 BYTE),

"HOSTED\_PORT" VARCHAR2(10 BYTE),

"CREATE\_HTACCESS\_REWRITE" CHAR(1 CHAR) DEFAULT 'Y'

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XOR\_REPORTS\_BAK\_10272011

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XOR\_REPORTS\_BAK\_10272011"

( "REPORT\_ID" NUMBER(5,0),

"REPORT\_NAME" VARCHAR2(50 BYTE),

"PROJECT\_NAME" VARCHAR2(50 BYTE),

"HOSTED\_APPLICATION" VARCHAR2(50 BYTE),

"HOSTED\_SERVER" VARCHAR2(50 BYTE),

"RELATIVE\_URL" VARCHAR2(200 BYTE),

"DESCRIPTION" VARCHAR2(1024 BYTE),

"REL\_ORDER" NUMBER(3,0),

"OPEN\_OPTION" NUMBER(1,0),

"REP\_LEVEL" VARCHAR2(10 BYTE),

"NEXT\_PAGE" VARCHAR2(100 BYTE),

"PERMANENT\_REWRITE\_NAME" VARCHAR2(100 BYTE),

"HOSTED\_PORT" VARCHAR2(10 BYTE),

"CREATE\_HTACCESS\_REWRITE" CHAR(1 CHAR)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XOR\_REPORTS\_WITH\_VH

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XOR\_REPORTS\_WITH\_VH"

( "REPORT\_ID" NUMBER(5,0),

"REPORT\_NAME" VARCHAR2(50 BYTE),

"PROJECT\_NAME" VARCHAR2(50 BYTE),

"HOSTED\_APPLICATION" VARCHAR2(50 BYTE),

"HOSTED\_SERVER" VARCHAR2(50 BYTE),

"RELATIVE\_URL" VARCHAR2(200 BYTE),

"DESCRIPTION" VARCHAR2(1024 BYTE),

"REL\_ORDER" NUMBER(3,0),

"OPEN\_OPTION" NUMBER(1,0) DEFAULT 0,

"REP\_LEVEL" VARCHAR2(10 BYTE),

"NEXT\_PAGE" VARCHAR2(100 BYTE),

"PERMANENT\_REWRITE\_NAME" VARCHAR2(100 BYTE),

"HOSTED\_PORT" VARCHAR2(10 BYTE),

"CREATE\_HTACCESS\_REWRITE" CHAR(1 CHAR) DEFAULT 'Y'

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XOR\_TABS

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XOR\_TABS"

( "TAB\_ID" NUMBER(5,0),

"TAB\_NAME" VARCHAR2(30 BYTE),

"REL\_ORDER" NUMBER(3,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Table XOR\_TABS\_MENUS\_LINK

--------------------------------------------------------

CREATE TABLE "XNG\_REPORTS"."XOR\_TABS\_MENUS\_LINK"

( "TAB\_ID" NUMBER(5,0),

"MENU\_ID" NUMBER(5,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS LOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for View BRIX\_FD\_SLA\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."BRIX\_FD\_SLA\_V" ("FRAME\_DELAY\_IN\_MICRO\_SEC", "MAC\_ADDRESS", "VLAN", "CSR\_DEVICE\_NAME", "BRIX\_SWITCH\_NAME", "NAURU\_ID", "BC\_SOURCE", "ACTUAL\_VLAN") AS

SELECT MIN (FRAME\_DELAY\_IN\_MICRO\_SEC) FRAME\_DELAY\_IN\_MICRO\_SEC,

MAC\_ADDRESS,

VLAN,

CSR\_DEVICE\_NAME,

BRIX\_SWITCH\_NAME,

NAURU\_ID,

BC\_SOURCE,

CASE

WHEN vlan > 999 AND vlan < 2000 THEN '3' || SUBSTR (vlan, 2)

END

AS ACTUAL\_VLAN

FROM xng\_reports.BRIX\_FD\_SLA\_20150824

WHERE FRAME\_DELAY\_IN\_MICRO\_SEC IS NOT NULL

GROUP BY MAC\_ADDRESS,

VLAN,

CSR\_DEVICE\_NAME,

BRIX\_SWITCH\_NAME,

NAURU\_ID,

BC\_SOURCE,

ACTUAL\_VLAN

;

--------------------------------------------------------

-- DDL for View BSM\_BMM\_VDDM\_CA\_ISSUE

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."BSM\_BMM\_VDDM\_CA\_ISSUE" ("VSM\_DEVICE\_NAME\_BSM", "IN\_BMM", "IN\_VDDM", "IN\_CA") AS

WITH

ALL\_DEVICE as

(

select vsm\_device\_name\_bsm

from bsm\_mtx\_map

union

select vsm\_device\_name

from vsm\_devices\_domain\_map vddm

where vddm.vsm\_class ='ntBsm'

union

SELECT vsm\_device\_name

FROM cell\_avail\_active ca

WHERE ca.vsm\_device\_name LIKE '%BSM'

)

select dev.vsm\_device\_name\_bsm

, case when bmm.vsm\_device\_name\_bsm is null then 'Missing in bmm' else bmm.vsm\_device\_name\_bsm end in\_bmm

, case when vddm.vsm\_device\_name is null then 'Missing in vddm' else vddm.vsm\_device\_name end in\_vddm

, CASE WHEN ca.vsm\_device\_name IS NULL THEN 'Missing in ca' ELSE ca.vsm\_device\_name END in\_ca

from ALL\_DEVICE dev

left outer join bsm\_mtx\_map bmm

on dev.VSM\_device\_NAME\_bsm = bmm.vsm\_device\_name\_bsm

left outer join vsm\_devices\_domain\_map vddm

on dev.VSM\_device\_NAME\_bsm = vddm.vsm\_device\_name

LEFT OUTER JOIN cell\_avail\_active ca

ON dev.vsm\_device\_name\_bsm = ca.vsm\_device\_name

where dev.vsm\_device\_name\_bsm is null

or vddm.vsm\_device\_name is null

or ca.vsm\_device\_name is null

;

--------------------------------------------------------

-- DDL for View BTOD\_XNG\_PP\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."BTOD\_XNG\_PP\_V" ("BAN\_CATEGORY\_1", "BAN\_CATEGORY\_2", "BAN\_CUSTOM\_FIELD\_1", "MASTER\_BAN", "BAN", "BAN\_CUSTOM\_FIELD\_4", "STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE", "CHARGE\_AMOUNT", "BTP\_EXTRACT\_DATE", "XNG\_TERRITORY", "XNG\_MARKET\_TERRITORY", "XNG\_SUB\_MARKET", "XNG\_AREA", "XNG\_REGION", "BTP\_XNG\_MATCH", "XNG\_CIRC\_INST\_ID", "XNG\_CIRC\_HUM\_ID", "XNG\_TYPE", "XNG\_VENDOR", "XNG\_BANDWIDTH", "XNG\_STATUS", "XNG\_EXTRACT\_DATE", "BAN\_CUSTOM\_FIELD\_3", "BTP\_IN\_SERVICE\_DATE", "XNG\_IN\_SERVICE\_DATE", "ORIG\_COMPANY\_NUM", "ORIG\_COMPANY\_NAME", "INVOICE", "INVOICE\_STATUS", "LATEST\_BILL\_DATE", "Z\_SIDE\_SITE", "Z\_SIDE\_LATITUDE", "Z\_SIDE\_LONGITUDE", "TERM\_LENGTH", "ESTIMATED\_COST", "RENEWAL\_DATE", "TERM\_DURATION", "PATH911") AS

WITH bdp

AS (SELECT bdp.BAN\_CATEGORY\_1,

bdp.BAN\_CATEGORY\_2,

bdp.BAN\_CUSTOM\_FIELD\_1,

bdp.MASTER\_BAN,

bdp.BAN,

bdp.BAN\_CUSTOM\_FIELD\_4,

bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

TO\_NUMBER (bdp.CHARGE\_AMOUNT) charge\_amount,

TO\_DATE (bdp.BTP\_EXTRACT\_DATE, 'yyyymmdd')

BTP\_EXTRACT\_DATE,

'Y' BTP\_XNG\_MATCH,

xng.XNG\_CIRC\_INST\_ID,

xng.XNG\_CIRC\_HUM\_ID,

xng.XNG\_TYPE,

xng.XNG\_VENDOR,

xng.XNG\_BANDWIDTH,

xng.XNG\_STATUS,

bdp.BAN\_CUSTOM\_FIELD\_3,

TO\_DATE (

SUBSTR (in\_service\_date,

1,

INSTR (bdp.in\_service\_date, ' ') - 1),

'yyyy/mm/dd')

in\_service\_date,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

REPLACE (bdp.invoice, CHR (13), '') invoice,

bdp.latest\_bill\_date,

bdp.term\_length,

1 AS match\_type

FROM (SELECT a.\*,

TO\_DATE (

SUBSTR (a.bill\_date,

1,

INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date

FROM xng\_reports.btp\_data\_pp a) bdp,

xng\_reports.stripped\_xng\_circuit\_data xng

WHERE bdp.stripped\_ec\_circuit\_id = xng.xng\_circ\_strip

AND bdp.inventory\_vendor\_code = xng.xng\_vendor\_strip

AND bdp.ban = xng.xng\_ban\_strip

UNION

SELECT bdp.BAN\_CATEGORY\_1,

bdp.BAN\_CATEGORY\_2,

bdp.BAN\_CUSTOM\_FIELD\_1,

bdp.MASTER\_BAN,

bdp.BAN,

bdp.BAN\_CUSTOM\_FIELD\_4,

bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

TO\_NUMBER (bdp.CHARGE\_AMOUNT) charge\_amount,

TO\_DATE (bdp.BTP\_EXTRACT\_DATE, 'yyyymmdd')

BTP\_EXTRACT\_DATE,

'Y' BTP\_XNG\_MATCH,

xng.XNG\_CIRC\_INST\_ID,

xng.XNG\_CIRC\_HUM\_ID,

xng.XNG\_TYPE,

xng.XNG\_VENDOR,

xng.XNG\_BANDWIDTH,

xng.XNG\_STATUS,

bdp.BAN\_CUSTOM\_FIELD\_3,

TO\_DATE (

SUBSTR (in\_service\_date,

1,

INSTR (bdp.in\_service\_date, ' ') - 1),

'yyyy/mm/dd')

in\_service\_date,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

REPLACE (bdp.invoice, CHR (13), '') invoice,

bdp.latest\_bill\_date,

bdp.term\_length,

2 AS match\_type

FROM (SELECT a.\*,

TO\_DATE (

SUBSTR (a.bill\_date,

1,

INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date

FROM xng\_reports.btp\_data\_pp a) bdp,

xng\_reports.stripped\_xng\_circuit\_data xng

WHERE bdp.stripped\_ec\_circuit\_id = xng.xng\_circ\_strip

AND bdp.inventory\_vendor\_code = xng.xng\_vendor\_strip

UNION

SELECT bdp.BAN\_CATEGORY\_1,

bdp.BAN\_CATEGORY\_2,

bdp.BAN\_CUSTOM\_FIELD\_1,

bdp.MASTER\_BAN,

bdp.BAN,

bdp.BAN\_CUSTOM\_FIELD\_4,

bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

TO\_NUMBER (bdp.CHARGE\_AMOUNT) charge\_amount,

TO\_DATE (bdp.BTP\_EXTRACT\_DATE, 'yyyymmdd')

BTP\_EXTRACT\_DATE,

'Y' BTP\_XNG\_MATCH,

xng.XNG\_CIRC\_INST\_ID,

xng.XNG\_CIRC\_HUM\_ID,

xng.XNG\_TYPE,

xng.XNG\_VENDOR,

xng.XNG\_BANDWIDTH,

xng.XNG\_STATUS,

bdp.BAN\_CUSTOM\_FIELD\_3,

TO\_DATE (

SUBSTR (in\_service\_date,

1,

INSTR (bdp.in\_service\_date, ' ') - 1),

'yyyy/mm/dd')

in\_service\_date,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

REPLACE (bdp.invoice, CHR (13), '') invoice,

bdp.latest\_bill\_date,

bdp.term\_length,

3 AS match\_type

FROM (SELECT a.\*,

TO\_DATE (

SUBSTR (a.bill\_date,

1,

INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date

FROM xng\_reports.btp\_data\_pp a) bdp,

xng\_reports.stripped\_xng\_circuit\_data xng

WHERE bdp.stripped\_ec\_circuit\_id = xng.xng\_circ\_strip),

mt

AS (SELECT bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

bdp.BAN,

1 match\_type

FROM xng\_reports.btp\_data\_pp bdp,

xng\_reports.stripped\_xng\_circuit\_data xng

WHERE bdp.stripped\_ec\_circuit\_id = xng.xng\_circ\_strip

AND bdp.inventory\_vendor\_code = xng.xng\_vendor\_strip

AND bdp.ban = xng.xng\_ban\_strip

UNION

SELECT bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

bdp.BAN,

2 match\_type

FROM xng\_reports.btp\_data\_pp bdp,

xng\_reports.stripped\_xng\_circuit\_data xng

WHERE bdp.stripped\_ec\_circuit\_id = xng.xng\_circ\_strip

AND bdp.inventory\_vendor\_code = xng.xng\_vendor\_strip

UNION

SELECT bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

bdp.BAN,

3 match\_type

FROM xng\_reports.btp\_data\_pp bdp,

xng\_reports.stripped\_xng\_circuit\_data xng

WHERE bdp.stripped\_ec\_circuit\_id = xng.xng\_circ\_strip),

mt\_min

AS ( SELECT stripped\_ec\_circuit\_id,

inventory\_vendor\_code,

ban,

MIN (match\_type) match\_type

FROM mt

GROUP BY stripped\_ec\_circuit\_id, inventory\_vendor\_code, ban),

circ

AS (SELECT bdp.BAN\_CATEGORY\_1,

bdp.BAN\_CATEGORY\_2,

bdp.BAN\_CUSTOM\_FIELD\_1,

bdp.MASTER\_BAN,

bdp.BAN,

bdp.BAN\_CUSTOM\_FIELD\_4,

bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

bdp.charge\_amount,

bdp.BTP\_EXTRACT\_DATE,

bdp.BTP\_XNG\_MATCH,

bdp.XNG\_CIRC\_INST\_ID,

bdp.XNG\_CIRC\_HUM\_ID,

bdp.XNG\_TYPE,

bdp.XNG\_VENDOR,

bdp.XNG\_BANDWIDTH,

bdp.XNG\_STATUS,

bdp.BAN\_CUSTOM\_FIELD\_3,

bdp.IN\_SERVICE\_DATE,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

bdp.invoice,

bdp.latest\_bill\_date,

bdp.term\_length,

1 extm

FROM bdp, mt\_min

WHERE bdp.stripped\_ec\_circuit\_id =

mt\_min.stripped\_ec\_circuit\_id

AND bdp.inventory\_vendor\_code =

mt\_min.inventory\_vendor\_code

AND bdp.ban = mt\_min.ban

AND bdp.match\_type = mt\_min.match\_type

UNION

SELECT a.BAN\_CATEGORY\_1,

a.BAN\_CATEGORY\_2,

a.BAN\_CUSTOM\_FIELD\_1,

a.MASTER\_BAN,

a.BAN,

a.BAN\_CUSTOM\_FIELD\_4,

a.STRIPPED\_EC\_CIRCUIT\_ID,

a.INVENTORY\_VENDOR\_CODE,

TO\_NUMBER (a.CHARGE\_AMOUNT) charge\_amount,

TO\_DATE (a.BTP\_EXTRACT\_DATE, 'yyyymmdd') BTP\_EXTRACT\_DATE,

'N' BTP\_XNG\_MATCH,

NULL,

NULL,

NULL,

NULL,

NULL,

NULL,

a.BAN\_CUSTOM\_FIELD\_3,

TO\_DATE (

SUBSTR (in\_service\_date,

1,

INSTR (a.in\_service\_date, ' ') - 1),

'yyyy/mm/dd')

in\_service\_date,

a.orig\_company\_num,

a.orig\_company\_name,

REPLACE (a.invoice, CHR (13), '') invoice,

TO\_DATE (

SUBSTR (a.bill\_date, 1, INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date,

a.term\_length,

1 extm

FROM xng\_reports.btp\_data\_pp a

WHERE NOT EXISTS

(SELECT 1

FROM mt\_min b

WHERE a.stripped\_ec\_circuit\_id =

b.stripped\_ec\_circuit\_id

AND a.inventory\_vendor\_code =

b.inventory\_vendor\_code

AND a.ban = b.ban)),

dom

AS (SELECT bxd.btp\_ban\_category\_1,

drr.territory AS xng\_territory,

drr.market\_territory AS xng\_market\_territory,

drr.sub\_market AS xng\_sub\_market,

drr.area AS xng\_area,

drr.region AS xng\_region,

drr.domain\_inst\_id

FROM xng\_reports.btp\_xng\_domain\_map bxd,

xng\_reports.domains\_regional\_reporting\_new drr

WHERE bxd.xng\_domain\_inst\_id = drr.domain\_inst\_id(+)),

xng\_dom

AS (SELECT cdm.circ\_inst\_id,

drr.territory AS xng\_territory,

drr.market\_territory AS xng\_market\_territory,

drr.sub\_market AS xng\_sub\_market,

drr.area xng\_area,

drr.region xng\_region,

cdm.domain\_inst\_id

FROM vzwnet.circ\_domain\_map cdm,

xng\_reports.domains\_regional\_reporting\_new drr

WHERE cdm.domain\_inst\_id = drr.domain\_inst\_id),

extd

AS (SELECT 1 extm, last\_successful\_exec\_date xng\_extract\_date

FROM xng\_reports.all\_processes

WHERE process\_name = 'XNG\_CIRC\_DATA\_STRIP'),

zs

AS (SELECT ci.circ\_inst\_id,

si.site\_hum\_id,

latitude,

longitude

FROM VZWNET.circ\_INST ci, VZWNET.SITE\_INST si

WHERE ci.z\_site\_id = si.site\_inst\_id),

p911

as (SELECT ci.circ\_inst\_id, 'Y' path911

FROM vzwnet.circ\_inst ci, vzwnet.circ\_path\_inst cpi

WHERE ci.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

and cpi.type like '%911')

SELECT DISTINCT

BAN\_CATEGORY\_1,

BAN\_CATEGORY\_2,

BAN\_CUSTOM\_FIELD\_1,

MASTER\_BAN,

BAN,

BAN\_CUSTOM\_FIELD\_4,

STRIPPED\_EC\_CIRCUIT\_ID,

INVENTORY\_VENDOR\_CODE,

charge\_amount,

BTP\_EXTRACT\_DATE,

NVL (dom.xng\_territory, NVL (xd.xng\_territory, 'Unknown')) AS xng\_territory,

NVL (dom.xng\_market\_territory, NVL (xd.xng\_market\_territory, 'Unknown')) AS xng\_market\_territory,

NVL (dom.xng\_sub\_market, NVL (xd.xng\_sub\_market, 'Unknown')) AS xng\_sub\_market,

NVL (dom.xng\_area, NVL (xd.xng\_area, 'Unknown')) AS xng\_area,

NVL (dom.xng\_region, NVL (xd.xng\_region, 'Unknown')) AS xng\_region,

BTP\_XNG\_MATCH,

XNG\_CIRC\_INST\_ID,

XNG\_CIRC\_HUM\_ID,

XNG\_TYPE,

XNG\_VENDOR,

XNG\_BANDWIDTH,

XNG\_STATUS,

extd.XNG\_EXTRACT\_DATE,

BAN\_CUSTOM\_FIELD\_3,

IN\_SERVICE\_DATE btp\_in\_service\_date,

ci.in\_service xng\_in\_service\_date,

orig\_company\_num,

orig\_company\_name,

RTRIM (LTRIM (SUBSTR (invoice,

1,

INSTR (invoice,

' - ',

-1,

1)

- 1)))

invoice,

RTRIM (LTRIM (SUBSTR (invoice,

INSTR (invoice,

' - ',

-1,

1)

+ 3)))

invoice\_status,

latest\_bill\_date,

zs.site\_hum\_id AS z\_side\_site,

zs.latitude AS z\_side\_latitude,

zs.longitude AS z\_side\_longitude,

CASE

WHEN ci.renewal\_date IS NULL THEN NULL

ELSE ROUND ( (ci.renewal\_date - ci.in\_service) / 30)

END

term\_length,

ci.non\_recur\_costs AS estimated\_cost,

ci.renewal\_date,

ci.distance AS term\_duration,

nvl(p911.path911,'N') as path911

FROM circ,

dom,

xng\_dom xd,

extd,

zs,

p911,

vzwnet.circ\_inst ci

WHERE circ.ban\_category\_1 = dom.btp\_ban\_category\_1(+)

AND circ.xng\_circ\_inst\_id = xd.circ\_inst\_id(+)

AND circ.extm = extd.extm(+)

AND circ.xng\_circ\_inst\_id = zs.circ\_inst\_id(+)

AND circ.xng\_circ\_inst\_id = p911.circ\_inst\_id(+)

AND circ.xng\_circ\_inst\_id = ci.circ\_inst\_id(+)

;

--------------------------------------------------------

-- DDL for View BTP\_XNG\_AUDIT\_HIST\_PP\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_HIST\_PP\_V" ("BAN\_CATEGORY\_1", "BAN\_CATEGORY\_2", "BAN\_CUSTOM\_FIELD\_1", "MASTER\_BAN", "BAN", "BAN\_CUSTOM\_FIELD\_4", "STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE", "CHARGE\_AMOUNT", "BILL\_DATE\_MONTH", "BILL\_DATE\_YEAR", "MANUAL\_ENTRY", "BTP\_EXTRACT\_DATE", "BTP\_XNG\_MATCH", "XNG\_AREA", "XNG\_REGION", "XNG\_CIRC\_INST\_ID", "XNG\_CIRC\_HUM\_ID", "XNG\_TYPE", "XNG\_VENDOR", "XNG\_BANDWIDTH", "XNG\_STATUS", "XNG\_CIRC\_PATH\_INST\_ID", "XNG\_NEXT\_PATH\_INST\_ID", "XNG\_EXTRACT\_DATE", "UNIQUELY\_IDENTIFIED", "BILL\_DATE", "BAN\_CUSTOM\_FIELD\_3", "IN\_SERVICE\_DATE", "MATCH\_CODE", "MATCH\_STATUS", "XNG\_BAN", "ORIG\_COMPANY\_NUM", "ORIG\_COMPANY\_NAME", "INVOICE", "TERM\_LENGTH", "LATEST\_BILL\_DATE") AS

SELECT bdp.BAN\_CATEGORY\_1,

nvl(bxa.BAN\_CATEGORY\_2, bdp.BAN\_CATEGORY\_2) BAN\_CATEGORY\_2,

nvl(bxa.BAN\_CUSTOM\_FIELD\_1, bdp.BAN\_CUSTOM\_FIELD\_1) BAN\_CUSTOM\_FIELD\_1,

nvl(bxa.MASTER\_BAN, bdp.MASTER\_BAN) MASTER\_BAN,

nvl(bxa.BAN, bdp.BAN) BAN,

nvl(bxa.BAN\_CUSTOM\_FIELD\_4, bdp.BAN\_CUSTOM\_FIELD\_4) BAN\_CUSTOM\_FIELD\_4,

nvl(bxa.STRIPPED\_EC\_CIRCUIT\_ID, bdp.STRIPPED\_EC\_CIRCUIT\_ID) STRIPPED\_EC\_CIRCUIT\_ID,

nvl(bxa.INVENTORY\_VENDOR\_CODE, bdp.INVENTORY\_VENDOR\_CODE) INVENTORY\_VENDOR\_CODE,

nvl(bxa.CHARGE\_AMOUNT, to\_number(bdp.CHARGE\_AMOUNT)) charge\_amount,

nvl(bxa.BILL\_DATE\_MONTH, bdp.BILL\_DATE\_MONTH) BILL\_DATE\_MONTH,

nvl(bxa.BILL\_DATE\_YEAR, bdp.BILL\_DATE\_YEAR) BILL\_DATE\_YEAR,

nvl(bxa.MANUAL\_ENTRY, bdp.MANUAL\_ENTRY) MANUAL\_ENTRY,

nvl(bxa.BTP\_EXTRACT\_DATE, TO\_DATE(bdp.BTP\_EXTRACT\_DATE, 'yyyymmdd')) BTP\_EXTRACT\_DATE,

bxa.BTP\_XNG\_MATCH,

bxa.XNG\_AREA,

bxa.XNG\_REGION,

bxa.XNG\_CIRC\_INST\_ID,

bxa.XNG\_CIRC\_HUM\_ID,

bxa.XNG\_TYPE,

bxa.XNG\_VENDOR,

bxa.XNG\_BANDWIDTH,

bxa.XNG\_STATUS,

bxa.XNG\_CIRC\_PATH\_INST\_ID,

bxa.XNG\_NEXT\_PATH\_INST\_ID,

bxa.XNG\_EXTRACT\_DATE,

bxa.UNIQUELY\_IDENTIFIED,

bxa.BILL\_DATE,

bxa.BAN\_CUSTOM\_FIELD\_3,

bxa.IN\_SERVICE\_DATE,

bxa.MATCH\_CODE,

bxa.MATCH\_STATUS,

bxa.XNG\_BAN,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

bdp.invoice,

bdp.term\_length,

bdp.latest\_bill\_date

FROM (SELECT a.\*,

TO\_DATE (SUBSTR (a.bill\_date, 1, INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date

FROM xng\_reports.btp\_data\_pp a) bdp,

xng\_reports.btp\_xng\_audit\_hist bxa

WHERE bdp.stripped\_ec\_circuit\_id = bxa.stripped\_ec\_circuit\_id(+)

AND bdp.inventory\_vendor\_code = bxa.inventory\_vendor\_code(+)

AND bdp.ban = bxa.ban(+)

UNION

SELECT bxa.BAN\_CATEGORY\_1,

bxa.BAN\_CATEGORY\_2,

bxa.BAN\_CUSTOM\_FIELD\_1,

bxa.MASTER\_BAN,

bxa.BAN,

bxa.BAN\_CUSTOM\_FIELD\_4,

bxa.STRIPPED\_EC\_CIRCUIT\_ID,

bxa.INVENTORY\_VENDOR\_CODE,

bxa.CHARGE\_AMOUNT,

bxa.BILL\_DATE\_MONTH,

bxa.BILL\_DATE\_YEAR,

bxa.MANUAL\_ENTRY,

bxa.BTP\_EXTRACT\_DATE,

bxa.BTP\_XNG\_MATCH,

bxa.XNG\_AREA,

bxa.XNG\_REGION,

bxa.XNG\_CIRC\_INST\_ID,

bxa.XNG\_CIRC\_HUM\_ID,

bxa.XNG\_TYPE,

bxa.XNG\_VENDOR,

bxa.XNG\_BANDWIDTH,

bxa.XNG\_STATUS,

bxa.XNG\_CIRC\_PATH\_INST\_ID,

bxa.XNG\_NEXT\_PATH\_INST\_ID,

bxa.XNG\_EXTRACT\_DATE,

bxa.UNIQUELY\_IDENTIFIED,

bxa.BILL\_DATE,

bxa.BAN\_CUSTOM\_FIELD\_3,

bxa.IN\_SERVICE\_DATE,

bxa.MATCH\_CODE,

bxa.MATCH\_STATUS,

bxa.XNG\_BAN,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

bdp.invoice,

bdp.term\_length,

bdp.latest\_bill\_date

FROM xng\_reports.btp\_xng\_audit\_hist bxa,

(SELECT a.\*,

TO\_DATE (SUBSTR (a.bill\_date, 1, INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date

FROM xng\_reports.btp\_data\_pp a) bdp

WHERE bxa.stripped\_ec\_circuit\_id = bdp.stripped\_ec\_circuit\_id(+)

AND bxa.inventory\_vendor\_code = bdp.inventory\_vendor\_code(+)

AND bxa.ban = bdp.ban(+)

;

--------------------------------------------------------

-- DDL for View BTP\_XNG\_PP\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."BTP\_XNG\_PP\_V" ("BAN\_CATEGORY\_1", "BAN\_CATEGORY\_2", "BAN\_CUSTOM\_FIELD\_1", "MASTER\_BAN", "BAN", "BAN\_CUSTOM\_FIELD\_4", "STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE", "CHARGE\_AMOUNT", "BTP\_EXTRACT\_DATE", "XNG\_AREA", "XNG\_REGION", "BTP\_XNG\_MATCH", "XNG\_CIRC\_INST\_ID", "XNG\_CIRC\_HUM\_ID", "XNG\_TYPE", "XNG\_VENDOR", "XNG\_BANDWIDTH", "XNG\_STATUS", "XNG\_EXTRACT\_DATE", "BAN\_CUSTOM\_FIELD\_3", "IN\_SERVICE\_DATE", "ORIG\_COMPANY\_NUM", "ORIG\_COMPANY\_NAME", "INVOICE", "LATEST\_BILL\_DATE", "Z\_SIDE\_SITE", "TERM\_LENGTH") AS

WITH circ

AS ( SELECT BAN\_CATEGORY\_1,

BAN\_CATEGORY\_2,

BAN\_CUSTOM\_FIELD\_1,

MASTER\_BAN,

BAN,

BAN\_CUSTOM\_FIELD\_4,

STRIPPED\_EC\_CIRCUIT\_ID,

INVENTORY\_VENDOR\_CODE,

charge\_amount,

BTP\_EXTRACT\_DATE,

MAX (BTP\_XNG\_MATCH) BTP\_XNG\_MATCH,

MAX (XNG\_CIRC\_INST\_ID) XNG\_CIRC\_INST\_ID,

MAX (XNG\_CIRC\_HUM\_ID) XNG\_CIRC\_HUM\_ID,

MAX (XNG\_TYPE) XNG\_TYPE,

MAX (XNG\_VENDOR) XNG\_VENDOR,

MAX (XNG\_BANDWIDTH) XNG\_BANDWIDTH,

MAX (XNG\_STATUS) XNG\_STATUS,

BAN\_CUSTOM\_FIELD\_3,

IN\_SERVICE\_DATE,

orig\_company\_num,

orig\_company\_name,

invoice,

latest\_bill\_date,

term\_length,

1 extm

FROM (SELECT bdp.BAN\_CATEGORY\_1,

bdp.BAN\_CATEGORY\_2,

bdp.BAN\_CUSTOM\_FIELD\_1,

bdp.MASTER\_BAN,

bdp.BAN,

bdp.BAN\_CUSTOM\_FIELD\_4,

bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

TO\_NUMBER (bdp.CHARGE\_AMOUNT) charge\_amount,

TO\_DATE (bdp.BTP\_EXTRACT\_DATE, 'yyyymmdd')

BTP\_EXTRACT\_DATE,

'N' BTP\_XNG\_MATCH,

NULL XNG\_CIRC\_INST\_ID,

NULL XNG\_CIRC\_HUM\_ID,

NULL XNG\_TYPE,

NULL XNG\_VENDOR,

NULL XNG\_BANDWIDTH,

NULL XNG\_STATUS,

bdp.BAN\_CUSTOM\_FIELD\_3,

bdp.IN\_SERVICE\_DATE,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

bdp.invoice,

bdp.latest\_bill\_date,

bdp.term\_length

FROM (SELECT a.\*,

TO\_DATE (

SUBSTR (a.bill\_date,

1,

INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date

FROM xng\_reports.btp\_data\_pp a) bdp

UNION

SELECT bdp.BAN\_CATEGORY\_1,

bdp.BAN\_CATEGORY\_2,

bdp.BAN\_CUSTOM\_FIELD\_1,

bdp.MASTER\_BAN,

bdp.BAN,

bdp.BAN\_CUSTOM\_FIELD\_4,

bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

TO\_NUMBER (bdp.CHARGE\_AMOUNT) charge\_amount,

TO\_DATE (bdp.BTP\_EXTRACT\_DATE, 'yyyymmdd')

BTP\_EXTRACT\_DATE,

'Y' BTP\_XNG\_MATCH,

xng.XNG\_CIRC\_INST\_ID,

xng.XNG\_CIRC\_HUM\_ID,

xng.XNG\_TYPE,

xng.XNG\_VENDOR,

xng.XNG\_BANDWIDTH,

xng.XNG\_STATUS,

bdp.BAN\_CUSTOM\_FIELD\_3,

bdp.IN\_SERVICE\_DATE,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

bdp.invoice,

bdp.latest\_bill\_date,

bdp.term\_length

FROM (SELECT a.\*,

TO\_DATE (

SUBSTR (a.bill\_date,

1,

INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date

FROM xng\_reports.btp\_data\_pp a) bdp,

xng\_reports.stripped\_xng\_circuit\_data xng

WHERE bdp.stripped\_ec\_circuit\_id =

xng.xng\_circ\_strip

AND bdp.inventory\_vendor\_code =

xng.xng\_vendor\_strip

AND bdp.ban = xng.xng\_ban\_strip

UNION

SELECT bdp.BAN\_CATEGORY\_1,

bdp.BAN\_CATEGORY\_2,

bdp.BAN\_CUSTOM\_FIELD\_1,

bdp.MASTER\_BAN,

bdp.BAN,

bdp.BAN\_CUSTOM\_FIELD\_4,

bdp.STRIPPED\_EC\_CIRCUIT\_ID,

bdp.INVENTORY\_VENDOR\_CODE,

TO\_NUMBER (bdp.CHARGE\_AMOUNT) charge\_amount,

TO\_DATE (bdp.BTP\_EXTRACT\_DATE, 'yyyymmdd')

BTP\_EXTRACT\_DATE,

'Y' BTP\_XNG\_MATCH,

xng.XNG\_CIRC\_INST\_ID,

xng.XNG\_CIRC\_HUM\_ID,

xng.XNG\_TYPE,

xng.XNG\_VENDOR,

xng.XNG\_BANDWIDTH,

xng.XNG\_STATUS,

bdp.BAN\_CUSTOM\_FIELD\_3,

bdp.IN\_SERVICE\_DATE,

bdp.orig\_company\_num,

bdp.orig\_company\_name,

bdp.invoice,

bdp.latest\_bill\_date,

bdp.term\_length

FROM (SELECT a.\*,

TO\_DATE (

SUBSTR (a.bill\_date,

1,

INSTR (a.bill\_date, ' ')),

'yyyy/mm/dd')

latest\_bill\_date

FROM xng\_reports.btp\_data\_pp a) bdp,

xng\_reports.stripped\_xng\_circuit\_data xng

WHERE bdp.stripped\_ec\_circuit\_id =

xng.xng\_circ\_strip

AND bdp.inventory\_vendor\_code =

xng.xng\_vendor\_strip)

GROUP BY BAN\_CATEGORY\_1,

BAN\_CATEGORY\_2,

BAN\_CUSTOM\_FIELD\_1,

MASTER\_BAN,

BAN,

BAN\_CUSTOM\_FIELD\_4,

STRIPPED\_EC\_CIRCUIT\_ID,

INVENTORY\_VENDOR\_CODE,

charge\_amount,

BTP\_EXTRACT\_DATE,

BAN\_CUSTOM\_FIELD\_3,

IN\_SERVICE\_DATE,

orig\_company\_num,

orig\_company\_name,

invoice,

latest\_bill\_date,

term\_length),

dom

AS (SELECT bxd.btp\_ban\_category\_1,

drr.area AS xng\_area,

drr.region AS xng\_region,

drr.domain\_inst\_id

FROM xng\_reports.btp\_xng\_domain\_map bxd,

xng\_reports.domains\_regional\_reporting drr

WHERE bxd.xng\_domain\_inst\_id = drr.domain\_inst\_id(+)),

extd

AS (SELECT 1 extm, last\_successful\_exec\_date xng\_extract\_date

FROM xng\_reports.all\_processes

WHERE process\_name = 'XNG\_CIRC\_DATA\_STRIP'),

zs

AS (SELECT ci.circ\_inst\_id, si.site\_hum\_id

FROM VZWNET.circ\_INST ci, VZWNET.SITE\_INST si

WHERE ci.z\_site\_id = si.site\_inst\_id)

SELECT BAN\_CATEGORY\_1,

BAN\_CATEGORY\_2,

BAN\_CUSTOM\_FIELD\_1,

MASTER\_BAN,

BAN,

BAN\_CUSTOM\_FIELD\_4,

STRIPPED\_EC\_CIRCUIT\_ID,

INVENTORY\_VENDOR\_CODE,

charge\_amount,

BTP\_EXTRACT\_DATE,

NVL (dom.xng\_area, 'Unknown') AS xng\_area,

NVL (dom.xng\_region, 'Unknown') AS xng\_region,

BTP\_XNG\_MATCH,

XNG\_CIRC\_INST\_ID,

XNG\_CIRC\_HUM\_ID,

XNG\_TYPE,

XNG\_VENDOR,

XNG\_BANDWIDTH,

XNG\_STATUS,

extd.XNG\_EXTRACT\_DATE,

BAN\_CUSTOM\_FIELD\_3,

IN\_SERVICE\_DATE,

orig\_company\_num,

orig\_company\_name,

invoice,

latest\_bill\_date,

zs.site\_hum\_id AS z\_side\_site,

term\_length

FROM circ,

dom,

extd,

zs

WHERE circ.ban\_category\_1 = dom.btp\_ban\_category\_1(+)

AND circ.extm = extd.extm(+)

AND circ.xng\_circ\_inst\_id = zs.circ\_inst\_id(+)

;

--------------------------------------------------------

-- DDL for View BTS\_COUNT\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."BTS\_COUNT\_V" ("AREA", "REGION", "NUMBER\_OF\_CELLS") AS

WITH

ALL\_ALU\_BTS as

(

select p.SYS\_ID, p.ECP\_SID, p.CELL\_NUMBER

from PROS\_ECP\_CELL\_DS1\_MAP p

union

select p.SYS\_ID, p.ECP\_SID, p.CELL\_NUMBER

from PROS\_ECP\_CELL\_EBH\_MAP p

)

,

ALL\_MT\_BTS as

(

select p.VSM\_DEVICE\_NAME\_OMCR , p.bts\_NUMBER

from MOTO\_CDMA\_SPANS p

union

select p.VSM\_DEVICE\_NAME\_OMCR, p.bts\_NUMBER

from MOTO\_CDMA\_ETHERNET\_INV p

)

,

CNT\_ALL\_BTS as

(

select drr.area, drr.region

, b.cell\_number

from ALL\_ALU\_BTS b

join ecp\_switch\_map esm

on esm.SYS\_ID = b.sys\_id

and esm.ecp\_sID = b.ecp\_sid

join domains\_regional\_reporting drr

on esm.XNG\_DOMAIN\_INST\_ID = drr.DOMAIN\_INST\_ID

-- group by rollup(drr.area, drr.region)

union all

select drr.area, drr.region

, to\_number(b.CELL\_NUMBER)

from BSM\_CELL\_DCG\_MAP b

join clli\_domain\_map esm

on substr(esm.CLLI, 1, 6) = substr(b.VSM\_DEVICE\_NAME\_BSM, 1,6)

join domains\_leaf\_reporting drr

on esm.LEAF\_DOMAIN\_INST\_ID = drr.DOMAIN\_INST\_ID

union all

select drr.area, drr.region

, to\_number(b.bts\_NUMBER)

from ALL\_MT\_BTS b

join clli\_domain\_map esm

on substr(esm.CLLI, 1, 6) = substr(b.VSM\_DEVICE\_NAME\_OMCR, 1,6)

join domains\_leaf\_reporting drr

on esm.LEAF\_DOMAIN\_INST\_ID = drr.DOMAIN\_INST\_ID

)

select area, region, count(cell\_number) number\_of\_cells

from CNT\_ALL\_BTS

group by rollup(area, region)

order by area, region

;

--------------------------------------------------------

-- DDL for View CHG\_REQ\_MGMT\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CHG\_REQ\_MGMT\_V" ("DOMAIN\_INST\_ID", "AREA", "REGION", "DOMAIN\_NAME", "CHG\_REQ\_INST\_ID", "ELEMENT\_TYPE", "ELEMENT\_INST\_ID", "ELEMENT\_ORIG\_NAME", "SUBMIT\_USER", "SUBMIT\_TIMESTAMP", "CHG\_DESCRIPTION", "STATUS\_ABBR", "RESP\_USER", "RESP\_TIMESTAMP", "CHG\_DISPOSITION", "DAYS\_SUBMIT", "DAYS\_RESP", "SUBMIT\_GRP", "RESP\_GRP", "STATUS\_NAME", "EXTRACT\_DATE") AS

WITH genchg

AS (SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

edm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.equip\_domain\_map edm

WHERE gcr.element\_type IN ('Q', 'E')

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = edm.equip\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

cdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.circ\_domain\_map cdm

WHERE gcr.element\_type = 'S'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = cdm.circ\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

bdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.cable\_domain\_map bdm

WHERE (gcr.element\_type = 'B')

AND gcr.element\_inst\_id = bdm.cable\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

pdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.path\_domain\_map pdm

WHERE gcr.element\_type IN ('P', 'K')

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = pdm.circ\_path\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

sdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.site\_domain\_map sdm

WHERE (gcr.element\_type = 'I')

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = sdm.site\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

udm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.udc\_domain\_map udm

WHERE gcr.element\_type = '1'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = udm.udc\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

adm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr,

vzwnet.ip\_address\_domain\_map adm

WHERE (gcr.element\_type = 'D')

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = adm.address\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

rdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr,

vzwnet.ip\_sub\_range\_domain\_map rdm

WHERE gcr.element\_type = 'U'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = rdm.sub\_range\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

ndm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr,

vzwnet.rm\_number\_range\_domain\_map ndm

WHERE gcr.element\_type = '4'

AND gcr.element\_inst\_id = ndm.number\_range\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

hdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.cloud\_domain\_map hdm

WHERE gcr.element\_type = 'H'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = hdm.cloud\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

qdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr,

vzwnet.qos\_policy\_domain\_map qdm

WHERE gcr.element\_type = 'J'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = qdm.qos\_policy\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

vdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.cust\_domain\_map vdm

WHERE gcr.element\_type = 'M'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = vdm.cust\_inst\_id),

leaf

AS ( SELECT domain\_inst\_id,

domain\_name,

area,

region

FROM xng\_reports.domains\_leaf\_reporting

WHERE area <> 'OSS'

ORDER BY area, region, domain\_name)

SELECT g.domain\_inst\_id,

NVL (l.area, 'Unknown') area,

NVL (l.region, 'Unknown') region,

NVL (l.domain\_name, 'Unknown') domain\_name,

g.chg\_req\_inst\_id,

g.element\_type,

g.element\_inst\_id,

g.element\_orig\_name,

g.submit\_user,

g.submit\_timestamp,

g.chg\_description,

g.status\_abbr,

g.resp\_user,

g.resp\_timestamp,

g.chg\_disposition,

g.days\_submit,

g.days\_resp,

CASE

WHEN g.days\_submit <= 7 THEN '1'

WHEN g.days\_submit >= 8 AND g.days\_submit <= 80 THEN '2'

WHEN g.days\_submit >= 31 AND g.days\_submit <= 60 THEN '3'

WHEN g.days\_submit >= 61 AND g.days\_submit <= 90 THEN '4'

ELSE '5'

END

submit\_grp,

CASE

WHEN g.days\_resp IS NULL THEN '0'

WHEN g.days\_resp <= 7 THEN '1'

WHEN g.days\_resp >= 8 AND g.days\_resp <= 80 THEN '2'

WHEN g.days\_resp >= 31 AND g.days\_resp <= 60 THEN '3'

WHEN g.days\_resp >= 61 AND g.days\_resp <= 90 THEN '4'

ELSE '5'

END

resp\_grp,

NVL (s.status\_name, 'Unknown') status\_name,

SYSDATE AS extract\_date

FROM genchg g, leaf l, vzwnet.chg\_req\_status\_codes s

WHERE g.domain\_inst\_id = l.domain\_inst\_id

AND g.status\_abbr = s.status\_abbr

;

--------------------------------------------------------

-- DDL for View CHG\_REQ\_MGMT\_\_SATYA\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CHG\_REQ\_MGMT\_\_SATYA\_V" ("DOMAIN\_INST\_ID", "AREA", "REGION", "DOMAIN\_NAME", "CHG\_REQ\_INST\_ID", "ELEMENT\_TYPE", "ELEMENT\_INST\_ID", "ELEMENT\_ORIG\_NAME", "SUBMIT\_USER", "SUBMIT\_TIMESTAMP", "CHG\_DESCRIPTION", "STATUS\_ABBR", "RESP\_USER", "RESP\_TIMESTAMP", "CHG\_DISPOSITION", "DAYS\_SUBMIT", "DAYS\_RESP", "SUBMIT\_GRP", "RESP\_GRP", "RESP\_GRP\_APPLIED", "STATUS\_NAME", "EXTRACT\_DATE") AS

WITH genchg

AS (SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

edm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.equip\_domain\_map edm

WHERE gcr.element\_type IN ('Q', 'E')

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = edm.equip\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

cdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.circ\_domain\_map cdm

WHERE gcr.element\_type = 'S'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = cdm.circ\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

bdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.cable\_domain\_map bdm

WHERE (gcr.element\_type = 'B')

AND gcr.element\_inst\_id = bdm.cable\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

pdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.path\_domain\_map pdm

WHERE gcr.element\_type IN ('P', 'K')

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = pdm.circ\_path\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

sdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.site\_domain\_map sdm

WHERE (gcr.element\_type = 'I')

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = sdm.site\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

udm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.udc\_domain\_map udm

WHERE gcr.element\_type = '1'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = udm.udc\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

adm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr,

vzwnet.ip\_address\_domain\_map adm

WHERE (gcr.element\_type = 'D')

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = adm.address\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

rdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr,

vzwnet.ip\_sub\_range\_domain\_map rdm

WHERE gcr.element\_type = 'U'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = rdm.sub\_range\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

ndm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr,

vzwnet.rm\_number\_range\_domain\_map ndm

WHERE gcr.element\_type = '4'

AND gcr.element\_inst\_id = ndm.number\_range\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

hdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.cloud\_domain\_map hdm

WHERE gcr.element\_type = 'H'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = hdm.cloud\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

qdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr,

vzwnet.qos\_policy\_domain\_map qdm

WHERE gcr.element\_type = 'J'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = qdm.qos\_policy\_inst\_id

UNION

SELECT gcr.chg\_req\_inst\_id,

gcr.element\_type,

gcr.element\_inst\_id,

gcr.element\_orig\_name,

gcr.submit\_user,

gcr.submit\_timestamp,

gcr.chg\_description,

gcr.status\_abbr,

gcr.resp\_user,

gcr.resp\_timestamp,

gcr.chg\_disposition,

ROUND (SYSDATE - submit\_timestamp) days\_submit,

ROUND (SYSDATE - resp\_timestamp) days\_resp,

vdm.domain\_inst\_id

FROM vzwnet.gen\_chg\_req\_inst gcr, vzwnet.cust\_domain\_map vdm

WHERE gcr.element\_type = 'M'

AND status\_abbr IN ('N', 'A')

AND gcr.element\_inst\_id = vdm.cust\_inst\_id),

leaf

AS ( SELECT domain\_inst\_id,

domain\_name,

area,

region

FROM xng\_reports.domains\_leaf\_reporting

WHERE area <> 'OSS'

ORDER BY area, region, domain\_name)

SELECT g.domain\_inst\_id,

NVL (l.area, 'Unknown') area,

NVL (l.region, 'Unknown') region,

NVL (l.domain\_name, 'Unknown') domain\_name,

g.chg\_req\_inst\_id,

g.element\_type,

g.element\_inst\_id,

g.element\_orig\_name,

g.submit\_user,

g.submit\_timestamp,

g.chg\_description,

g.status\_abbr,

g.resp\_user,

g.resp\_timestamp,

g.chg\_disposition,

g.days\_submit,

g.days\_resp,

CASE

WHEN g.days\_resp IS NULL and g.days\_submit <= 7 THEN '1'

WHEN g.days\_resp IS NULL and g.days\_submit >= 8 AND g.days\_submit <= 80 THEN '2'

WHEN g.days\_resp IS NULL and g.days\_submit >= 31 AND g.days\_submit <= 60 THEN '3'

WHEN g.days\_resp IS NULL and g.days\_submit >= 61 AND g.days\_submit <= 90 THEN '4'

ELSE case when g.days\_resp IS NULL then '5' else '0' end

END

submit\_grp,

CASE

WHEN g.days\_resp IS NULL THEN '0'

WHEN g.days\_resp <= 7 THEN '1'

WHEN g.days\_resp >= 8 AND g.days\_resp <= 80 THEN '2'

WHEN g.days\_resp >= 31 AND g.days\_resp <= 60 THEN '3'

WHEN g.days\_resp >= 61 AND g.days\_resp <= 90 THEN '4'

ELSE '5'

END

resp\_grp,

CASE

WHEN s.status\_name = 'Applied' and g.days\_resp IS NULL THEN '0'

WHEN s.status\_name = 'Applied' and g.days\_resp <= 7 THEN '1'

WHEN s.status\_name = 'Applied' and g.days\_resp >= 8 AND g.days\_resp <= 80 THEN '2'

WHEN s.status\_name = 'Applied' and g.days\_resp >= 31 AND g.days\_resp <= 60 THEN '3'

WHEN s.status\_name = 'Applied' and g.days\_resp >= 61 AND g.days\_resp <= 90 THEN '4'

ELSE case when s.status\_name = 'Applied' then '5' end

END

resp\_grp\_applied,

NVL (s.status\_name, 'Unknown') status\_name,

SYSDATE AS extract\_date

FROM genchg g, leaf l, vzwnet.chg\_req\_status\_codes s

WHERE g.domain\_inst\_id = l.domain\_inst\_id

AND g.status\_abbr = s.status\_abbr

;

--------------------------------------------------------

-- DDL for View CIRC\_BC\_THRESHOLD\_DATA\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CIRC\_BC\_THRESHOLD\_DATA\_VW" ("VLAN\_CIRC\_PATH\_INST\_ID", "BC\_FRAME\_DELAY", "FD\_WARN\_THRESHOLD", "FD\_CRITICAL\_THRESHOLD", "BC\_FRAME\_DELAY\_VARIATION", "FDV\_WARN\_THRESHOLD", "FDV\_CRITICAL\_THRESHOLD", "CLASS\_OF\_SERVICE") AS

select pas2."VLAN\_CIRC\_PATH\_INST\_ID",pas2."BC\_FRAME\_DELAY",pas2."FD\_WARN\_THRESHOLD",pas2."FD\_CRITICAL\_THRESHOLD",pas2."BC\_FRAME\_DELAY\_VARIATION",pas2."FDV\_WARN\_THRESHOLD",pas2."FDV\_CRITICAL\_THRESHOLD",pas2."CLASS\_OF\_SERVICE"

from vzwnet.circ\_path\_inst cpi

join ( select PAS.CIRC\_PATH\_INST\_ID VLAN\_CIRC\_PATH\_INST\_ID, PAS.ATTR\_VALUE BC\_FRAME\_DELAY,

to\_char(case when PAS.ATTR\_VALUE \* 1.4 > 5.0 then PAS.ATTR\_VALUE \* 1.4 else 5.0 end) FD\_WARN\_THRESHOLD,

to\_char(case when PAS.ATTR\_VALUE \* 1.7 > 10.0 then PAS.ATTR\_VALUE \* 1.7 else 10.0 end) FD\_CRITICAL\_THRESHOLD,

trim(to\_char(PAS1.ATTR\_VALUE, '999.999999999')) BC\_FRAME\_DELAY\_VARIATION,

to\_char( 1.5 ) FDV\_WARN\_THRESHOLD,to\_char( 2.0 ) FDV\_CRITICAL\_THRESHOLD, 'p5' class\_of\_service

from (select \* from vzwnet.circ\_path\_attr\_settings where VAL\_ATTR\_INST\_ID = 7327 and attr\_value <= 50 ) pas

full outer join

(select \* from vzwnet.circ\_path\_attr\_settings where VAL\_ATTR\_INST\_ID = 7328 and attr\_value <= 2) pas1 on PAS.CIRC\_PATH\_INST\_ID = PAS1.CIRC\_PATH\_INST\_ID

) pas2 on CPI.CIRC\_PATH\_INST\_ID = PAS2.VLAN\_CIRC\_PATH\_INST\_ID

union

select pas2."VLAN\_CIRC\_PATH\_INST\_ID",pas2."BC\_FRAME\_DELAY",pas2."FD\_WARN\_THRESHOLD",pas2."FD\_CRITICAL\_THRESHOLD",pas2."BC\_FRAME\_DELAY\_VARIATION",pas2."FDV\_WARN\_THRESHOLD",pas2."FDV\_CRITICAL\_THRESHOLD",pas2."CLASS\_OF\_SERVICE"

from vzwnet.circ\_path\_inst cpi

join ( select PAS.CIRC\_PATH\_INST\_ID VLAN\_CIRC\_PATH\_INST\_ID, PAS.ATTR\_VALUE BC\_FRAME\_DELAY,

to\_char(case when PAS.ATTR\_VALUE \* 1.7 > 30.0 then PAS.ATTR\_VALUE \* 1.7 else 30.0 end) FD\_WARN\_THRESHOLD,

to\_char(case when PAS.ATTR\_VALUE \* 1.9 > 50.0 then PAS.ATTR\_VALUE \* 1.9 else 50.0 end) FD\_CRITICAL\_THRESHOLD,

trim(to\_char(PAS1.ATTR\_VALUE, '999.999999999')) BC\_FRAME\_DELAY\_VARIATION,

to\_char(7.0) FDV\_WARN\_THRESHOLD,to\_char(10.0) FDV\_CRITICAL\_THRESHOLD, 'p0' class\_of\_service

from (select \* from vzwnet.circ\_path\_attr\_settings where VAL\_ATTR\_INST\_ID = 7327 and attr\_value <= 50 ) pas

full outer join

(select \* from vzwnet.circ\_path\_attr\_settings where VAL\_ATTR\_INST\_ID = 7328 and attr\_value <= 2) pas1 on PAS.CIRC\_PATH\_INST\_ID = PAS1.CIRC\_PATH\_INST\_ID

) pas2 on CPI.CIRC\_PATH\_INST\_ID = PAS2.VLAN\_CIRC\_PATH\_INST\_ID

;

--------------------------------------------------------

-- DDL for View CIRC\_SEG\_NEW\_DECOM\_DET\_TESS\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DET\_TESS\_VW" ("CIRC\_INST\_ID", "SEGMENT\_NAME", "STATUS", "SEGMENT\_TYPE", "BANDWIDTH", "VENDOR", "RPT\_ID", "DOMAIN\_INST\_ID", "DOMAIN", "AREA", "REGION", "MARKET", "Z\_SITE\_ID", "SITE\_NAME", "ACTION\_TYPE", "CHG\_TS", "CHG\_BY", "MRC", "ESTIMATED\_COSTS", "DISCONNECT\_PON", "DISCONNECT\_ORDER\_NUM", "DISCONNECT\_ORDER\_TS", "DISCONNECT\_TS", "BILLING\_CODE", "LEASE\_BANDWIDTH", "CONTRACT\_DOC", "NEW", "NEW\_TOTAL", "NEW\_BY\_TESS", "NEW\_NOT\_TESS\_CONTRACT", "NEW\_NOT\_TESS\_NO\_CONTRACT", "DECOM") AS

WITH NEW\_SEGS

AS (SELECT CI.CIRC\_INST\_ID,

CI.CIRC\_HUM\_ID SEGMENT\_NAME,

CI.STATUS,

CI.TYPE SEGMENT\_TYPE,

CI.BANDWIDTH,

CI.VENDOR,

SD.RPT\_ID,

SD.DOMAIN\_INST\_ID,

SD.DOMAIN,

SD.AREA,

SD.REGION,

SD.MARKET,

SD.Z\_SITE\_ID,

SD.SITE\_NAME,

SD.ACTION\_TYPE,

SD.CHG\_TS,

SD.CHG\_BY,

--SD.MRC,

CI.RECUR\_COSTS MRC,

CI.NON\_RECUR\_COSTS ESTIMATED\_COSTS,

SD.DISCONNECT\_PON,

SD.DISCONNECT\_ORDER\_NUM,

SD.DISCONNECT\_ORDER\_TS,

SD.DISCONNECT\_TS,

CASE

WHEN CI.BILLING\_CODE IS NULL THEN SD.BILLING\_CODE

ELSE CI.BILLING\_CODE

END

BILLING\_CODE,

CASE

WHEN ( ( CI.TYPE = 'LEASED SERVICE'

OR ( CI.TYPE != 'LEASED SERVICE'

AND CI.VENDOR NOT LIKE 'VZW%'

AND CI.VENDOR NOT LIKE

'VERIZON WIRELESS%'))

AND ( CI.BANDWIDTH IN

('OC192',

'10 Gbps',

'100 Gbps',

'DARK FIBER')

OR ( CI.BANDWIDTH LIKE '% Mbps'

AND TO\_NUMBER (

SUBSTR (

CI.BANDWIDTH,

1,

INSTR (CI.BANDWIDTH, ' '))) >=

50)))

THEN

1

ELSE

0

END

AS LEASE\_BANDWIDTH

FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DETAIL SD

JOIN

vzwnet.circ\_inst ci

ON SD.CIRC\_INST\_ID = CI.CIRC\_INST\_ID

WHERE SD.ACTION\_TYPE = 'NEW'),

SEG\_CA

AS (SELECT NEW\_SEGS.\*, SCD.ATTR\_VALUE CONTRACT\_DOC

FROM NEW\_SEGS

LEFT OUTER JOIN

(SELECT CAS.\*

FROM VZWNET.CIRC\_ATTR\_SETTINGS CAS

JOIN

VZWNET.VAL\_ATTR\_NAME VAN

ON CAS.VAL\_ATTR\_INST\_ID =

VAN.VAL\_ATTR\_INST\_ID

AND VAN.ATTR\_NAME = 'Contract Documentation') SCD

ON NEW\_SEGS.CIRC\_INST\_ID = SCD.CIRC\_INST\_ID)

SELECT SC."CIRC\_INST\_ID",SC."SEGMENT\_NAME",SC."STATUS",SC."SEGMENT\_TYPE",SC."BANDWIDTH",SC."VENDOR",SC."RPT\_ID",SC."DOMAIN\_INST\_ID",SC."DOMAIN",SC."AREA",SC."REGION",SC."MARKET",SC."Z\_SITE\_ID",SC."SITE\_NAME",SC."ACTION\_TYPE",SC."CHG\_TS",SC."CHG\_BY",SC."MRC",SC."ESTIMATED\_COSTS",SC."DISCONNECT\_PON",SC."DISCONNECT\_ORDER\_NUM",SC."DISCONNECT\_ORDER\_TS",SC."DISCONNECT\_TS",SC."BILLING\_CODE",SC."LEASE\_BANDWIDTH",SC."CONTRACT\_DOC",

1 NEW,

CASE WHEN (LEASE\_BANDWIDTH = 1) THEN 1 ELSE 0 END NEW\_TOTAL,

CASE

WHEN (LEASE\_BANDWIDTH = 1)

AND ( SC.CHG\_BY = 'tessUser'

OR SC.CONTRACT\_DOC LIKE

'https://erpprd-fnprd.erp.vzwcorp.com/psp/ps/EMPLOYEE/ERP/c/CONTRACT\_MGMT.CS\_DOC\_MAINT.GBL%')

THEN

1

ELSE

0

END

NEW\_BY\_TESS,

CASE

WHEN (LEASE\_BANDWIDTH = 1)

AND ( SC.CHG\_BY != 'tessUser'

AND SC.CONTRACT\_DOC NOT LIKE

'https://erpprd-fnprd.erp.vzwcorp.com/psp/ps/EMPLOYEE/ERP/c/CONTRACT\_MGMT.CS\_DOC\_MAINT.GBL%')

THEN

1

ELSE

0

END

NEW\_NOT\_TESS\_CONTRACT,

CASE

WHEN (LEASE\_BANDWIDTH = 1)

AND (SC.CHG\_BY != 'tessUser' AND SC.CONTRACT\_DOC IS NULL)

THEN

1

ELSE

0

END

NEW\_NOT\_TESS\_NO\_CONTRACT,

0 DECOM

FROM SEG\_CA SC

UNION

SELECT SD.CIRC\_INST\_ID,

SD.SEGMENT\_NAME,

SD.STATUS,

SD.SEGMENT\_TYPE,

SD.BANDWIDTH,

SD.VENDOR,

SD.RPT\_ID,

SD.DOMAIN\_INST\_ID,

SD.DOMAIN,

SD.AREA,

SD.REGION,

SD.MARKET,

SD.Z\_SITE\_ID,

SD.SITE\_NAME,

SD.ACTION\_TYPE,

SD.CHG\_TS,

SD.CHG\_BY,

SD.MRC,

null ESTIMATED\_COSTS,

SD.DISCONNECT\_PON,

SD.DISCONNECT\_ORDER\_NUM,

SD.DISCONNECT\_ORDER\_TS,

SD.DISCONNECT\_TS,

SD.BILLING\_CODE,

0 LEASE\_BANDWIDTH,

NULL CONTRACT\_DOC,

0 NEW,

0 NEW\_TOTAL,

0 NEW\_BY\_TESS,

0 NEW\_NOT\_TESS\_CONTRACT,

0 NEW\_NOT\_TESS\_NO\_CONTRACT,

1 DECOM

FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DETAIL SD

WHERE SD.ACTION\_TYPE = 'DECOM'

;

--------------------------------------------------------

-- DDL for View CLLI\_DOMAIN\_MAP\_DETAILED\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP\_DETAILED\_V" ("CLLI", "AREA", "REGION", "LEAF\_DOMAIN\_NAME", "LEAF\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_INST\_ID", "ADDR\_NUMBER", "ADDR\_PREFIX", "ADDR\_STREET\_NAME", "ADDR\_STREET\_TYPE", "UNFORMATTED\_ADDR\_DATA", "CITY", "STATE", "ZIP", "COUNTY\_NAME", "COMMENTS", "LATITUDE\_IN\_DECIMAL\_FORMAT", "LONGITUDE\_IN\_DECIMAL\_FORMAT") AS

SELECT DISTINCT v.clli, dlr.area, dlr.region,

dlr.domain\_name leaf\_domain\_name,

dlr.domain\_inst\_id leaf\_domain\_inst\_id,

dlr.parent\_domain\_inst\_id,

n.addr\_number,

n.addr\_prefix,

n.addr\_street\_name,

n.addr\_street\_type,

n.unformatted\_addr\_data,

n.geo\_name city,

n.GEOPOLITICAL\_CODE state,

n.POSTAL\_CODE zip,

n.COUNTY\_NAME,

v.COMMENTS,

n.LATITUDE\_IN\_DECIMAL\_FORMAT,

n.LONGITUDE\_IN\_DECIMAL\_FORMAT

FROM clli\_domain\_map v

LEFT OUTER JOIN domains\_leaf\_reporting dlr

ON dlr.domain\_inst\_id = v.leaf\_domain\_inst\_id

left outer join clones.NTWK n

on n.GEO\_CODE||n.GEOPOLITICAL\_CODE||n.NETWORK\_SITE\_CODE = v.clli

and n.ADDR\_SEQ\_NUMBER = 1

order by area, region, leaf\_domain\_name, clli

;

--------------------------------------------------------

-- DDL for View CLLI\_DOMAIN\_MAP\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP\_V" ("CLLI", "AREA", "REGION", "MARKET", "SUB\_MARKET", "MARKET\_TERRITORY", "TERRITORY", "LEAF\_DOMAIN\_NAME", "LEAF\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_INST\_ID", "COMMENTS") AS

SELECT DISTINCT v.clli,

dlr.area,

DLR.REGION,

dlr.market,

dlr.sub\_market,

dlr.market\_territory,

DLR.TERRITORY,

dlr.domain\_name leaf\_domain\_name,

dlr.domain\_inst\_id leaf\_domain\_inst\_id,

dlr.parent\_domain\_inst\_id,

v.comments

FROM CLLI\_DOMAIN\_MAP V

LEFT OUTER JOIN DOMAINS\_LEAF\_REPORTING dlr

ON dlr.DOMAIN\_INST\_ID = v.LEAF\_DOMAIN\_INST\_ID

;

--------------------------------------------------------

-- DDL for View CLLI\_DOMAIN\_MAP\_V\_OLD

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP\_V\_OLD" ("CLLI", "AREA", "REGION", "LEAF\_DOMAIN\_NAME", "LEAF\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_INST\_ID", "COMMENTS", "MARKET") AS

SELECT DISTINCT v.clli,

dlr.area,

dlr.region,

dlr.domain\_name leaf\_domain\_name,

dlr.domain\_inst\_id leaf\_domain\_inst\_id,

dlr.parent\_domain\_inst\_id,

v.comments,

dlr.market

FROM CLLI\_DOMAIN\_MAP V

LEFT OUTER JOIN DOMAINS\_LEAF\_REPORTING dlr

ON dlr.DOMAIN\_INST\_ID = v.LEAF\_DOMAIN\_INST\_ID

/\*select distinct substr(vsm\_device\_name, 1,8) clli, area, region, market

, domain\_name leaf\_domain\_name, leaf\_domain\_inst\_id, parent\_domain\_inst\_id

from VSM\_DEVICES\_DOMAIN\_MAP V

JOIN DOMAINS\_LEAF\_REPORTING dlr

on dlr.DOMAIN\_INST\_ID = v.LEAF\_DOMAIN\_INST\_ID

\*/

;

--------------------------------------------------------

-- DDL for View CLLI\_UNQ\_DOMAIN\_ISSUES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CLLI\_UNQ\_DOMAIN\_ISSUES\_V" ("AREA", "REGION", "VSM\_DEVICE\_NAME", "VSM\_CLASS", "AS\_OF\_DATE", "ALLTEL\_DO\_NOT\_REMOVE", "LEAF\_DOMAIN\_INST\_ID") AS

WITH

DUP\_CLLIs as

(

select clli

from clli\_domain\_map\_v cdm

where cdm.AREA = 'NNO'

intersect

select clli

from clli\_domain\_map\_v cdm

where cdm.AREA <> 'NNO'

)

select dlr.area, dlr.region, vddm."VSM\_DEVICE\_NAME",vddm."VSM\_CLASS",vddm."AS\_OF\_DATE",vddm."ALLTEL\_DO\_NOT\_REMOVE",vddm."LEAF\_DOMAIN\_INST\_ID"

from vsm\_devices\_domain\_map vddm

join DUP\_CLLIs c

on substr(vddm.VSM\_DEVICE\_NAME, 1, 6) = substr(c.clli, 1, 6)

join domains\_leaf\_reporting dlr

on dlr.DOMAIN\_INST\_ID = vddm.leaf\_domain\_inst\_id

;

--------------------------------------------------------

-- DDL for View COPY\_WK\_TABLES\_READY\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."COPY\_WK\_TABLES\_READY\_V" ("PROCESS\_ID", "PROCESS\_NAME", "SEQ\_NO", "SCHEMA\_NAME", "WK\_TABLE\_NAME", "PRD\_TABLE\_NAME", "WHERE\_CLAUSE") AS

SELECT a.process\_id,

b.process\_name,

a.seq\_no,

a.schema\_name,

a.wk\_table\_name,

a.prd\_table\_name,

a.where\_clause

FROM copy\_wk\_tables a, xng\_reports.all\_processes b

WHERE a.process\_id = b.process\_id

AND b.RUN\_CRON = 'Y'

and b.EXEC\_STATUS in (1,3)

and b.is\_ready = 'Y'

;

--------------------------------------------------------

-- DDL for View CSR\_CLLI\_ISSUES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."CSR\_CLLI\_ISSUES\_V" ("CSR\_DEVICE\_NAME", "CSR\_CLLI", "VSM\_CLLI", "ISSUE") AS

WITH

CLLI\_6\_MATCH AS

(

select distinct substr(clli, 1, 6) clli\_6

from clli\_domain\_map\_V cdm

)

,

CLLI\_MISMATCH as

(

-- get devices where 6-chars

select ei.csr\_device\_name

, substr(ei.csr\_device\_name, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

from csr\_devices ei

join CLLI\_6\_MATCH cdm6

on cdm6.clli\_6 = substr(ei.csr\_device\_name , 1, 6)

where not exists (select cdm.clli

from clli\_domain\_map cdm

where substr(ei.csr\_device\_name, 1, 8) = cdm.clli )

)

,

BAD\_CLLI as

(

select ei.csr\_device\_name

from csr\_devices ei

minus

select ei.csr\_device\_name

from csr\_devices ei

join clli\_domain\_map\_V cdm

on substr(cdm.clli, 1, 8) = substr(ei.csr\_device\_name , 1, 8)

minus

select ei.csr\_device\_name

from csr\_devices ei

join clli\_domain\_map\_V cdm

on substr(cdm.clli, 1, 6) = substr(ei.csr\_device\_name , 1, 6)

)

select cm.csr\_device\_name,

cm.CSR\_CLLI,cm.VSM\_CLLI, '1st 6-chars of CLLI match. All 8 dont' issue

from CLLI\_MISMATCH cm

union

select bc.csr\_device\_name,

substr(bc.csr\_device\_name, 1, 8), null vsm\_clli, 'Unknown CLLI' issues

from BAD\_CLLI bc

;

--------------------------------------------------------

-- DDL for View DAVID\_VW1

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DAVID\_VW1" ("CIRC\_INST\_ID", "SEGMENT\_ID", "A\_SIDE\_SITE", "A\_SITE\_TYPE", "Z\_SIDE\_SITE", "Z\_SITE\_TYPE", "BANDWIDTH", "STATUS", "TYPE", "VENDOR", "SEGMENT\_BILL\_DATE", "ORDERED", "ORDER\_NUM", "DUE", "SCHEDULED", "IN\_SERVICE", "FOC", "CURRENT\_PATH\_WITH\_REV\_NUM", "NEXT\_PATH\_WITH\_REV\_NUM", "MRC", "ESTIMATED\_COST", "DOMAIN\_NAME", "EBH\_SEGMENT\_ID", "EBH\_STATUS", "EBH\_BANDWIDTH", "EBH\_VENDOR", "EBH\_IN\_SERVICE", "EBH\_MRC", "EBH\_ESTIMATED\_COST") AS

WITH ckts

AS (SELECT ci.circ\_inst\_id,

CI.CIRC\_HUM\_ID segment\_id,

CI.A\_SITE\_ID,

SI1.SITE\_HUM\_ID a\_side\_site,

SI1.NUM a\_site\_type,

CI.Z\_SITE\_ID,

SI2.SITE\_HUM\_ID z\_side\_site,

SI2.NUM z\_site\_type,

CI.BANDWIDTH,

CI.STATUS,

CI.TYPE,

CI.VENDOR,

ci.segment\_bill\_date,

CI.ORDERED,

CI.ORDER\_NUM,

CI.DUE,

CI.SCHED\_DATE scheduled,

CI.IN\_SERVICE,

CI.INSTALLED foc,

p1.pn current\_path\_with\_rev\_num,

p2.pn next\_path\_with\_rev\_num,

CI.RECUR\_COSTS mrc,

CI.NON\_RECUR\_COSTS estimated\_cost,

DI.DOMAIN\_NAME

FROM (SELECT TO\_DATE (CAS1.ATTR\_VALUE, 'DD-MON-YYYY')

segment\_bill\_date,

ci1.\*

FROM vzwnet.circ\_inst ci1

LEFT OUTER JOIN

(SELECT cas.\*

FROM vzwnet.circ\_attr\_settings cas

JOIN

vzwnet.val\_attr\_name van

ON CAS.VAL\_ATTR\_INST\_ID =

VAN.VAL\_ATTR\_INST\_ID

AND VAN.ATTR\_NAME = 'Segment Bill Date') cas1

ON CI1.CIRC\_INST\_ID = cas1.circ\_inst\_id

WHERE CI1.CIRC\_HUM\_ID LIKE '%KFGS%') ci,

vzwnet.site\_inst si1,

(SELECT \*

FROM vzwnet.site\_inst

WHERE num IN ('CELL', 'CELL/FUT')) si2,

(SELECT CPI.CIRC\_PATH\_HUM\_ID || '|' || CPI.CIRC\_PATH\_REV\_NBR

pn,

CPE.SEGMENT\_INST\_ID

FROM vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe,

vzwnet.val\_deploy\_status vds

WHERE CPI.STATUS = VDS.STATUS\_NAME

AND CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

AND VDS.ROOT\_STATUS IN ('A', 'D') ) p1,

(SELECT CPI.CIRC\_PATH\_HUM\_ID || '|' || CPI.CIRC\_PATH\_REV\_NBR

pn,

CPE.SEGMENT\_INST\_ID

FROM vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe,

vzwnet.val\_deploy\_status vds

WHERE CPI.STATUS = VDS.STATUS\_NAME

AND CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

AND VDS.ROOT\_STATUS = 'P') p2,

vzwnet.circ\_domain\_map cdm,

vzwnet.domain\_inst di

WHERE CI.A\_SITE\_ID = SI1.SITE\_INST\_ID(+)

AND CI.Z\_SITE\_ID = SI2.SITE\_INST\_ID(+)

AND CI.BANDWIDTH IN ('1 Gbps', '2 Gps')

AND CI.CIRC\_INST\_ID = p1.segment\_inst\_id(+)

AND CI.CIRC\_INST\_ID = p2.segment\_inst\_id(+)

AND CI.NON\_RECUR\_COSTS IS NOT NULL

AND TO\_NUMBER (CI.NON\_RECUR\_COSTS) = 675

AND CI.CIRC\_INST\_ID = CDM.CIRC\_INST\_ID

AND CDM.DOMAIN\_INST\_ID = DI.DOMAIN\_INST\_ID

AND CI.VENDOR LIKE 'VERIZONCAB-001%'

AND DI.DOMAIN\_NAME IN

('DC\_BALT\_DOMAIN',

'NEW\_ENGLAND\_EAST\_DOMAIN',

'NEW\_ENGLAND\_WEST\_DOMAIN',

'NEW\_JERSEY\_DOMAIN',

'NEW\_YORK\_CITY\_DOMAIN',

'PHILADELPHIA\_DOMAIN',

'UPSTATE\_NY\_DOMAIN',

'VA\_DOMAIN'))

SELECT ckts.circ\_inst\_id,

ckts.segment\_id,

ckts.a\_side\_site,

ckts.a\_site\_type,

ckts.z\_side\_site,

ckts.z\_site\_type,

ckts.BANDWIDTH,

ckts.STATUS,

ckts.TYPE,

ckts.VENDOR,

ckts.segment\_bill\_date,

ckts.ORDERED,

ckts.ORDER\_NUM,

ckts.DUE,

ckts.scheduled,

ckts.IN\_SERVICE,

ckts.foc,

ckts.current\_path\_with\_rev\_num,

ckts.next\_path\_with\_rev\_num,

ckts.mrc,

ckts.estimated\_cost,

ckts.DOMAIN\_NAME,

CI2.CIRC\_HUM\_ID EBH\_SEGMENT\_ID,

CI2.STATUS EBH\_STATUS,

CI2.BANDWIDTH EBH\_BANDWIDTH,

CI2.VENDOR EBH\_VENDOR,

CI2.IN\_SERVICE EBH\_IN\_SERVICE,

CI2.RECUR\_COSTS EBH\_MRC,

CI2.NON\_RECUR\_COSTS EBH\_ESTIMATED\_COST

FROM ckts

LEFT OUTER JOIN

(SELECT \*

FROM vzwnet.circ\_inst

WHERE CIRC\_HUM\_ID LIKE '%KRGS%' AND bandwidth != 'STS1') ci2

ON ckts.z\_site\_id = CI2.Z\_SITE\_ID

;

--------------------------------------------------------

-- DDL for View DAVID\_VW2

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DAVID\_VW2" ("CIRC\_INST\_ID", "SEGMENT\_ID", "A\_SIDE\_SITE", "A\_SITE\_TYPE", "Z\_SIDE\_SITE", "Z\_SITE\_TYPE", "BANDWIDTH", "STATUS", "TYPE", "VENDOR", "SEGMENT\_BILL\_DATE", "ORDERED", "ORDER\_NUM", "DUE", "SCHEDULED", "IN\_SERVICE", "FOC", "CURRENT\_PATH\_WITH\_REV\_NUM", "NEXT\_PATH\_WITH\_REV\_NUM", "MRC", "ESTIMATED\_COST", "DOMAIN\_NAME", "EBH\_SEGMENT\_ID", "EBH\_STATUS", "EBH\_BANDWIDTH", "EBH\_VENDOR", "EBH\_IN\_SERVICE", "EBH\_MRC", "EBH\_ESTIMATED\_COST") AS

WITH ckts

AS (SELECT ci.circ\_inst\_id,

CI.CIRC\_HUM\_ID segment\_id,

CI.A\_SITE\_ID,

SI1.SITE\_HUM\_ID a\_side\_site,

SI1.NUM a\_site\_type,

CI.Z\_SITE\_ID,

SI2.SITE\_HUM\_ID z\_side\_site,

SI2.NUM z\_site\_type,

CI.BANDWIDTH,

CI.STATUS,

CI.TYPE,

CI.VENDOR,

ci.segment\_bill\_date,

CI.ORDERED,

CI.ORDER\_NUM,

CI.DUE,

CI.SCHED\_DATE scheduled,

CI.IN\_SERVICE,

CI.INSTALLED foc,

p1.pn current\_path\_with\_rev\_num,

p2.pn next\_path\_with\_rev\_num,

CI.RECUR\_COSTS mrc,

CI.NON\_RECUR\_COSTS estimated\_cost,

DI.DOMAIN\_NAME

FROM (SELECT TO\_DATE (CAS1.ATTR\_VALUE, 'DD-MON-YYYY')

segment\_bill\_date,

ci1.\*

FROM vzwnet.circ\_inst ci1

LEFT OUTER JOIN

(SELECT cas.\*

FROM vzwnet.circ\_attr\_settings cas

JOIN

vzwnet.val\_attr\_name van

ON CAS.VAL\_ATTR\_INST\_ID =

VAN.VAL\_ATTR\_INST\_ID

AND VAN.ATTR\_NAME = 'Segment Bill Date') cas1

ON CI1.CIRC\_INST\_ID = cas1.circ\_inst\_id

WHERE CI1.CIRC\_HUM\_ID LIKE '%KFGS%') ci,

vzwnet.site\_inst si1,

(SELECT \*

FROM vzwnet.site\_inst

WHERE num IN ('CELL', 'CELL/FUT')) si2,

(SELECT CPI.CIRC\_PATH\_HUM\_ID || '|' || CPI.CIRC\_PATH\_REV\_NBR

pn,

CPE.SEGMENT\_INST\_ID

FROM vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe,

vzwnet.val\_deploy\_status vds

WHERE CPI.STATUS = VDS.STATUS\_NAME

AND CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

/\*AND VDS.ROOT\_STATUS IN ('A', 'D')\*/ ) p1,

(SELECT CPI.CIRC\_PATH\_HUM\_ID || '|' || CPI.CIRC\_PATH\_REV\_NBR

pn,

CPE.SEGMENT\_INST\_ID

FROM vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe,

vzwnet.val\_deploy\_status vds

WHERE CPI.STATUS = VDS.STATUS\_NAME

AND CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

/\*AND VDS.ROOT\_STATUS = 'P'\*/) p2,

vzwnet.circ\_domain\_map cdm,

vzwnet.domain\_inst di

WHERE CI.A\_SITE\_ID = SI1.SITE\_INST\_ID(+)

AND CI.Z\_SITE\_ID = SI2.SITE\_INST\_ID(+)

AND CI.BANDWIDTH IN ('1 Gbps', '2 Gps')

AND CI.CIRC\_INST\_ID = p1.segment\_inst\_id(+)

AND CI.CIRC\_INST\_ID = p2.segment\_inst\_id(+)

AND CI.NON\_RECUR\_COSTS IS NOT NULL

AND TO\_NUMBER (CI.NON\_RECUR\_COSTS) = 675

AND CI.CIRC\_INST\_ID = CDM.CIRC\_INST\_ID

AND CDM.DOMAIN\_INST\_ID = DI.DOMAIN\_INST\_ID

AND CI.VENDOR LIKE 'VERIZONCAB-001%'

AND DI.DOMAIN\_NAME IN

('DC\_BALT\_DOMAIN',

'NEW\_ENGLAND\_EAST\_DOMAIN',

'NEW\_ENGLAND\_WEST\_DOMAIN',

'NEW\_JERSEY\_DOMAIN',

'NEW\_YORK\_CITY\_DOMAIN',

'PHILADELPHIA\_DOMAIN',

'UPSTATE\_NY\_DOMAIN',

'VA\_DOMAIN'))

SELECT ckts.circ\_inst\_id,

ckts.segment\_id,

ckts.a\_side\_site,

ckts.a\_site\_type,

ckts.z\_side\_site,

ckts.z\_site\_type,

ckts.BANDWIDTH,

ckts.STATUS,

ckts.TYPE,

ckts.VENDOR,

ckts.segment\_bill\_date,

ckts.ORDERED,

ckts.ORDER\_NUM,

ckts.DUE,

ckts.scheduled,

ckts.IN\_SERVICE,

ckts.foc,

ckts.current\_path\_with\_rev\_num,

ckts.next\_path\_with\_rev\_num,

ckts.mrc,

ckts.estimated\_cost,

ckts.DOMAIN\_NAME,

CI2.CIRC\_HUM\_ID EBH\_SEGMENT\_ID,

CI2.STATUS EBH\_STATUS,

CI2.BANDWIDTH EBH\_BANDWIDTH,

CI2.VENDOR EBH\_VENDOR,

CI2.IN\_SERVICE EBH\_IN\_SERVICE,

CI2.RECUR\_COSTS EBH\_MRC,

CI2.NON\_RECUR\_COSTS EBH\_ESTIMATED\_COST

FROM ckts

LEFT OUTER JOIN

(SELECT \*

FROM vzwnet.circ\_inst

WHERE CIRC\_HUM\_ID LIKE '%KRGS%' AND bandwidth != 'STS1') ci2

ON ckts.z\_site\_id = CI2.Z\_SITE\_ID

;

--------------------------------------------------------

-- DDL for View DAVID\_VW3

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DAVID\_VW3" ("CIRC\_INST\_ID", "SEGMENT\_ID", "A\_SIDE\_SITE", "A\_SITE\_TYPE", "Z\_SIDE\_SITE", "Z\_SITE\_TYPE", "BANDWIDTH", "STATUS", "TYPE", "VENDOR", "SEGMENT\_BILL\_DATE", "ORDERED", "ORDER\_NUM", "DUE", "SCHEDULED", "IN\_SERVICE", "FOC", "CURRENT\_PATH\_WITH\_REV\_NUM", "NEXT\_PATH\_WITH\_REV\_NUM", "MRC", "ESTIMATED\_COST", "DOMAIN\_NAME", "EBH\_SEGMENT\_ID", "EBH\_STATUS", "EBH\_BANDWIDTH", "EBH\_VENDOR", "EBH\_IN\_SERVICE", "EBH\_MRC", "EBH\_ESTIMATED\_COST") AS

WITH ckts

AS (SELECT ci.circ\_inst\_id,

CI.CIRC\_HUM\_ID segment\_id,

CI.A\_SITE\_ID,

SI1.SITE\_HUM\_ID a\_side\_site,

SI1.NUM a\_site\_type,

CI.Z\_SITE\_ID,

SI2.SITE\_HUM\_ID z\_side\_site,

SI2.NUM z\_site\_type,

CI.BANDWIDTH,

CI.STATUS,

CI.TYPE,

CI.VENDOR,

ci.segment\_bill\_date,

CI.ORDERED,

CI.ORDER\_NUM,

CI.DUE,

CI.SCHED\_DATE scheduled,

CI.IN\_SERVICE,

CI.INSTALLED foc,

p1.pn current\_path\_with\_rev\_num,

p2.pn next\_path\_with\_rev\_num,

CI.RECUR\_COSTS mrc,

CI.NON\_RECUR\_COSTS estimated\_cost,

DI.DOMAIN\_NAME

FROM (SELECT TO\_DATE (CAS1.ATTR\_VALUE, 'DD-MON-YYYY')

segment\_bill\_date,

ci1.\*

FROM vzwnet.circ\_inst ci1

LEFT OUTER JOIN

(SELECT cas.\*

FROM vzwnet.circ\_attr\_settings cas

JOIN

vzwnet.val\_attr\_name van

ON CAS.VAL\_ATTR\_INST\_ID =

VAN.VAL\_ATTR\_INST\_ID

AND VAN.ATTR\_NAME = 'Segment Bill Date') cas1

ON CI1.CIRC\_INST\_ID = cas1.circ\_inst\_id

WHERE CI1.CIRC\_HUM\_ID LIKE '%KFGS%'

and ci1.STATUS not in ('Decommissioned','Pending Disconnect','Cancelled')

) ci,

vzwnet.site\_inst si1,

(SELECT \*

FROM vzwnet.site\_inst

WHERE num IN ('CELL', 'CELL/FUT')) si2,

(SELECT CPI.CIRC\_PATH\_HUM\_ID || '|' || CPI.CIRC\_PATH\_REV\_NBR

pn,

CPE.SEGMENT\_INST\_ID

FROM vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe,

vzwnet.val\_deploy\_status vds

WHERE CPI.STATUS = VDS.STATUS\_NAME

AND CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

AND VDS.ROOT\_STATUS IN ('A', 'D') ) p1,

(SELECT CPI.CIRC\_PATH\_HUM\_ID || '|' || CPI.CIRC\_PATH\_REV\_NBR

pn,

CPE.SEGMENT\_INST\_ID

FROM vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe,

vzwnet.val\_deploy\_status vds

WHERE CPI.STATUS = VDS.STATUS\_NAME

AND CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

AND VDS.ROOT\_STATUS = 'P') p2,

vzwnet.circ\_domain\_map cdm,

vzwnet.domain\_inst di

WHERE CI.A\_SITE\_ID = SI1.SITE\_INST\_ID(+)

AND CI.Z\_SITE\_ID = SI2.SITE\_INST\_ID(+)

AND CI.BANDWIDTH IN ('1 Gbps', '2 Gps')

AND CI.CIRC\_INST\_ID = p1.segment\_inst\_id(+)

AND CI.CIRC\_INST\_ID = p2.segment\_inst\_id(+)

AND CI.NON\_RECUR\_COSTS IS NOT NULL

AND TO\_NUMBER (CI.NON\_RECUR\_COSTS) = 675

AND CI.CIRC\_INST\_ID = CDM.CIRC\_INST\_ID

AND CDM.DOMAIN\_INST\_ID = DI.DOMAIN\_INST\_ID

AND CI.VENDOR LIKE 'VERIZONCAB-001%'

AND DI.DOMAIN\_NAME IN

('DC\_BALT\_DOMAIN',

'NEW\_ENGLAND\_EAST\_DOMAIN',

'NEW\_ENGLAND\_WEST\_DOMAIN',

'NEW\_JERSEY\_DOMAIN',

'NEW\_YORK\_CITY\_DOMAIN',

'PHILADELPHIA\_DOMAIN',

'UPSTATE\_NY\_DOMAIN',

'VA\_DOMAIN'))

SELECT ckts.circ\_inst\_id,

ckts.segment\_id,

ckts.a\_side\_site,

ckts.a\_site\_type,

ckts.z\_side\_site,

ckts.z\_site\_type,

ckts.BANDWIDTH,

ckts.STATUS,

ckts.TYPE,

ckts.VENDOR,

ckts.segment\_bill\_date,

ckts.ORDERED,

ckts.ORDER\_NUM,

ckts.DUE,

ckts.scheduled,

ckts.IN\_SERVICE,

ckts.foc,

ckts.current\_path\_with\_rev\_num,

ckts.next\_path\_with\_rev\_num,

ckts.mrc,

ckts.estimated\_cost,

ckts.DOMAIN\_NAME,

CI2.CIRC\_HUM\_ID EBH\_SEGMENT\_ID,

CI2.STATUS EBH\_STATUS,

CI2.BANDWIDTH EBH\_BANDWIDTH,

CI2.VENDOR EBH\_VENDOR,

CI2.IN\_SERVICE EBH\_IN\_SERVICE,

CI2.RECUR\_COSTS EBH\_MRC,

CI2.NON\_RECUR\_COSTS EBH\_ESTIMATED\_COST

FROM ckts

LEFT OUTER JOIN

(SELECT \*

FROM vzwnet.circ\_inst

WHERE CIRC\_HUM\_ID LIKE '%KRGS%' AND bandwidth != 'STS1') ci2

ON ckts.z\_site\_id = CI2.Z\_SITE\_ID

where segment\_bill\_date is not null

;

--------------------------------------------------------

-- DDL for View DOMAINS\_ALL\_LEAVES

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DOMAINS\_ALL\_LEAVES" ("DOMAIN\_INST\_ID", "DOMAIN\_NAME", "PARENT\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_NAME", "REGION\_DOMAIN\_NAME", "REGION\_DOMAIN\_INST\_ID", "IS\_REGION") AS

WITH leaves AS

(SELECT domain\_inst\_id

FROM vzwnet.domain\_inst

MINUS

-- minus the parents

SELECT parent\_domain\_inst\_id

FROM vzwnet.domain\_inst

)

SELECT distinct di.domain\_inst\_id, di.domain\_name, di.parent\_domain\_inst\_id,

p.domain\_name parent\_domain\_name

, case when vno.REGION\_DOMAIN\_NAME is null then p.domain\_name

else di.domain\_name

end region\_domain\_name

, case when vno.REGION\_DOMAIN\_NAME is null then p.domain\_inst\_id

else di.DOMAIN\_inst\_id

end region\_domain\_inst\_id

, case when vno.REGION\_DOMAIN\_NAME is not null then 'Y' end is\_region

FROM vzwnet.domain\_inst di

join vzwnet.leaves l on l.domain\_inst\_id = di.domain\_inst\_id

join vzwnet.domain\_inst p on p.domain\_inst\_id = di.parent\_domain\_inst\_id

left outer join vzw\_network\_org vno on di.domain\_name = vno.REGION\_DOMAIN\_NAME

;

--------------------------------------------------------

-- DDL for View DOMAINS\_LEAF\_REPORTING

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DOMAINS\_LEAF\_REPORTING" ("DOMAIN\_INST\_ID", "DOMAIN\_NAME", "PARENT\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_NAME", "AREA", "REGION", "MARKET", "SUB\_MARKET", "MARKET\_TERRITORY", "TERRITORY", "REGION\_DOMAIN\_INST\_ID", "REGION\_DOMAIN\_NAME", "IS\_REGION") AS

SELECT DISTINCT d.DOMAIN\_INST\_ID,

d.DOMAIN\_NAME,

d.PARENT\_DOMAIN\_INST\_ID,

d.PARENT\_DOMAIN\_NAME,

vno.area,

vno.region,

vno.market,

vno.sub\_market,

vno.market\_territory,

vno.territory,

d.REGION\_DOMAIN\_INST\_ID,

d.REGION\_DOMAIN\_NAME,

d.IS\_REGION

FROM domains\_all\_leaves d

JOIN xng\_reports.vzw\_network\_org vno

ON vno.market\_domain\_name = domain\_name

WHERE domain\_inst\_id NOT IN (1130, 1128)

;

--------------------------------------------------------

-- DDL for View DOMAINS\_LEAF\_REPORTING\_OLD

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DOMAINS\_LEAF\_REPORTING\_OLD" ("DOMAIN\_INST\_ID", "DOMAIN\_NAME", "PARENT\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_NAME", "AREA", "REGION", "MARKET", "REGION\_DOMAIN\_INST\_ID", "REGION\_DOMAIN\_NAME", "IS\_REGION") AS

SELECT DISTINCT d.DOMAIN\_INST\_ID,

d.DOMAIN\_NAME,

d.PARENT\_DOMAIN\_INST\_ID,

d.PARENT\_DOMAIN\_NAME,

vno.area,

vno.region,

vno.market,

d.REGION\_DOMAIN\_INST\_ID,

d.REGION\_DOMAIN\_NAME,

d.IS\_REGION

FROM domains\_all\_leaves d

JOIN xng\_reports.vzw\_network\_org vno

ON vno.market\_domain\_name = domain\_name

WHERE domain\_inst\_id NOT IN (1210, 1191, 1130, 1150)

-- ALLTEL, ORPHAN, TXSL\_LAB, NWF respectively;

;

--------------------------------------------------------

-- DDL for View DOMAINS\_REGIONAL\_REPORTING

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DOMAINS\_REGIONAL\_REPORTING" ("DOMAIN\_INST\_ID", "DOMAIN\_NAME", "AREA", "REGION", "MARKET\_TERRITORY", "TERRITORY", "SUB\_MARKET") AS

SELECT DISTINCT domain\_inst\_id,

domain\_name,

area,

region,

market\_territory,

territory,

sub\_market

--area, region, di.domain\_inst\_id, domain\_name

FROM vzwnet.domain\_inst di

JOIN vzw\_network\_org vno

ON vno.region\_domain\_name = di.domain\_name

;

--------------------------------------------------------

-- DDL for View DOMAINS\_REGIONAL\_REPORTING\_NEW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DOMAINS\_REGIONAL\_REPORTING\_NEW" ("DOMAIN\_INST\_ID", "DOMAIN\_NAME", "AREA", "REGION", "MARKET\_TERRITORY", "TERRITORY", "SUB\_MARKET") AS

SELECT DISTINCT domain\_inst\_id,

domain\_name,

area,

region,

market\_territory,

territory,

sub\_market

--area, region, di.domain\_inst\_id, domain\_name

FROM vzwnet.domain\_inst di

JOIN vzw\_network\_org\_new vno ON vno.region\_domain\_name = di.domain\_name

;

--------------------------------------------------------

-- DDL for View DOMAINS\_REGIONAL\_REPORTING\_OLD

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."DOMAINS\_REGIONAL\_REPORTING\_OLD" ("DOMAIN\_INST\_ID", "DOMAIN\_NAME", "AREA", "REGION") AS

SELECT DISTINCT domain\_inst\_id,

domain\_name,

area,

region

--area, region, di.domain\_inst\_id, domain\_name

FROM vzwnet.domain\_inst di

JOIN vzw\_network\_org vno ON vno.region\_domain\_name = di.domain\_name

/\*WITH

-- GET HIERARCHY OF DOMAINS

all\_hierarchy AS

(

SELECT di.\*, LEVEL deep

FROM vzwnet.domain\_inst di

START WITH domain\_inst\_id = 1000

CONNECT BY PRIOR domain\_inst\_id = parent\_domain\_inst\_id

)

,

-- get all distinct names of domains

REGIONS as

(

select distinct area, region, vno.region\_domain\_name

from xng\_reports.vzw\_network\_org vno

)

-- get all domaining attributes of all regions. These are all on level = 3

SELECT domain\_inst\_id, domain\_name, area, region

FROM all\_hierarchy a

join REGIONS vno

on vno.region\_domain\_name = a.domain\_name

WHERE a.deep = 3

UNION

-- have to get these 2 separately

--since they are at level = 2 in the domaining hierarchy

SELECT domain\_inst\_id, domain\_name, area, region

FROM all\_hierarchy a

join REGIONS vno

on vno.region\_domain\_name = a.domain\_name

WHERE a.domain\_name IN ('NNO\_DOMAIN', 'OSS\_DOMAIN')

\*/

;

--------------------------------------------------------

-- DDL for View EBH\_TERM\_BTS\_REGIONAL\_SUMMARY

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."EBH\_TERM\_BTS\_REGIONAL\_SUMMARY" ("AREA", "REGION", "LIVE", "RPTD", "DISCOVERED", "LU\_EBH\_COMP", "NT\_EBH\_CDMA\_COMP", "NT\_EBH\_EVDO\_COMP", "MT\_EBH\_CDMA\_COMP", "MT\_EBH\_EVDO\_COMP", "TOTAL\_COMPLIANCE") AS

WITH REGIONS AS (SELECT area, region

FROM domains\_leaf\_reporting dlr

WHERE area <> 'OSS'

UNION

SELECT DISTINCT area, NULL

FROM domains\_leaf\_reporting dlr

UNION

SELECT 'zEnterprise', NULL FROM DUAL

MINUS

SELECT 'OSS', NULL FROM DUAL),

lucent

AS (SELECT CASE

WHEN area IS NULL AND region IS NULL

THEN

'zEnterprise'

ELSE

area

END

area,

region,

NVL (ebh\_CSR\_matched\_live, 0) lu\_ebh\_live,

NVL (ebh\_csr\_reported, 0) lu\_ebh\_reported,

NVL (tot.EBH\_DISCOVERED, 0) lu\_ebh\_discovered,

NVL (ebh\_csr\_comp, 0) lu\_ebh\_comp

FROM lucent\_audit\_reg\_summ tot),

nt\_cdma

AS (SELECT CASE

WHEN area IS NULL AND region IS NULL

THEN

'zEnterprise'

ELSE

area

END

area,

region,

NVL (ebh\_csr\_matched\_live, 0) nt\_ebh\_cdma\_live,

NVL (ebh\_reported, 0) nt\_ebh\_cdma\_reported,

NVL (tot.EBH\_DISCOVERED, 0) nt\_ebh\_cdma\_discovered,

NVL (ebh\_comp, 0) nt\_ebh\_cdma\_comp

FROM nortel\_cdma\_audit\_reg\_summ tot),

nt\_evdo

AS (SELECT CASE

WHEN area IS NULL AND region IS NULL

THEN

'zEnterprise'

ELSE

area

END

area,

region,

NVL (tot\_up\_ebh\_spans\_w\_l\_pathscsr, 0)

nt\_ebh\_evdo\_live,

NVL (ebh\_reported, 0) nt\_ebh\_evdo\_reported,

NVL (tot.TOT\_UP\_EBH\_SPANS, 0) nt\_ebh\_evdo\_discovered,

NVL (ebh\_comp, 0) nt\_ebh\_evdo\_comp

FROM nt\_evdo\_bts\_term\_reg\_summ tot),

mt\_cdma

AS (SELECT CASE

WHEN area IS NULL AND region IS NULL

THEN

'zEnterprise'

ELSE

area

END

area,

region,

NVL (ebh\_matched\_live\_csr, 0) mt\_ebh\_cdma\_live,

NVL (ebh\_reported, 0) mt\_ebh\_cdma\_reported,

NVL (tot.EBH\_DISCOVERED, 0) mt\_ebh\_cdma\_discovered,

NVL (ebh\_comp, 0) mt\_ebh\_cdma\_comp

FROM moto\_cdma\_audit\_reg\_summ tot),

mt\_evdo

AS (SELECT CASE

WHEN area IS NULL AND region IS NULL

THEN

'zEnterprise'

ELSE

area

END

area,

region,

NVL (ebh\_matched\_live\_csr, 0) mt\_ebh\_evdo\_live,

NVL (ebh\_reported, 0) mt\_ebh\_evdo\_reported,

NVL (tot.EBH\_DISCOVERED, 0) mt\_ebh\_evdo\_discovered,

NVL (ebh\_comp, 0) mt\_ebh\_evdo\_comp

FROM moto\_evdo\_audit\_reg\_summ tot)

SELECT dlr.area,

dlr.region,

NVL (lu\_ebh\_live, 0)

+ NVL (nt\_ebh\_cdma\_live, 0)

+ NVL (nt\_ebh\_evdo\_live, 0)

+ NVL (mt\_ebh\_cdma\_live, 0)

+ NVL (mt\_ebh\_evdo\_live, 0)

live,

NVL (lu\_ebh\_reported, 0)

+ NVL (nt\_ebh\_cdma\_reported, 0)

+ NVL (nt\_ebh\_evdo\_reported, 0)

+ NVL (mt\_ebh\_cdma\_reported, 0)

+ NVL (mt\_ebh\_evdo\_reported, 0)

rptd,

NVL (lu\_ebh\_discovered, 0)

+ NVL (nt\_ebh\_cdma\_discovered, 0)

+ NVL (nt\_ebh\_evdo\_discovered, 0)

+ NVL (mt\_ebh\_cdma\_discovered, 0)

+ NVL (mt\_ebh\_evdo\_discovered, 0)

discovered,

NVL (lu\_ebh\_comp, 0) lu\_ebh\_comp,

NVL (nt\_ebh\_cdma\_comp, 0) nt\_ebh\_cdma\_comp,

NVL (nt\_ebh\_evdo\_comp, 0) nt\_ebh\_evdo\_comp,

NVL (mt\_ebh\_cdma\_comp, 0) mt\_ebh\_cdma\_comp,

NVL (mt\_ebh\_evdo\_comp, 0) mt\_ebh\_evdo\_comp,

CASE

WHEN NVL (lu\_ebh\_reported, 0) -- dr is = 0 tot\_comp = null

+ NVL (nt\_ebh\_cdma\_reported, 0)

+ NVL (nt\_ebh\_evdo\_reported, 0)

+ NVL (mt\_ebh\_cdma\_reported, 0)

+ NVL (mt\_ebh\_evdo\_reported, 0) = 0

THEN

0 --null

ELSE

ROUND (

( NVL (lu\_ebh\_live, 0)

+ NVL (nt\_ebh\_cdma\_live, 0)

+ NVL (nt\_ebh\_evdo\_live, 0)

+ NVL (mt\_ebh\_cdma\_live, 0)

+ NVL (mt\_ebh\_evdo\_live, 0))

/ ( NVL (lu\_ebh\_reported, 0)

+ NVL (nt\_ebh\_cdma\_reported, 0)

+ NVL (nt\_ebh\_evdo\_reported, 0)

+ NVL (mt\_ebh\_cdma\_reported, 0)

+ NVL (mt\_ebh\_evdo\_reported, 0))

\* 100,

2

)

END

Total\_compliance

FROM REGIONS dlr

LEFT OUTER JOIN

lucent lu

ON (UPPER (lu.area) = UPPER (dlr.area)

AND UPPER (lu.region) = UPPER (dlr.region)

OR UPPER (lu.area) = UPPER (dlr.area)

AND lu.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

nt\_cdma nt\_c

ON (UPPER (nt\_c.area) = UPPER (dlr.area)

AND UPPER (nt\_c.region) = UPPER (dlr.region)

OR UPPER (nt\_c.area) = UPPER (dlr.area)

AND nt\_c.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

nt\_evdo nt\_e

ON (UPPER (nt\_e.area) = UPPER (dlr.area)

AND UPPER (nt\_e.region) = UPPER (dlr.region)

OR UPPER (nt\_e.area) = UPPER (dlr.area)

AND nt\_e.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

mt\_cdma mt\_c

ON (UPPER (mt\_c.area) = UPPER (dlr.area)

AND UPPER (mt\_c.region) = UPPER (dlr.region)

OR UPPER (mt\_c.area) = UPPER (dlr.area)

AND mt\_c.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

mt\_evdo mt\_e

ON (UPPER (mt\_e.area) = UPPER (dlr.area)

AND UPPER (mt\_e.region) = UPPER (dlr.region)

OR UPPER (mt\_e.area) = UPPER (dlr.area)

AND mt\_e.region IS NULL

AND dlr.region IS NULL)

ORDER BY dlr.area, dlr.region

;

--------------------------------------------------------

-- DDL for View ECP\_CA\_ESM\_DISCOVERY\_ISSUE

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."ECP\_CA\_ESM\_DISCOVERY\_ISSUE" ("SWITCH\_ID", "SYS\_ID", "ECP\_SID", "IN\_DISCOVERY", "IN\_CA", "IN\_ESM") AS

WITH

ALL\_DISCOVERY as

(

select p.SYS\_ID, p.ECP\_SID

from PROS\_ECP\_CELL\_DS1\_MAP p

union

select p.SYS\_ID, p.ECP\_SID

from PROS\_ECP\_CELL\_EBH\_MAP p

)

,

ALL\_CA as

(

select substr(ca.vsm\_device\_name, 1, 6) || substr(ca.vsm\_device\_name, 11, 2) switch\_id, vsm\_device\_name

from cell\_avail\_active ca

where ca.vsm\_device\_name like '%ECP%' or ca.vsm\_device\_name like '%MMC%'

)

,

ALL\_DEV as

(

select case when esm.switch\_id is not null then esm.switch\_id

else p.sys\_id ||'-'||p.ecp\_sid

end switch\_id,

case when esm.SYS\_ID is not null then esm.sys\_id else p.sys\_id end sys\_id

, case when esm.ECP\_SID is not null then esm.ECP\_SID else p.ecp\_sid end ecp\_sid

from ALL\_DISCOVERY p

full outer join ecp\_switch\_map esm

on p.sys\_id = esm.sys\_id

and p.ecp\_sid = esm.ecp\_sid

union

select case when esm.switch\_id is not null then esm.switch\_id

else substr(ca.vsm\_device\_name, 1, 6) || substr(ca.vsm\_device\_name, 11, 2)

end switch\_id, esm.SYS\_ID, esm.ECP\_SID

from ecp\_switch\_map esm

full outer join ALL\_CA ca

on ca.switch\_id = esm.switch\_id

union

select esm.switch\_id, esm.SYS\_ID, esm.ECP\_SID

from ecp\_switch\_map esm

)

SELECT dev.switch\_id

, dev.sys\_id, dev.ecp\_sid

, CASE

WHEN dis.sys\_id IS NULL

THEN 'Missing in discovery'

else 'In discovery'

END in\_discovery,

CASE

WHEN ca.vsm\_device\_name IS NULL

THEN 'Missing in ca'

ELSE ca.vsm\_device\_name

END in\_ca,

CASE

WHEN esm.SWITCH\_ID IS NULL

THEN 'Missing in esm'

ELSE esm.SWITCH\_ID

END in\_esm

FROM ALL\_DEV dev

LEFT OUTER JOIN ALL\_DISCOVERY dis

ON dev.sys\_id = dis.sys\_id

and dev.ecp\_sid = dis.ecp\_sid

LEFT OUTER JOIN ecp\_switch\_map esm

ON dev.switch\_id = esm.SWITCH\_ID

LEFT OUTER JOIN ALL\_CA ca

ON dev.switch\_id = ca.switch\_id

WHERE dev.switch\_id IS NULL

OR ca.vsm\_device\_name IS NULL

or esm.SWITCH\_ID is null

;

--------------------------------------------------------

-- DDL for View EHEALTH\_CSR\_LIST

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."EHEALTH\_CSR\_LIST" ("HOSTNAME", "IP", "EXTRACT\_DATE", "ERROR\_CODE", "VENDOR") AS

SELECT SC.DEVICE\_NAME hostname,

SC.DEVICE\_IP ip,

SC.SYS\_CREATION\_DATE extract\_date,

SC.ERROR\_CODE ERROR\_CODE,

SC.VENDOR vendor

FROM sevone\_csr sc

;

--------------------------------------------------------

-- DDL for View ENB\_EMS\_CLLI\_ISSUES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."ENB\_EMS\_CLLI\_ISSUES\_V" ("AREA", "REGION", "MARKET\_ID", "LTE\_MARKET\_NAME", "ENB\_CLLI", "VSM\_CLLI", "COMMENTS") AS

WITH

eNB\_MKT as

(

select e.MARKET\_ID, e.lte\_market\_name, e.clli

from MARKET\_PREFIX\_ENB\_NAMING e

)

,

CLLI\_6\_MATCH AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region

from clli\_domain\_map\_v cdm

where cdm.REGION not in ('NNO', 'OSS') -- (1012, 1016) --

)

,

CLLI\_MISMATCH as

(

-- get devices where 6-chars

select area, region, ei.MARKET\_ID, ei.lte\_market\_name, ei.clli

, substr(ei.clli, 1, 8) lte\_clli, cdm6.clli\_6 vsm\_clli

from eNB\_MKT ei

join CLLI\_6\_MATCH cdm6

on cdm6.clli\_6 = substr(ei.clli , 1, 6)

where not exists (select cdm.clli

from clli\_domain\_map\_V cdm

where substr(ei.clli, 1, 8) = cdm.clli )

)

,

BAD\_CLLI as

(

select ei.MARKET\_ID, ei.lte\_market\_name, ei.clli

from eNB\_MKT ei

minus

select ei.MARKET\_ID, ei.lte\_market\_name, ei.clli

from eNB\_MKT ei

join clli\_domain\_map\_V cdm

on substr(cdm.clli, 1, 6) = substr(ei.clli , 1, 6)

)

select area, region, ei.MARKET\_ID, ei.lte\_market\_name, ei.clli, ei.VSM\_CLLI, '1st 6-chars of CLLI match. All 8 dont' issue

from CLLI\_MISMATCH ei

union

select null area, null region, ei.MARKET\_ID, ei.lte\_market\_name, ei.clli, null vsm\_clli, 'Unknown CLLI' issues

from BAD\_CLLI ei

;

--------------------------------------------------------

-- DDL for View EQPT\_CLLI\_BY\_LEAF\_DOMAIN

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."EQPT\_CLLI\_BY\_LEAF\_DOMAIN" ("CLLI", "DOMAIN\_INST\_ID", "CLLI\_MAX\_OCCURRENCES\_IN\_DOMAIN") AS

SELECT clli, domain\_inst\_id, clli\_count clli\_max\_occurrences\_in\_domain

FROM (SELECT clli, domain\_inst\_id, clli\_count,

MAX (clli\_count) OVER (PARTITION BY clli) max\_clli\_occurs

FROM (SELECT clli, domain\_inst\_id, COUNT (1) clli\_count

FROM (SELECT SUBSTR (ei.clli, 1, 8) clli,

edm.domain\_inst\_id

FROM vzwnet.equip\_inst ei JOIN vzwnet.equip\_domain\_map edm

ON ei.equip\_inst\_id = edm.equip\_inst\_id

JOIN xng\_reports.domains\_leaf\_reporting dlr

ON edm.domain\_inst\_id =

dlr.domain\_inst\_id

AND dlr.domain\_name NOT LIKE

'%OSS\_%DOMAIN%'

WHERE ei.clli IS NOT NULL

AND REGEXP\_LIKE (ei.clli,

'^[A-Z]{6}[A-Z0-9]{2,}$'

))

GROUP BY clli, domain\_inst\_id))

WHERE clli\_count = max\_clli\_occurs

ORDER BY clli

;

--------------------------------------------------------

-- DDL for View GIGE\_VLAN\_MSPP\_NGMLS\_QT\_MAP\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."GIGE\_VLAN\_MSPP\_NGMLS\_QT\_MAP\_V" ("GIGE\_INST\_ID", "VLAN\_INST\_ID", "VLAN\_NUMBER", "PATH\_C\_NGMLS\_MSPP", "EQUIP\_INST\_ID", "EQUIP\_TYPE", "NE\_TEST\_HEAD\_NAME", "EQ\_TEST\_HEAD\_NAME") AS

WITH gige\_vlan\_vparent\_hasngmls

AS (SELECT DISTINCT

CASE

WHEN CGP.CIRC\_PATH\_INST\_ID IS NULL

THEN

CGP.NEXT\_PATH\_INST\_ID

ELSE

CGP.CIRC\_PATH\_INST\_ID

END

gige\_inst\_id,

GVM.VLAN\_INST\_ID,

GVM.VLAN\_NUMBER,

VP.PARENT\_PATH\_INST\_ID

FROM xng\_csr\_gige\_paths cgp,

xng\_gige\_vlan\_map gvm,

xng\_vlan\_parents vp

WHERE (CGP.CIRC\_PATH\_INST\_ID = GVM.GIGE\_PATH\_INST\_ID

OR CGP.NEXT\_PATH\_INST\_ID = GVM.GIGE\_PATH\_INST\_ID)

AND GVM.VLAN\_INST\_ID = VP.VLAN\_INST\_ID

AND vp.has\_ngmls = 1),

vlan\_ngmls\_qt\_map

AS (SELECT DISTINCT gvh.gige\_inst\_id,

gvh.vlan\_inst\_id,

gvh.vlan\_number --, gvh.parent\_path\_inst\_id

,

XVNG.CIRC\_PATH\_INST\_ID path\_c\_ngmls\_mspp,

XVNG.EQUIP\_INST\_ID,

NGT.EQUIP\_TYPE --, XVNG.CARD\_INST\_ID, XVNG.PORT\_INST\_ID

,

NGT.NE\_TEST\_HEAD\_NAME,

NGT.EQ\_TEST\_HEAD\_NAME

--,XNVA.NGMLS\_DEVICE\_NAME, XNVA.NGMLS\_VENDOR

FROM gige\_vlan\_vparent\_hasngmls gvh,

xng\_vlan\_ngmls\_map xvng,

ngt\_qt\_ne\_vs\_xng\_audit ngt

-- ,ncm\_ngmls\_vlan\_audit xnva

WHERE gvh.parent\_path\_inst\_id = XVNG.CIRC\_PATH\_INST\_ID

AND NGT.EQUIP\_INST\_ID = XVNG.EQUIP\_INST\_ID

AND NGT.EQUIP\_TYPE = 'NGMLS' -- and XNVA.VLAN\_INST\_ID = gvh.vlan\_inst\_id

-- and XNVA.NE\_INST\_ID = xvng.equip\_inst\_id

),

gige\_vlan\_vparent\_hasmspp

AS (SELECT DISTINCT

CASE

WHEN CGP.CIRC\_PATH\_INST\_ID IS NULL

THEN

CGP.NEXT\_PATH\_INST\_ID

ELSE

CGP.CIRC\_PATH\_INST\_ID

END

gige\_inst\_id,

GVM.VLAN\_INST\_ID,

GVM.VLAN\_NUMBER,

VP.PARENT\_PATH\_INST\_ID

FROM xng\_csr\_gige\_paths cgp,

xng\_gige\_vlan\_map gvm,

xng\_vlan\_parents vp

WHERE (CGP.CIRC\_PATH\_INST\_ID = GVM.GIGE\_PATH\_INST\_ID

OR CGP.NEXT\_PATH\_INST\_ID = GVM.GIGE\_PATH\_INST\_ID)

AND GVM.VLAN\_INST\_ID = VP.VLAN\_INST\_ID

AND vp.has\_mspp = 1),

vlan\_mspp\_qt\_map

AS (SELECT DISTINCT gvh.gige\_inst\_id,

gvh.vlan\_inst\_id,

gvh.vlan\_number --, gvh.parent\_path\_inst\_id

,

XVNG.CIRC\_PATH\_INST\_ID path\_c\_ngmls\_mspp,

XVNG.EQUIP\_INST\_ID,

NGT.EQUIP\_TYPE --, XVNG.CARD\_INST\_ID, XVNG.PORT\_INST\_ID

,

NGT.NE\_TEST\_HEAD\_NAME,

NGT.EQ\_TEST\_HEAD\_NAME

--,XNVA.NGMLS\_DEVICE\_NAME, XNVA.NGMLS\_VENDOR

FROM gige\_vlan\_vparent\_hasmspp gvh,

xng\_vlan\_mspp\_map xvng,

ngt\_qt\_ne\_vs\_xng\_audit ngt

-- ,ncm\_ngmls\_vlan\_audit xnva

WHERE gvh.parent\_path\_inst\_id = XVNG.CIRC\_PATH\_INST\_ID

AND NGT.EQUIP\_INST\_ID = XVNG.EQUIP\_INST\_ID

AND NGT.EQUIP\_TYPE = 'MSPP' -- and XNVA.VLAN\_INST\_ID = gvh.vlan\_inst\_id

-- and XNVA.NE\_INST\_ID = xvng.equip\_inst\_id

)

SELECT "GIGE\_INST\_ID",

"VLAN\_INST\_ID",

"VLAN\_NUMBER",

"PATH\_C\_NGMLS\_MSPP",

"EQUIP\_INST\_ID",

"EQUIP\_TYPE",

"NE\_TEST\_HEAD\_NAME",

"EQ\_TEST\_HEAD\_NAME"

FROM vlan\_mspp\_qt\_map

UNION

SELECT "GIGE\_INST\_ID",

"VLAN\_INST\_ID",

"VLAN\_NUMBER",

"PATH\_C\_NGMLS\_MSPP",

"EQUIP\_INST\_ID",

"EQUIP\_TYPE",

"NE\_TEST\_HEAD\_NAME",

"EQ\_TEST\_HEAD\_NAME"

FROM vlan\_ngmls\_qt\_map

;

--------------------------------------------------------

-- DDL for View LUCENT\_BTS\_Z\_SITE\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."LUCENT\_BTS\_Z\_SITE\_V" ("VENDOR", "SYS\_ID", "ECP\_SID", "SWITCH\_ID", "NE\_CELL\_NUM", "NE\_BTS\_NAME", "SITE\_HUM\_ID", "SITE\_INST\_ID", "COUNTY") AS

WITH

ALL\_NE\_BTS as

(

select distinct nexa.SYS\_ID, nexa.ECP\_SID, nexa.switch\_id, nexa.NE\_CELL\_NUM, nexa.NE\_BTS\_NAME

from lucent\_ne\_vs\_xng\_audit nexa

where match\_code <> 'Xng Only'

)

,

ALL\_BTS\_Z\_SIDE\_SITES as

(

--brings all distinct set of all ECP, ne\_bts, z\_site

select distinct nexa.SYS\_ID, nexa.ECP\_SID, nexa.switch\_id, nexa.NE\_CELL\_NUM, nexa.NE\_BTS\_NAME, si.SITE\_HUM\_ID, si.SITE\_INST\_ID, si.county

from lucent\_ne\_vs\_xng\_audit nexa

join vzwnet.circ\_path\_inst cpi on nexa.CIRC\_PATH\_INST\_ID = cpi.CIRC\_PATH\_INST\_ID

join vzwnet.site\_inst si on cpi.Z\_SIDE\_SITE\_ID = si.SITE\_INST\_ID

where match\_code <> 'Xng Only'

)

,

ALL\_NE\_ONLY\_BTS as

(

--brings all distinct set of all ECP, ne\_bts that dont have any paths

select nexa.SYS\_ID, nexa.ECP\_SID, nexa.switch\_id, nexa.NE\_CELL\_NUM, nexa.NE\_BTS\_NAME, null site\_hum\_id, null site\_inst\_id

from ALL\_NE\_BTS nexa

minus

select sys\_id, ecp\_sid, a.switch\_id, ne\_cell\_num, ne\_bts\_name, null, null

from ALL\_BTS\_Z\_SIDE\_SITES a

)

,

GET\_ALL as

(

select 'lucent' vendor, sys\_id, ecp\_sid, switch\_id, ne\_cell\_num, ne\_bts\_name, site\_hum\_id, site\_inst\_id, county

from ALL\_BTS\_Z\_SIDE\_SITES

union

select 'lucent', sys\_id, ecp\_sid, switch\_id, ne\_cell\_num, ne\_bts\_name, null, null, null

from ALL\_NE\_ONLY\_BTS

)

select --"VENDOR","SYS\_ID","ECP\_SID","SWITCH\_ID","NE\_CELL\_NUM","NE\_BTS\_NAME","SITE\_HUM\_ID","SITE\_INST\_ID", county

vendor, sys\_id, ecp\_sid, switch\_id, ne\_cell\_num, ne\_bts\_name, site\_hum\_id, site\_inst\_id, county

from GET\_ALL ;

COMMENT ON TABLE "XNG\_REPORTS"."LUCENT\_BTS\_Z\_SITE\_V" IS 'Maps all ECP-CELL discovered from Prospect to Xng-Site. If one of the many spans to a BTS goes to a site, and others dont, still this view will map the whole BTS to the Xng-Site.'

;

--------------------------------------------------------

-- DDL for View MOTO\_CDMA\_CELL\_Z\_SITE\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MOTO\_CDMA\_CELL\_Z\_SITE\_V" ("VENDOR", "SERVICE", "NE\_OMCR", "NE\_BTS\_NUMBER", "NE\_BTS\_NAME", "SITE\_HUM\_ID", "SITE\_INST\_ID", "COUNTY") AS

WITH

ALL\_NE\_CELL as

(

-- get CDMA only

select distinct NEXA.NE\_OMCR, nexa.NE\_BTS\_NUMBER, blm.BTS\_NAME NE\_BTS\_NAME

from MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT nexa

join MOTO\_BTS\_LOCATION\_MAP blm

on NEXA.NE\_OMCR = blm.vsm\_device\_name\_omcr

and nexa.NE\_BTS\_NUMBER = blm.bts\_number

where match\_code <> 'Xng Only'

)

,

ALL\_CELL\_Z\_SIDE\_SITES as

(

--brings all distinct set of all ECP, ne\_bts, z\_site

select distinct NEXA.NE\_OMCR, nexa.NE\_BTS\_NUMBER, blm.BTS\_NAME ne\_bts\_name, si.SITE\_HUM\_ID, si.SITE\_INST\_ID, si.county

from moto\_cdma\_ne\_vs\_xng\_audit nexa

join MOTO\_BTS\_LOCATION\_MAP blm

on NEXA.NE\_OMCR = blm.vsm\_device\_name\_omcr

and nexa.NE\_BTS\_NUMBER = blm.bts\_number

join vzwnet.circ\_path\_inst cpi on nexa.xng\_PATH\_INST\_ID = cpi.CIRC\_PATH\_INST\_ID

join vzwnet.site\_inst si on cpi.Z\_SIDE\_SITE\_ID = si.SITE\_INST\_ID

where match\_code <> 'Xng Only'

)

,

ALL\_NE\_ONLY\_BTS as

(

--brings all distinct set of all ECP, ne\_bts that dont have any paths

select NEXA.NE\_OMCR, nexa.NE\_BTS\_NUMBER, NE\_BTS\_NAME

from ALL\_NE\_CELL nexa

minus

select A.NE\_OMCR, a.NE\_BTS\_NUMBER, a.NE\_BTS\_NAME

from ALL\_CELL\_Z\_SIDE\_SITES a

)

,

GET\_ALL as

(

select 'motorola' vendor, 'cdma' service, NE\_OMCR, NE\_BTS\_NUMBER, NE\_BTS\_NAME, site\_hum\_id, site\_inst\_id, county

from ALL\_CELL\_Z\_SIDE\_SITES

union

select 'motorola' vendor, 'cdma' service, NE\_OMCR, NE\_BTS\_NUMBER, NE\_BTS\_NAME, null, null, null

from ALL\_NE\_ONLY\_BTS

)

select "VENDOR","SERVICE","NE\_OMCR","NE\_BTS\_NUMBER","NE\_BTS\_NAME","SITE\_HUM\_ID","SITE\_INST\_ID", county

--vsm\_device\_name\_ems, vsm\_device\_name\_bsm, ne\_cell\_number, count(1)

from GET\_ALL ;

COMMENT ON TABLE "XNG\_REPORTS"."MOTO\_CDMA\_CELL\_Z\_SITE\_V" IS 'Maps all OMCR-BTS discovered via SANE to Xng-Site. If one of the many spans to a BTS goes to a site, and others dont, still this view will map the whole BTS to the Xng-Site.'

;

--------------------------------------------------------

-- DDL for View MOTO\_EVDO\_MCCDO\_Z\_SITE\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MOTO\_EVDO\_MCCDO\_Z\_SITE\_V" ("VENDOR", "SERVICE", "AEMS", "NE\_IPBSCDO\_NUMBER", "NE\_MCCDO\_IDENTIFIER", "SITE\_HUM\_ID", "SITE\_INST\_ID", "COUNTY") AS

WITH

ALL\_NE\_CELL as

(

-- get CDMA only

select distinct NEXA.aems, nexa.NE\_ipbscdo\_number, nexa.NE\_mccdo\_identifier

from MOTO\_evdo\_NE\_VS\_XNG\_AUDIT nexa

where match\_code <> 'Xng Only'

)

,

ALL\_CELL\_Z\_SIDE\_SITES as

(

--brings all distinct set of all ECP, ne\_mccdo, z\_site

select distinct NEXA.aems, nexa.NE\_ipbscdo\_number, nexa.NE\_mccdo\_identifier, si.SITE\_HUM\_ID, si.SITE\_INST\_ID, si.county

from moto\_evdo\_ne\_vs\_xng\_audit nexa

join vzwnet.circ\_path\_inst cpi on nexa.xng\_PATH\_INST\_ID = cpi.CIRC\_PATH\_INST\_ID

join vzwnet.site\_inst si on cpi.Z\_SIDE\_SITE\_ID = si.SITE\_INST\_ID

where match\_code <> 'Xng Only'

)

,

ALL\_NE\_ONLY\_MCCDO as

(

--brings all distinct set of all ECP, ne\_mccdo that dont have any paths

select NEXA.aems, nexa.NE\_ipbscdo\_number, nexa.NE\_mccdo\_identifier

from ALL\_NE\_CELL nexa

minus

select A.aems, a.NE\_ipbscdo\_number, a.NE\_mccdo\_identifier

from ALL\_CELL\_Z\_SIDE\_SITES a

)

,

GET\_ALL as

(

select 'motorola' vendor, 'evdo' service, aems, NE\_ipbscdo\_number, NE\_mccdo\_identifier, site\_hum\_id, site\_inst\_id, county

from ALL\_CELL\_Z\_SIDE\_SITES

union

select 'motorola' vendor, 'evdo' service, aems, NE\_ipbscdo\_number, NE\_mccdo\_identifier, null, null, null

from ALL\_NE\_ONLY\_MCCDO

)

select "VENDOR","SERVICE","AEMS","NE\_IPBSCDO\_NUMBER","NE\_MCCDO\_IDENTIFIER","SITE\_HUM\_ID","SITE\_INST\_ID", county

--aems, NE\_ipbscdo\_number, ne\_mccdo\_identifier, count(1)

from GET\_ALL

order by aems, NE\_ipbscdo\_number, ne\_mccdo\_identifier ;

COMMENT ON TABLE "XNG\_REPORTS"."MOTO\_EVDO\_MCCDO\_Z\_SITE\_V" IS 'Maps all IPBSCDO-MCCDO discovered from EMH to Xng-Site. If one of the many spans to a MCCDO goes to a site, and others dont, still this view will map the whole MCCDO to the Xng-Site.'

;

--------------------------------------------------------

-- DDL for View MPLS\_CLLI\_ISSUES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MPLS\_CLLI\_ISSUES\_V" ("ROUTER\_NAME", "CSR\_CLLI", "VSM\_CLLI", "ISSUE") AS

WITH

CLLI\_6\_MATCH AS

(

select distinct substr(clli, 1, 6) clli\_6

from clli\_domain\_map\_V cdm

)

,

CLLI\_MISMATCH as

(

-- get devices where 6-chars

select ei.router\_name

, substr(ei.router\_name, 1, 8) router\_clli, cdm6.clli\_6 vsm\_clli

from ana\_mpls\_rtrs ei

join CLLI\_6\_MATCH cdm6

on cdm6.clli\_6 = substr(ei.router\_name , 1, 6)

where not exists (select cdm.clli

from clli\_domain\_map\_V cdm

where substr(ei.router\_name, 1, 8) = cdm.clli )

)

,

BAD\_CLLI as

(

select ei.router\_name

from ana\_mpls\_rtrs ei

minus

select ei.router\_name

from ana\_mpls\_rtrs ei

join clli\_domain\_map\_V cdm

on substr(cdm.clli, 1, 6) = substr(ei.router\_name , 1, 6)

)

select cm.router\_name,cm.router\_CLLI,cm.VSM\_CLLI, '1st 6-chars of CLLI match. All 8 dont' issue

from CLLI\_MISMATCH cm

union

select bc.router\_name, substr(bc.router\_name, 1, 8), null vsm\_clli, 'Unknown CLLI' issues

from BAD\_CLLI bc

;

--------------------------------------------------------

-- DDL for View MSPP\_CIRC\_PATH\_PORTS\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MSPP\_CIRC\_PATH\_PORTS\_V" ("EQUIP\_INST\_ID", "PORT\_INST\_ID", "PORT\_HUM\_ID", "DESCR", "EQUIP\_STATUS", "CATEGORY", "DEVICE\_XCONN\_ELEMENTS", "AUDIT\_STATUS", "CIRC\_PATH\_INST\_ID", "LEG\_INST\_ID", "SEQUENCE", "MEMBER\_NBR", "CIRC\_PATH\_HUM\_ID", "BANDWIDTH", "PATH\_STATUS", "NEXT\_PATH\_INST\_ID", "PREV\_PATH\_INST\_ID") AS

SELECT DISTINCT xd.xng\_equip\_inst\_id equip\_inst\_id,

p.port\_inst\_id,

p.port\_hum\_id,

ei.descr,

ei.status equip\_status,

ei.TYPE CATEGORY,

xd.device\_xconn\_elements,

xd.audit\_status,

cpe.circ\_path\_inst\_id,

li.leg\_inst\_id,

cpe.sequence,

cpe.member\_nbr,

cpi.circ\_path\_hum\_id,

cpi.bandwidth,

cpi.status path\_status,

cpi.next\_path\_inst\_id,

cpi.prev\_path\_inst\_id

FROM (SELECT \*

FROM xng\_reports.inc\_mspp\_xconn\_audit\_details

WHERE xng\_circ\_path\_inst\_id > 0) xd

JOIN xng\_reports.mspp\_circ\_path\_elements pe

ON xd.xng\_circ\_path\_inst\_id = pe.circ\_path\_inst\_id

AND xd.member\_nbr = NVL (pe.member\_nbr, 0)

JOIN vzwnet.circ\_path\_element cpe

ON xd.xng\_circ\_path\_inst\_id = cpe.circ\_path\_inst\_id

JOIN vzwnet.path\_leg\_inst li

ON cpe.circ\_path\_inst\_id = li.circ\_path\_inst\_id

JOIN (SELECT \*

FROM vzwnet.circ\_path\_inst cpi2

WHERE cpi2.status != 'Decommissioned'

AND cpi2.status != 'Cancelled'

AND cpi2.status != 'INVALID DECOMMISSION') cpi

ON cpe.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

JOIN vzwnet.epa p

ON cpe.port\_inst\_id = p.port\_inst\_id

JOIN (SELECT \*

FROM vzwnet.equip\_inst ei2

WHERE ei2.TYPE = 'MSPP') ei

ON p.equip\_inst\_id = ei.equip\_inst\_id

WHERE xd.device\_xconn\_elements LIKE

REGEXP\_REPLACE (p.port\_hum\_id, '-0+', '-') || ',%'

OR xd.device\_xconn\_elements LIKE

'%,' || REGEXP\_REPLACE (p.port\_hum\_id, '-0+', '-')

OR xd.device\_xconn\_elements LIKE

REGEXP\_REPLACE (p.port\_hum\_id, '-0+', '-')

;

--------------------------------------------------------

-- DDL for View MSPP\_CLLI\_ISSUES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MSPP\_CLLI\_ISSUES\_V" ("TID", "TID\_CLLI", "VSM\_CLLI", "ISSUE") AS

WITH

CLLI\_6\_MATCH AS

(

select distinct substr(clli, 1, 6) clli\_6

from clli\_domain\_map\_V cdm

)

,

CLLI\_MISMATCH as

(

-- get devices where 6-chars

select ei.tid

, substr(ei.tid, 1, 8) tid\_clli, cdm6.clli\_6 vsm\_clli

from inc\_devices ei

join CLLI\_6\_MATCH cdm6

on cdm6.clli\_6 = substr(ei.tid , 1, 6)

where regexp\_like(ei.ne\_type, 'ALU 167[58]')

and not exists (select cdm.clli

from clli\_domain\_map\_V cdm

where substr(ei.tid, 1, 8) = cdm.clli )

)

,

BAD\_CLLI as

(

select ei.tid

from inc\_devices ei

where regexp\_like(ei.ne\_type, 'ALU 167[58]')

minus

select ei.tid

from inc\_devices ei

join clli\_domain\_map\_V cdm

on substr(cdm.clli, 1, 6) = substr(ei.tid , 1, 6)

)

select cm.tid,cm.tid\_CLLI,cm.VSM\_CLLI, '1st 6-chars of CLLI match. All 8 dont' issue

from CLLI\_MISMATCH cm

union

select bc.tid, substr(bc.tid, 1, 8), null vsm\_clli, 'Unknown CLLI' issues

from BAD\_CLLI bc

;

--------------------------------------------------------

-- DDL for View MSPP\_DEVICES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MSPP\_DEVICES\_V" ("EQ\_CLASS\_TYPE", "EQ\_CLASS", "TYPE", "STATUS", "ACCESS\_ID", "NE\_ACCESS\_ID", "INST\_ID", "PARENT\_EQ\_INST\_ID", "NE\_INST\_ID", "VENDOR", "MODEL", "DESCR", "CLLI", "SITE\_INST\_ID") AS

SELECT DECODE (LEVEL,

1, 'NE',

DECODE (e.eq\_class, 'S', 'SHELF', 'N/A')

) eq\_class\_type,

e.eq\_class, e.TYPE, e.status, e.target\_id access\_id,

TRIM

(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH (DECODE (LEVEL,

1, e.target\_id,

''

),

'/'

)

) ne\_access\_id,

e.equip\_inst\_id inst\_id, e.parent\_eq\_inst\_id,

TO\_NUMBER

(TRIM

(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH

(DECODE (LEVEL,

1, e.equip\_inst\_id,

''

),

'/'

)

)

) ne\_inst\_id,

e.vendor, e.model, e.descr, e.clli, e.site\_inst\_id

FROM vzwnet.equip\_inst e

WHERE (LEVEL = 1 OR e.eq\_class = 'S')

START WITH e.TYPE = 'MSPP'

AND (REGEXP\_LIKE (descr, '167[85]')

OR REGEXP\_LIKE (model, '167[85]')

)

-- AND ( e.parent\_eq\_inst\_id IS NULL or e.parent\_eq\_inst\_id in (select ei.equip\_inst\_id from vzwnet.equip\_inst ei where ei.type = 'MSC') )

AND ( e.eq\_class = 'C'

OR ( e.eq\_class = 'S'

AND ( e.parent\_eq\_inst\_id IS NULL

OR e.parent\_eq\_inst\_id IN (SELECT ei.equip\_inst\_id

FROM vzwnet.equip\_inst ei

WHERE ei.TYPE = 'MSC')

)

)

)

CONNECT BY PRIOR e.equip\_inst\_id = e.parent\_eq\_inst\_id

;

--------------------------------------------------------

-- DDL for View MSPP\_FM\_NODE\_ID\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MSPP\_FM\_NODE\_ID\_V" ("SITE\_INST\_ID", "EQUIP\_INST\_ID", "NE\_INST\_ID", "EQ\_CLASS", "STATUS", "TYPE", "VENDOR", "DESCR", "MODEL", "NODE\_ID") AS

SELECT distinct e.site\_inst\_id, e.inst\_id equip\_inst\_id, e.ne\_inst\_id, e.eq\_class,

e.status, e.TYPE, e.vendor, e.descr, e.model,

uda.attr\_value node\_id

--, drr.area, drr.region

FROM vzwnet.val\_attr\_name van JOIN vzwnet.equip\_attr\_settings uda

ON van.group\_name = 'Fault Management'

AND van.attr\_name = 'Node Id'

AND van.val\_attr\_inst\_id = uda.val\_attr\_inst\_id

RIGHT OUTER JOIN xng\_reports.mspp\_devices\_v e

ON uda.equip\_inst\_id = e.inst\_id

--JOIN vzwnet.equip\_domain\_map edm ON e.inst\_id = edm.equip\_inst\_id

--JOIN xng\_reports.domains\_leaf\_reporting drr

--ON edm.domain\_inst\_id = drr.domain\_inst\_id

WHERE TYPE = 'MSPP'

;

--------------------------------------------------------

-- DDL for View MSPP\_SUMMARY\_REGION\_MAP\_V\_DEL

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP\_V\_DEL" ("AREA", "REGION", "LEAF\_DOMAIN\_NAME", "LEAF\_DOMAIN\_INST\_ID", "XNG\_SITE\_INST\_ID", "XNG\_EQUIP\_INST\_ID", "TID", "XNG\_DEVICE\_TYPE", "AUDIT\_STATUS", "AUDIT\_DATE", "MATCHED\_XCONN\_COUNT", "DISCREPANCY\_XCONN\_COUNT", "TOTAL\_XCONN\_COUNT", "TOTAL\_CIRC\_PATH\_COUNT", "MATCHED\_CIRC\_PATH\_COUNT", "DESCRIPTION", "STATUS", "CARDTYPE", "IP\_ADDRESS", "EQUIP\_STATUS", "LIVE\_COUNT", "NOT\_LIVE\_COUNT", "NAMING\_STANDARD") AS

select "AREA","REGION","LEAF\_DOMAIN\_NAME","LEAF\_DOMAIN\_INST\_ID","XNG\_SITE\_INST\_ID","XNG\_EQUIP\_INST\_ID","TID","XNG\_DEVICE\_TYPE","AUDIT\_STATUS","AUDIT\_DATE","MATCHED\_XCONN\_COUNT","DISCREPANCY\_XCONN\_COUNT","TOTAL\_XCONN\_COUNT","TOTAL\_CIRC\_PATH\_COUNT","MATCHED\_CIRC\_PATH\_COUNT","DESCRIPTION","STATUS","CARDTYPE","IP\_ADDRESS","EQUIP\_STATUS","LIVE\_COUNT","NOT\_LIVE\_COUNT","NAMING\_STANDARD" from XNG\_REPORTS.MSPP\_SUMMARY\_REGION\_MAP

;

--------------------------------------------------------

-- DDL for View MSPP\_TEST\_HEAD\_V\_DEL

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MSPP\_TEST\_HEAD\_V\_DEL" ("VAL\_ATTR\_INST\_ID", "XNG\_TEST\_HEAD\_NAME", "EQUIP\_INST\_ID", "DESCR") AS

(SELECT van.val\_attr\_inst\_id, eas.attr\_value xng\_test\_head\_name,

eas.equip\_inst\_id, eq.descr --, si.site\_hum\_id

FROM vzwnet.val\_attr\_name van,

vzwnet.equip\_attr\_settings eas,

(SELECT ei.descr, ei.equip\_inst\_id

FROM vzwnet.equip\_inst ei

WHERE ei.TYPE = 'MSPP') eq

WHERE van.group\_name = 'Remote Test Units'

AND van.attr\_name = 'Test Head Name'

AND eas.val\_attr\_inst\_id = van.val\_attr\_inst\_id

AND eq.equip\_inst\_id = eas.equip\_inst\_id)

;

--------------------------------------------------------

-- DDL for View MSPP\_VSM\_MATCH\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MSPP\_VSM\_MATCH\_V" ("VSM\_NODE\_ID", "TID", "NETYPE", "LEAF\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_INST\_ID", "AREA", "REGION") AS

SELECT vvxx.vsm\_device\_name AS vsm\_node\_id, ne.tid, ne.netype,

vvxx.leaf\_domain\_inst\_id, vvxx.parent\_domain\_inst\_id, vvxx.area,

vvxx.region

FROM xng\_reports.inc\_mspp\_ne ne JOIN xng\_reports.vsm\_devices\_ipaddresses vdi

ON TRIM (ne.ip\_address) =

TRIM (SUBSTR (vdi.vsm\_ip,

1,

DECODE (INSTR (vdi.vsm\_ip, ' '),

0, LENGTH (vdi.vsm\_ip),

INSTR (vdi.vsm\_ip, ' ')

)

)

)

JOIN xng\_reports.inc\_mspp\_xconn\_audit\_summary xa

ON ne.tid = xa.tid AND NVL (xa.status, 'ENABLED') = 'ENABLED'

JOIN xng\_reports.vzw\_vsm\_xng\_xref\_v vvxx

ON vdi.vsm\_device\_name = vvxx.vsm\_device\_name

WHERE REGEXP\_LIKE (ne.netype, '167[85]')

AND vvxx.vsm\_class LIKE '%MSPP'

AND vvxx.vsm\_device\_name NOT LIKE '%CNFW%'

AND ( ( ne.netype LIKE '%1675%'

AND vvxx.vsm\_device\_name LIKE '%LUCENT\_SONET%'

)

OR (ne.netype LIKE '%1678%' AND vvxx.vsm\_device\_name LIKE '%1678%'

)

)

;

--------------------------------------------------------

-- DDL for View MSPP\_VSM\_MATCH\_V\_DEL

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MSPP\_VSM\_MATCH\_V\_DEL" ("VSM\_NODE\_ID", "TID", "NETYPE", "CARDTYPE", "LEAF\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_INST\_ID", "AREA", "REGION") AS

SELECT "VSM\_NODE\_ID", "TID", "NETYPE", "CARDTYPE", "LEAF\_DOMAIN\_INST\_ID",

"PARENT\_DOMAIN\_INST\_ID", "AREA", "REGION"

FROM mspp\_vsm\_match

;

--------------------------------------------------------

-- DDL for View MTX\_BMM\_VDDM\_ISSUE

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."MTX\_BMM\_VDDM\_ISSUE" ("VSM\_DEVICE\_NAME\_MTX", "IN\_BMM", "IN\_VDDM") AS

WITH

ALL\_DEVICE as

(

SELECT vsm\_device\_name\_mtx

FROM bsm\_mtx\_map

WHERE mtx\_status = 'Live'

UNION

SELECT vsm\_device\_name

FROM vsm\_devices\_domain\_map vddm

WHERE vddm.vsm\_class IN ('ntDmsMsc', 'MOTOROLA\_EMX')

)

select dev.vsm\_device\_name\_mtx

, case when bmm.vsm\_device\_name\_mtx is null then 'Missing in mbb' else bmm.vsm\_device\_name\_mtx end in\_bmm

, case when vddm.vsm\_device\_name is null then 'Missing in vddm' else vddm.vsm\_device\_name end in\_vddm

from ALL\_DEVICE dev

left outer join bsm\_mtx\_map bmm

on dev.VSM\_device\_NAME\_mtx = bmm.vsm\_device\_name\_mtx

left outer join vsm\_devices\_domain\_map vddm

on dev.VSM\_device\_NAME\_mtx = vddm.vsm\_device\_name

where dev.vsm\_device\_name\_mtx is null

or vddm.vsm\_device\_name is null

;

--------------------------------------------------------

-- DDL for View N2N\_OVERALL\_COMPLIANCE\_SUMM

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."N2N\_OVERALL\_COMPLIANCE\_SUMM" ("AREA", "REGION", "LIVE", "RPTD", "ENB\_TOTAL\_COMPLIANCE", "E2BTS\_TOTAL\_COMPLIANCE", "EBH\_TOTAL\_COMPLIANCE", "T1\_TOTAL\_COMPLIANCE", "CSR\_TOTAL\_COMPLIANCE", "VLAN\_TOTAL\_COMPLIANCE", "OVERALL\_COMPLIANCE", "REGIONAL\_MONTHLY\_DELTA") AS

WITH regions

AS (SELECT area, region

FROM domains\_leaf\_reporting dlr

WHERE area NOT IN ('OSS', 'NNO')

UNION

SELECT DISTINCT area, NULL

FROM domains\_leaf\_reporting dlr

WHERE area <> 'NNO' AND area <> 'OSS'

UNION

SELECT 'unknown', 'unknown' FROM DUAL

UNION

SELECT 'unknown', NULL FROM DUAL

UNION

SELECT 'zEnterprise', NULL FROM DUAL),

t1

AS (SELECT CASE

WHEN area IS NULL --= 'Grand Total'

AND region IS NULL THEN 'zEnterprise'

ELSE DECODE (area, 'UnMapped', 'unknown', area)

END

area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL (t1.active\_live\_spans\_testable, 0) live,

NVL (t1.active\_reported, 0) rptd,

t1.percentage\_t1\_testable testability\_compliance

FROM xng\_reports.switch\_path\_t1\_summary t1),

enb

AS (SELECT CASE

WHEN area IS NULL AND region IS NULL THEN 'zEnterprise'

ELSE area

END

area,

region,

NVL (e.live, 0) live,

NVL (e.enb\_reported, 0) enb\_reported,

e.total\_compliance

FROM enb\_summary\_by\_region e),

e2bts

AS (SELECT CASE

WHEN area = 'Grand Total' AND region IS NULL

THEN

'zEnterprise' --area IS NULL

ELSE

DECODE (area, 'UnMapped', 'unknown', area)

END

area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL (e2bts.live, 0) live,

NVL (e2bts.rptd, 0) rptd,

e2bts.total\_compliance

-- NVL (e2BTS.TOTAL\_COMPLIANCE, 0) TOTAL\_COMPLIANCE

FROM xng\_reports.ebh\_term\_bts\_regional\_summary e2bts),

ebh

AS (SELECT CASE

WHEN area IS NULL AND region IS NULL THEN 'zEnterprise' --area IS NULL

ELSE DECODE (area, 'UnMapped', 'unknown', area)

END

area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL (ebh.live\_csr\_w\_ebh\_pathset, 0) live,

NVL (ebh.den\_for\_compliance\_per, 0) rptd,

ebh.compliance

FROM xng\_reports.comp\_ebh\_summary\_by\_region ebh),

csr

AS (SELECT CASE

WHEN area IS NULL AND region IS NULL THEN 'zEnterprise'

ELSE DECODE (area, 'UnMapped', 'unknown', area)

END

area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL (csr.BOTH\_LIVE\_CSR, 0) live,

NVL (csr.den\_for\_comper, 0) rptd,

csr.compliance

FROM xng\_reports.CSR\_SUMMARY\_BY\_REGION csr),

vlan

AS (select area, region, sum(live) live, sum(rptd) rptd,

round(avg(overall\_compliance), 2) vlan\_compliance

from VLAN\_SUMM\_ALL

group by area, region),

-- NVL (hpov.tot\_csr\_matched\_per\_region, 0) live,

--NVL (

-- hpov.tot\_xo\_live\_csr\_per\_region

-- + hpov.tot\_csr\_matched\_per\_region,

-- 0)

-- rptd,

-- ROUND (

-- ( hpov.tot\_csr\_matched\_per\_region

-- / ( hpov.tot\_csr\_matched\_per\_region

---- + hpov.tot\_xo\_live\_csr\_per\_region))

-- \* 100,

-- 2)

-- compliance

--FROM xng\_reports.csr\_missing\_in\_hpov hpov),

calc\_summ

AS (SELECT dlr.area,

dlr.region,

NVL (e.live, 0)

+ NVL (e2b.live, 0)

+ NVL (ebh.live, 0)

+ NVL (t.live, 0)

+ NVL (c.live, 0)

live,

NVL (e.enb\_reported, 0)

+ NVL (e2b.rptd, 0)

+ NVL (ebh.rptd, 0)

+ NVL (t.rptd, 0)

+ NVL (c.rptd, 0)

rptd,

NVL (e.total\_compliance, 0) enb\_total\_compliance,

NVL (e2b.total\_compliance, 0) e2bts\_total\_compliance,

NVL (ebh.compliance, 0) ebh\_total\_compliance,

NVL (t.testability\_compliance, 0) t1\_total\_compliance,

NVL (c.compliance, 0) csr\_total\_compliance,

NVL (v.vlan\_compliance, 0) vlan\_total\_compliance,

ROUND (

( ( NVL (e.live, 0)

+ NVL (e2b.live, 0)

+ NVL (ebh.live, 0)

+ NVL (t.live, 0)

+ NVL (c.live, 0))

/ ( NVL (e.enb\_reported, 0)

+ NVL (e2b.rptd, 0)

+ NVL (ebh.rptd, 0)

+ NVL (t.rptd, 0)

+ NVL (c.rptd, 0)))

\* 100,

2)

overall\_compliance

FROM regions dlr

LEFT OUTER JOIN

enb e

ON ( UPPER (e.area) = UPPER (dlr.area)

AND UPPER (e.region) = UPPER (dlr.region)

OR UPPER (e.area) = UPPER (dlr.area)

AND e.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

e2bts e2b

ON ( UPPER (e2b.area) = UPPER (dlr.area)

AND UPPER (e2b.region) = UPPER (dlr.region)

OR UPPER (e2b.area) = UPPER (dlr.area)

AND e2b.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

ebh ebh

ON ( UPPER (ebh.area) = UPPER (dlr.area)

AND UPPER (ebh.region) = UPPER (dlr.region)

OR UPPER (ebh.area) = UPPER (dlr.area)

AND ebh.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

t1 t

ON ( UPPER (t.area) = UPPER (dlr.area)

AND UPPER (t.region) = UPPER (dlr.region)

OR UPPER (t.area) = UPPER (dlr.area)

AND t.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

csr c

ON ( UPPER (c.area) = UPPER (dlr.area)

AND UPPER (c.region) = UPPER (dlr.region)

OR UPPER (c.area) = UPPER (dlr.area)

AND c.region IS NULL

AND dlr.region IS NULL)

LEFT OUTER JOIN

vlan v

ON ( UPPER (v.area) = UPPER (dlr.area)

AND UPPER (v.region) = UPPER (dlr.region)

OR UPPER (v.area) = UPPER (dlr.area)

AND v.region IS NULL

AND dlr.region IS NULL)),

calc\_deltas

AS ( -- to calculate regional delta

SELECT dlr.area,

dlr.region,

dlr.live,

dlr.rptd,

dlr.enb\_total\_compliance,

dlr.e2bts\_total\_compliance,

dlr.ebh\_total\_compliance,

dlr.t1\_total\_compliance,

dlr.csr\_total\_compliance,

dlr.vlan\_total\_compliance,

dlr.overall\_compliance,

NVL (dlr.overall\_compliance, 0)

- NVL (prev.overall\_compliance, 0)

regional\_monthly\_delta

FROM calc\_summ dlr

JOIN n2n\_overall\_summ\_prev\_month prev

ON prev.region = dlr.region

UNION

-- to calculate regional delta

SELECT dlr.area,

dlr.region,

dlr.live,

dlr.rptd,

dlr.enb\_total\_compliance,

dlr.e2bts\_total\_compliance,

dlr.ebh\_total\_compliance,

dlr.t1\_total\_compliance,

dlr.csr\_total\_compliance,

dlr.vlan\_total\_compliance,

dlr.overall\_compliance,

NVL (dlr.overall\_compliance, 0)

- NVL (prev.overall\_compliance, 0)

regional\_monthly\_delta

FROM calc\_summ dlr

JOIN

n2n\_overall\_summ\_prev\_month prev

ON prev.area = dlr.area

AND prev.region IS NULL

AND dlr.region IS NULL --calculate column deltas

UNION

SELECT 'zMetricDelta' area,

NULL region,

dlr.live - prev.live live,

dlr.rptd - prev.rptd rptd,

dlr.enb\_total\_compliance

- NVL (prev.enb\_total\_compliance, 0),

dlr.e2bts\_total\_compliance

- NVL (prev.e2bts\_total\_compliance, 0),

dlr.ebh\_total\_compliance

- NVL (prev.ebh\_total\_compliance, 0),

dlr.t1\_total\_compliance

- NVL (prev.t1\_total\_compliance, 0),

dlr.csr\_total\_compliance

- NVL (prev.csr\_total\_compliance, 0),

dlr.vlan\_total\_compliance

- NVL (prev.vlan\_total\_compliance, 0),

dlr.overall\_compliance - NVL (prev.overall\_compliance, 0),

NULL regional\_monthly\_delta

FROM calc\_summ dlr

JOIN n2n\_overall\_summ\_prev\_month prev

ON prev.area = dlr.area AND prev.area = 'zEnterprise')

SELECT "AREA",

"REGION",

"LIVE",

"RPTD",

"ENB\_TOTAL\_COMPLIANCE",

"E2BTS\_TOTAL\_COMPLIANCE",

"EBH\_TOTAL\_COMPLIANCE",

"T1\_TOTAL\_COMPLIANCE",

"CSR\_TOTAL\_COMPLIANCE",

"VLAN\_TOTAL\_COMPLIANCE",

"OVERALL\_COMPLIANCE",

"REGIONAL\_MONTHLY\_DELTA"

FROM calc\_deltas dlr

ORDER BY dlr.area, dlr.region

;

--------------------------------------------------------

-- DDL for View N2N\_OVERALL\_COMPLIANCE\_SUMM\_D

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."N2N\_OVERALL\_COMPLIANCE\_SUMM\_D" ("AREA", "REGION", "LIVE", "RPTD", "ENB\_TOTAL\_COMPLIANCE", "MSPP\_TOTAL\_COMPLIANCE", "E2BTS\_TOTAL\_COMPLIANCE", "CSR\_TOTAL\_COMPLIANCE", "T1\_TOTAL\_COMPLIANCE", "HPOV\_TOTAL\_COMPLIANCE", "OVERALL\_COMPLIANCE", "REGIONAL\_MONTHLY\_DELTA") AS

WITH regions AS

(SELECT area, region

FROM domains\_leaf\_reporting dlr

WHERE area NOT IN ('OSS', 'NNO')

UNION

SELECT DISTINCT area, NULL

FROM domains\_leaf\_reporting dlr

WHERE area <> 'NNO' AND area <> 'OSS'

UNION

SELECT 'unknown', 'unknown'

FROM DUAL

UNION

SELECT 'unknown', NULL

FROM DUAL

UNION

SELECT 'zEnterprise', NULL

FROM DUAL),

t1 AS

(SELECT CASE

WHEN area IS NULL --= 'Grand Total'

AND region IS NULL

THEN 'zEnterprise'

ELSE DECODE (area, 'UnMapped', 'unknown', area)

END area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL (t1.active\_live\_spans\_testable, 0) live,

NVL (t1.active\_reported, 0) rptd,

t1.percentage\_t1\_testable testability\_compliance

FROM xng\_reports.switch\_path\_t1\_summary t1),

enb AS

(SELECT CASE

WHEN area IS NULL AND region IS NULL

THEN 'zEnterprise'

ELSE area

END area,

region, NVL (e.live, 0) live,

NVL (e.enb\_reported, 0) enb\_reported, e.total\_compliance

FROM enb\_summary\_by\_region e),

mspp AS

(SELECT CASE

WHEN area = 'Grand Total' AND region IS NULL

THEN 'zEnterprise' --area IS NULL

ELSE DECODE (area, 'UnMapped', 'unknown', area)

END area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL

(mspp.live\_xconnect\_match\_count,

0

) live\_xconnect\_match\_count,

NVL (mspp.total\_for\_mspp\_reported, 0) total\_mspp\_for\_reported,

mspp.total\_compliance

FROM inc\_mspp\_compliance\_summary mspp),

-- TEST\_HEAD

-- AS (SELECT CASE

-- WHEN area IS NULL AND region IS NULL

-- THEN

-- 'zEnterprise'

-- ELSE

-- area

-- END

-- area,

-- region,

-- NVL (th.gud, 0) live,

-- NVL (th.total, 0) th\_reported,

-- th.TOTAL\_COMPLIANCE

-- FROM NGT\_QT\_REG\_SUMM th),

e2bts AS

(SELECT CASE

WHEN area = 'Grand Total' AND region IS NULL

THEN 'zEnterprise' --area IS NULL

ELSE DECODE (area, 'UnMapped', 'unknown', area)

END area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL (e2bts.live, 0) live, NVL (e2bts.rptd, 0) rptd,

e2bts.total\_compliance

-- NVL (e2BTS.TOTAL\_COMPLIANCE, 0) TOTAL\_COMPLIANCE

FROM xng\_reports.ebh\_term\_bts\_regional\_summary e2bts),

csr AS

(SELECT CASE

WHEN area IS NULL AND region IS NULL

THEN 'zEnterprise' --area IS NULL

ELSE DECODE (area, 'UnMapped', 'unknown', area)

END area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL (csr.live\_csr\_w\_ebh\_pathset, 0) live,

NVL (csr.den\_for\_compliance\_per, 0) rptd, csr.compliance

FROM xng\_reports.csr\_summary\_by\_region csr),

hpov AS

(SELECT CASE

WHEN area = 'Enterprise Totals' AND region IS NULL

THEN 'zEnterprise'

ELSE DECODE (area, 'UnMapped', 'unknown', area)

END area,

DECODE (region, 'UnMapped', 'unknown', region) region,

NVL (hpov.tot\_csr\_matched\_per\_region, 0) live,

NVL ( hpov.tot\_xo\_live\_csr\_per\_region

+ hpov.tot\_csr\_matched\_per\_region,

0

) rptd,

ROUND ( ( hpov.tot\_csr\_matched\_per\_region

/ ( hpov.tot\_csr\_matched\_per\_region

+ hpov.tot\_xo\_live\_csr\_per\_region

)

)

\* 100,

2

) compliance

FROM xng\_reports.csr\_missing\_in\_hpov hpov),

calc\_summ AS

(SELECT dlr.area, dlr.region,

NVL (e.live, 0)

+ NVL (m.live\_xconnect\_match\_count, 0)

--+ th.live

+ NVL (e2b.live, 0)

+ NVL (c.live, 0)

+ NVL (t.live, 0)

+ NVL (h.live, 0) live,

NVL (e.enb\_reported, 0)

+ NVL (m.total\_mspp\_for\_reported, 0)

--+ th.th\_reported

+ NVL (e2b.rptd, 0)

+ NVL (c.rptd, 0)

+ NVL (t.rptd, 0)

+ NVL (h.rptd, 0) rptd,

NVL (e.total\_compliance, 0) enb\_total\_compliance,

NVL (m.total\_compliance, 0) mspp\_total\_compliance,

--th.TOTAL\_COMPLIANCE th\_TOTAL\_COMPLIANCE,

NVL (e2b.total\_compliance, 0) e2bts\_total\_compliance,

NVL (c.compliance, 0) csr\_total\_compliance,

NVL (t.testability\_compliance, 0) t1\_total\_compliance,

NVL (h.compliance, 0) hpov\_total\_compliance,

ROUND

( ( ( NVL (e.live, 0)

+ NVL (m.live\_xconnect\_match\_count, 0)

-- + th.live

+ NVL (e2b.live, 0)

+ NVL (c.live, 0)

+ NVL (t.live, 0)

+ NVL (h.live, 0)

)

/ ( NVL (e.enb\_reported, 0)

+ NVL (m.total\_mspp\_for\_reported, 0)

--+ th.th\_reported

+ NVL (e2b.rptd, 0)

+ NVL (c.rptd, 0)

+ NVL (t.rptd, 0)

+ NVL (h.rptd, 0)

)

)

\* 100,

2

) overall\_compliance

FROM regions dlr LEFT OUTER JOIN enb e

ON ( UPPER (e.area) = UPPER (dlr.area)

AND UPPER (e.region) = UPPER (dlr.region)

OR UPPER (e.area) = UPPER (dlr.area)

AND e.region IS NULL

AND dlr.region IS NULL

)

LEFT OUTER JOIN mspp m

ON ( UPPER (m.area) = UPPER (dlr.area)

AND UPPER (m.region) = UPPER (dlr.region)

OR UPPER (m.area) = UPPER (dlr.area)

AND m.region IS NULL

AND dlr.region IS NULL

)

-- LEFT OUTER JOIN TEST\_HEAD th

-- ON (UPPER (th.area) = UPPER (dlr.area)

-- AND UPPER (th.region) =

-- UPPER (dlr.region)

-- OR UPPER (th.area) = UPPER (dlr.area)

-- AND th.region IS NULL

-- AND dlr.region IS NULL)

LEFT OUTER JOIN e2bts e2b

ON ( UPPER (e2b.area) = UPPER (dlr.area)

AND UPPER (e2b.region) = UPPER (dlr.region)

OR UPPER (e2b.area) = UPPER (dlr.area)

AND e2b.region IS NULL

AND dlr.region IS NULL

)

LEFT OUTER JOIN csr c

ON ( UPPER (c.area) = UPPER (dlr.area)

AND UPPER (c.region) = UPPER (dlr.region)

OR UPPER (c.area) = UPPER (dlr.area)

AND c.region IS NULL

AND dlr.region IS NULL

)

LEFT OUTER JOIN t1 t

ON ( UPPER (t.area) = UPPER (dlr.area)

AND UPPER (t.region) = UPPER (dlr.region)

OR UPPER (t.area) = UPPER (dlr.area)

AND t.region IS NULL

AND dlr.region IS NULL

)

LEFT OUTER JOIN hpov h

ON ( UPPER (h.area) = UPPER (dlr.area)

AND UPPER (h.region) = UPPER (dlr.region)

OR UPPER (h.area) = UPPER (dlr.area)

AND h.region IS NULL

AND dlr.region IS NULL

)

),

calc\_deltas AS

( -- to calculate regional delta

SELECT dlr.area, dlr.region, dlr.live, dlr.rptd,

dlr.enb\_total\_compliance, dlr.mspp\_total\_compliance,

--dlr.th\_TOTAL\_COMPLIANCE,

dlr.e2bts\_total\_compliance, dlr.csr\_total\_compliance,

dlr.t1\_total\_compliance, dlr.hpov\_total\_compliance,

dlr.overall\_compliance,

NVL (dlr.overall\_compliance, 0)

- NVL (prev.overall\_compliance, 0) regional\_monthly\_delta

FROM calc\_summ dlr JOIN n2n\_overall\_summ\_prev\_month prev

ON prev.region = dlr.region

UNION

-- to calculate regional delta

SELECT dlr.area, dlr.region, dlr.live, dlr.rptd,

dlr.enb\_total\_compliance, dlr.mspp\_total\_compliance,

--dlr.th\_TOTAL\_COMPLIANCE,

dlr.e2bts\_total\_compliance, dlr.csr\_total\_compliance,

dlr.t1\_total\_compliance, dlr.hpov\_total\_compliance,

dlr.overall\_compliance,

NVL (dlr.overall\_compliance, 0)

- NVL (prev.overall\_compliance, 0) regional\_monthly\_delta

FROM calc\_summ dlr JOIN n2n\_overall\_summ\_prev\_month prev

ON prev.area = dlr.area

AND prev.region IS NULL

AND dlr.region IS NULL --calculate column deltas

UNION

SELECT 'zMetricDelta' area, NULL region, dlr.live - prev.live live,

dlr.rptd - prev.rptd rptd,

dlr.enb\_total\_compliance - NVL (prev.enb\_total\_compliance, 0),

dlr.mspp\_total\_compliance

- NVL (prev.mspp\_total\_compliance, 0),

--dlr.th\_total\_compliance - nvl(prev.th\_total\_compliance, 0),

dlr.e2bts\_total\_compliance

- NVL (prev.e2bts\_total\_compliance, 0),

dlr.csr\_total\_compliance - NVL (prev.csr\_total\_compliance, 0),

dlr.t1\_total\_compliance - NVL (prev.t1\_total\_compliance, 0),

dlr.hpov\_total\_compliance

- NVL (prev.hpov\_total\_compliance, 0),

dlr.overall\_compliance - NVL (prev.overall\_compliance, 0),

NULL regional\_monthly\_delta

FROM calc\_summ dlr JOIN n2n\_overall\_summ\_prev\_month prev

ON prev.area = dlr.area AND prev.area = 'zEnterprise'

)

SELECT "AREA", "REGION", "LIVE", "RPTD", "ENB\_TOTAL\_COMPLIANCE",

"MSPP\_TOTAL\_COMPLIANCE" --,"TH\_TOTAL\_COMPLIANCE"

,

"E2BTS\_TOTAL\_COMPLIANCE", "CSR\_TOTAL\_COMPLIANCE",

"T1\_TOTAL\_COMPLIANCE", "HPOV\_TOTAL\_COMPLIANCE",

"OVERALL\_COMPLIANCE", "REGIONAL\_MONTHLY\_DELTA"

FROM calc\_deltas dlr

ORDER BY dlr.area, dlr.region

;

--------------------------------------------------------

-- DDL for View NCM\_CSR\_VLAN\_DISC\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."NCM\_CSR\_VLAN\_DISC\_V" ("HOSTNAME", "LOOPBACK0\_IP", "S1\_X2\_IP\_1\_100", "S1\_X2\_IP\_2\_100", "DEVICE\_IP\_AKA\_LOOPBACK300\_IP", "LOOPBACK400\_IPV4\_IP", "LOOPBACK400\_IPV6\_IP", "OAM\_IP\_1\_101", "OAM\_IP\_2\_101") AS

select NE.HOSTNAME, LP0.PORT\_IP LOOPBACK0\_IP, LP1100.PORT\_IP S1\_X2\_IP\_1\_100,

LP2100.PORT\_IP S1\_X2\_IP\_2\_100,

lp300.PORT\_IP DEVICE\_IP\_AKA\_LOOPBACK300\_IP, lp400\_ipv4.PORT\_IP LOOPBACK400\_IPV4\_IP, lp400\_ipv6.PORT\_IP LOOPBACK400\_IPV6\_IP,

oam1.PORT\_IP OAM\_IP\_1\_101, oam2.PORT\_IP OAM\_IP\_2\_101

from (select distinct hostname from XNG\_REPORTS.NCM\_CSR\_WK ) ne

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_CSR\_VLAN\_WK where sub\_interface\_name='Loopback0' and port\_ip like '%.%.%.%') lp0

on NE.HOSTNAME = LP0.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_CSR\_VLAN\_WK where sub\_interface\_name='GigabitEthernet0/0/1' and port\_ip like '%:%' and VLAN\_NUMBER=100) lp1100

on NE.HOSTNAME = LP1100.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_CSR\_VLAN\_WK where sub\_interface\_name='GigabitEthernet0/0/2' and port\_ip like '%:%' and VLAN\_NUMBER=100) lp2100

on NE.HOSTNAME = LP2100.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_CSR\_VLAN\_WK where sub\_interface\_name='Loopback300' and port\_ip like '%.%.%.%' ) lp300

on NE.HOSTNAME = LP300.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_CSR\_VLAN\_WK where sub\_interface\_name='Loopback400' and port\_ip like '%.%.%.%') lp400\_ipv4

on NE.HOSTNAME = LP400\_ipv4.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_CSR\_VLAN\_WK where sub\_interface\_name='Loopback400' and port\_ip like '%:%') lp400\_ipv6

on NE.HOSTNAME = LP400\_ipv6.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_CSR\_VLAN\_WK where sub\_interface\_name='GigabitEthernet0/0/1' and port\_ip like '%.%.%.%' and VLAN\_NUMBER=101) oam1

on NE.HOSTNAME = oam1.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_CSR\_VLAN\_WK where sub\_interface\_name='GigabitEthernet0/0/2' and port\_ip like '%.%.%.%' and VLAN\_NUMBER=101) oam2

on NE.HOSTNAME = oam2.HOSTNAME

;

--------------------------------------------------------

-- DDL for View NCM\_NGMLS\_VLAN\_DISC\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_DISC\_V" ("HOSTNAME", "DEVICE\_IP", "GLOBAL\_VRF\_EBH\_LP\_0\_IP", "NMS\_VIRTUAL\_LP\_10", "RAN\_VRF\_LP\_100", "DEVICE\_IP\_AKA\_LOOPBACK300\_IP", "LTE\_VRF\_LP\_400\_IPV6\_IP", "LTE\_VRF\_LP\_400\_IPV4\_IP") AS

select distinct NE.HOSTNAME, NE.DEVICE\_IP, LP0.PORT\_IP GLOBAL\_VRF\_EBH\_LP\_0\_IP, LP10.PORT\_IP NMS\_VIRTUAL\_LP\_10,

LP100.PORT\_IP RAN\_VRF\_LP\_100, lp300.PORT\_IP DEVICE\_IP\_AKA\_LOOPBACK300\_IP, lp400\_ipv6.PORT\_IP LTE\_VRF\_LP\_400\_IPV6\_IP, lp400\_ipv4.PORT\_IP LTE\_VRF\_LP\_400\_IPV4\_IP

from (select distinct hostname, device\_ip from XNG\_REPORTS.NCM\_NGMLS\_WK ) ne

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_NGMLS\_VLAN\_WK where sub\_interface\_name='Loopback0' and port\_ip like '%.%.%.%') lp0

on NE.HOSTNAME = LP0.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_NGMLS\_VLAN\_WK where sub\_interface\_name='Loopback10' and port\_ip like '%.%.%.%') lp10

on NE.HOSTNAME = LP10.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_NGMLS\_VLAN\_WK where sub\_interface\_name='Loopback100' and port\_ip like '%.%.%.%' ) lp100

on NE.HOSTNAME = LP100.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_NGMLS\_VLAN\_WK where sub\_interface\_name='Loopback300' and port\_ip like '%.%.%.%' ) lp300

on NE.HOSTNAME = LP300.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_NGMLS\_VLAN\_WK where sub\_interface\_name='Loopback400' and port\_ip like '%:%') lp400\_ipv6

on NE.HOSTNAME = LP400\_ipv6.HOSTNAME

left outer join (select hostname, port\_ip from XNG\_REPORTS.NCM\_NGMLS\_VLAN\_WK where sub\_interface\_name='Loopback400' and port\_ip like '%.%.%.%') lp400\_ipv4

on NE.HOSTNAME = LP400\_ipv4.HOSTNAME

;

--------------------------------------------------------

-- DDL for View NGMLS\_ALL\_GI\_AUDIT\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."NGMLS\_ALL\_GI\_AUDIT\_V" ("AREA", "REGION", "NGMLS\_DEVICE\_NAME", "SAM\_NCM\_DEVICE\_NAME", "NGMLS\_VENDOR", "ISSUES", "MATCH\_CODE", "MATCH\_STATUS", "GI\_DEVICE\_NAME", "NE\_INST\_ID", "LIVE\_IN\_XNG", "EQ\_STATUS", "MARKET", "LEAF\_DOMAIN\_INST\_ID") AS

WITH sam

AS (SELECT DISTINCT

n.area,

n.region,

n.ngmls\_device\_name,

n.sam\_ngmls\_device\_name sam\_ncm\_device\_name,

n.ngmls\_vendor,

i.description issues,

DECODE (n.match\_code,

'Xng Only', 'Granite Only',

n.match\_code)

match\_code,

n.match\_status,

n.gi\_device\_name,

n.ne\_inst\_id,

n.live\_in\_xng,

n.eq\_status,

n.market,

n.leaf\_domain\_inst\_id

FROM xng\_reports.SAM\_GI\_AL\_NGMLS\_AUDIT n

LEFT OUTER JOIN xng\_reports.ngmls\_device\_audit\_issues d

ON N.NGMLS\_DEVICE\_NAME = d.NGMLS\_DEVICE\_NAME

LEFT OUTER JOIN xng\_reports.ngmls\_issue i

ON I.NGMLS\_ISSUE\_ID = d.NGMLS\_ISSUE\_ID),

ncm

AS (SELECT DISTINCT

n.area,

n.region,

n.ngmls\_device\_name,

n.ncm\_ngmls\_hostname sam\_ncm\_device\_name,

n.ngmls\_vendor,

i.description issues,

DECODE (n.match\_code,

'Xng Only', 'Granite Only',

n.match\_code)

match\_code,

n.match\_status,

n.gi\_device\_name,

n.ne\_inst\_id,

n.live\_in\_xng,

n.eq\_status,

n.market,

n.leaf\_domain\_inst\_id

FROM xng\_reports.ncm\_gi\_ci\_ngmls\_audit n

LEFT OUTER JOIN xng\_reports.ngmls\_device\_audit\_issues d

ON N.NGMLS\_DEVICE\_NAME = d.NGMLS\_DEVICE\_NAME

LEFT OUTER JOIN xng\_reports.ngmls\_issue i

ON I.NGMLS\_ISSUE\_ID = d.NGMLS\_ISSUE\_ID)

SELECT area,

region,

ngmls\_device\_name,

sam\_ncm\_device\_name,

ngmls\_vendor,

issues,

match\_code,

match\_status,

gi\_device\_name,

ne\_inst\_id,

live\_in\_xng,

eq\_status,

market,

leaf\_domain\_inst\_id

FROM sam

UNION

SELECT area,

region,

ngmls\_device\_name,

sam\_ncm\_device\_name,

ngmls\_vendor,

issues,

match\_code,

match\_status,

gi\_device\_name,

ne\_inst\_id,

live\_in\_xng,

eq\_status,

market,

leaf\_domain\_inst\_id

FROM ncm

;

--------------------------------------------------------

-- DDL for View NGMLS\_ALL\_GI\_REGION\_SUMM\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."NGMLS\_ALL\_GI\_REGION\_SUMM\_V" ("AREA", "REGION", "NGMLSS\_ALL", "GI\_NGMLSS", "GI\_L\_NGMLSS", "GI\_NL\_NGMLSS", "NGMLS\_ALL\_GI\_MATCH", "NGMLS\_ALL\_GI\_L\_MATCH", "NGMLS\_ALL\_GI\_NL\_MATCH", "NGMLS\_ALL\_GI\_MISMATCH", "NGMLS\_ALL\_GI\_PER") AS

WITH DOMAINS

AS (SELECT DISTINCT area,

region,

market,

leaf\_domain\_inst\_id,

ngmls\_device\_name

FROM ngmls\_all\_gi\_audit\_v

WHERE region <> 'NNO' AND region <> 'OSS'

UNION

SELECT 'unknown' area,

'unknown' region,

'unknown' market,

NULL leaf\_domain\_inst\_id,

'unknown' ngmls\_device\_name

FROM DUAL),

ngmls

AS ( SELECT ngmls\_device\_name, COUNT (\*) all\_ngmlss

FROM ngmls\_all\_gi\_audit\_v

WHERE match\_code <> 'Granite Only'

GROUP BY ngmls\_device\_name),

Xng

AS (SELECT DISTINCT

ngMLS\_device\_name,

ngMLS.ne\_inst\_id,

NVL (CASE WHEN status = 'Live' THEN 1 ELSE 0 END, 0) l,

NVL (CASE WHEN status <> 'Live' THEN 1 ELSE 0 END, 0) nl

FROM XNG\_ngMLS\_equip\_wk ngMLS

WHERE eq\_class\_type = 'NE'),

GI\_CNTS

AS ( SELECT NVL (area, 'unknown') area,

NVL (region, 'unknown') region,

CASE

WHEN di.ngmls\_device\_name IS NULL THEN 'unknown'

ELSE xng.ngmls\_device\_name

END

ngmls\_device\_name,

SUM (l) l,

SUM (nl) nl

FROM Xng, DOMAINS di

WHERE xng.ngmls\_device\_name = di.ngmls\_device\_name(+)

GROUP BY NVL (area, 'unknown'),

NVL (region, 'unknown'),

CASE

WHEN di.ngmls\_device\_name IS NULL THEN 'unknown'

ELSE xng.ngmls\_device\_name

END),

--GI mismatch details

gi\_mismatch

AS ( -- any issue with the device is a mismatch

SELECT DISTINCT cdai.ngmls\_device\_name

FROM ngmls\_device\_audit\_issues\_wk cdai),

gud\_matches

AS ( -- all devices which are not in Granite Only

SELECT ngmls\_device\_name

FROM ngmls\_all\_gi\_audit\_v aud

WHERE match\_code <> 'Granite Only'

MINUS

-- remove all devices that have critical issues

SELECT cdai.ngmls\_device\_name

FROM ngmls\_device\_audit\_issues\_wk cdai, ngmls\_issue n

WHERE n.is\_critical = 'Y'

AND n.ngmls\_issue\_id = cdai.ngmls\_issue\_id),

gi\_match

AS (SELECT aud.ngmls\_device\_name,

CASE WHEN live\_in\_xng = 'Y' THEN 1 ELSE 0 END gi\_l\_match,

CASE WHEN live\_in\_xng <> 'Y' THEN 1 ELSE 0 END gi\_nl\_match

FROM ngmls\_all\_gi\_audit\_v aud

JOIN gud\_matches gud

ON gud.ngmls\_device\_name = aud.ngmls\_device\_name),

match\_counts

AS ( SELECT CASE

WHEN ma.ngmls\_device\_name IS NOT NULL

THEN

ma.ngmls\_device\_name

WHEN ms.ngmls\_device\_name IS NOT NULL

THEN

ms.ngmls\_device\_name

END

ngmls\_device\_name,

COUNT (ma.ngmls\_device\_name) gi\_match,

SUM (ma.gi\_l\_match) gi\_l\_match,

SUM (ma.gi\_nl\_match) gi\_nl\_match,

COUNT (ms.ngmls\_device\_name) gi\_mismatch

FROM gi\_match ma

FULL OUTER JOIN gi\_mismatch ms

ON ma.ngmls\_device\_name = ms.ngmls\_device\_name

GROUP BY CASE

WHEN ma.ngmls\_device\_name IS NOT NULL

THEN

ma.ngmls\_device\_name

WHEN ms.ngmls\_device\_name IS NOT NULL

THEN

ms.ngmls\_device\_name

END)

SELECT di.area,

di.region,

SUM (n.all\_ngmlss) ngmlss\_all,

SUM (NVL (l, 0)) + SUM (NVL (nl, 0)) gi\_ngmlss,

SUM (l) gi\_l\_ngmlss,

SUM (nl) gi\_nl\_ngmlss,

SUM (gi\_match) ngmls\_all\_gi\_match,

SUM (gi\_l\_match) ngmls\_all\_gi\_l\_match,

SUM (gi\_nl\_match) ngmls\_all\_gi\_nl\_match,

SUM (gi\_mismatch) ngmls\_all\_gi\_mismatch,

ROUND (

NVL (SUM (gi\_match), 0)

\* 2

/ DECODE (

NVL (SUM (n.all\_ngmlss), 0)

+ NVL (SUM (NVL (l, 0)) + SUM (NVL (nl, 0)), 0),

0, 1,

NVL (SUM (n.all\_ngmlss), 0)

+ NVL (SUM (NVL (l, 0)) + SUM (NVL (nl, 0)), 0))

\* 100,

2)

ngmls\_all\_gi\_per

FROM DOMAINS di,

ngmls n,

GI\_CNTS xng,

match\_counts cnt

WHERE di.ngmls\_device\_name = n.ngmls\_device\_name(+)

AND di.ngmls\_device\_name = xng.ngmls\_device\_name(+)

AND di.ngmls\_device\_name = cnt.ngmls\_device\_name(+)

GROUP BY ROLLUP (di.area, di.region)

ORDER BY di.area, di.region

;

--------------------------------------------------------

-- DDL for View NORTEL\_EVDO\_AUDIT\_DOM\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."NORTEL\_EVDO\_AUDIT\_DOM\_V" ("MATCH\_CODE", "MATCH\_STATUS", "TERMINATION\_TYPE", "VSM\_DEVICE\_NAME\_EMS", "VSM\_DEVICE\_NAME\_BSM", "VSM\_DEVICE\_NAME\_MTX", "NE\_DCG\_NUMBER", "NE\_DOM\_IP", "NE\_SLOT\_NUMBER", "NE\_SPAN\_NUMBER", "NE\_SPAN\_TYPE", "NE\_STATUS", "SWITCH\_ID", "XNG\_DOM\_IP", "XNG\_SLOT\_NUMBER", "XNG\_DCG\_NUMBER", "XNG\_SPAN\_NUMBER", "XNG\_PATH\_STATUS", "XNG\_PATH\_TYPE", "CIRC\_PATH\_HUM\_ID", "CIRC\_PATH\_INST\_ID", "XNG\_BANDWIDTH", "LEAF\_DOMAIN\_INST\_ID", "LEAF\_DOMAIN\_NAME", "AREA", "REGION") AS

WITH nedom

AS (SELECT DISTINCT SUBSTR (clli, 1, 6) clli6,

leaf\_domain\_inst\_id,

leaf\_domain\_name,

area,

region

FROM xng\_reports.clli\_domain\_map\_v),

gidom

AS (SELECT circ\_path\_inst\_id,

dlr.domain\_inst\_id leaf\_domain\_inst\_id,

dlr.domain\_name leaf\_domain\_name,

area,

region

FROM vzwnet.path\_domain\_map pdm,

xng\_reports.DOMAINS\_LEAF\_REPORTING dlr

WHERE pdm.domain\_inst\_id = dlr.domain\_inst\_id),

nad

AS (SELECT nnvxa.match\_code,

NVL (nnvxa.match\_status, nnvxa.match\_code) match\_status,

nnvxa.termination\_type,

nnvxa.vsm\_device\_name\_ems,

nnvxa.vsm\_device\_name\_bsm,

nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number,

nnvxa.ne\_dom\_ip,

nnvxa.ne\_slot\_number,

nnvxa.ne\_span\_number,

nnvxa.ne\_span\_type,

nnvxa.ne\_status,

nnvxa.switch\_id,

nnvxa.xng\_dom\_ip,

nnvxa.xng\_slot\_number,

nnvxa.xng\_dcg\_number,

nnvxa.xng\_span\_number,

nnvxa.xng\_path\_status,

nnvxa.circ\_type xng\_path\_type,

nnvxa.circ\_path\_hum\_id,

nnvxa.circ\_path\_inst\_id,

nnvxa.xng\_bandwidth,

SUBSTR (nnvxa.vsm\_device\_name\_ems, 1, 6) clli6

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

WHERE nnvxa.ne\_span\_type = 'DATA\_SPAN')

SELECT DISTINCT

nad.match\_code,

nad.match\_status,

nad.termination\_type,

nad.vsm\_device\_name\_ems,

nad.vsm\_device\_name\_bsm,

nad.vsm\_device\_name\_mtx,

nad.ne\_dcg\_number,

nad.ne\_dom\_ip,

nad.ne\_slot\_number,

nad.ne\_span\_number,

nad.ne\_span\_type,

nad.ne\_status,

nad.switch\_id,

nad.xng\_dom\_ip,

nad.xng\_slot\_number,

nad.xng\_dcg\_number,

nad.xng\_span\_number,

nad.xng\_path\_status,

nad.xng\_path\_type,

nad.circ\_path\_hum\_id,

nad.circ\_path\_inst\_id,

nad.xng\_bandwidth,

NVL (nedom.leaf\_domain\_inst\_id, gidom.leaf\_domain\_inst\_id)

leaf\_domain\_inst\_id,

NVL (nedom.leaf\_domain\_name, gidom.leaf\_domain\_name)

leaf\_domain\_name,

NVL (nedom.area, gidom.area) area,

NVL (nedom.region, gidom.region) region

FROM nad, nedom, gidom

WHERE nad.clli6 = nedom.clli6(+)

AND nad.circ\_path\_inst\_id = gidom.circ\_path\_inst\_id(+)

;

--------------------------------------------------------

-- DDL for View NT\_EMS\_GI\_VDDM\_CA\_ISSUES

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."NT\_EMS\_GI\_VDDM\_CA\_ISSUES" ("VSM\_DEVICE\_NAME\_EMS", "IN\_VSM\_NAME\_NAME\_EMS", "EMS\_STATUS", "IN\_VDDM", "IN\_CA") AS

WITH

ALL\_EMS as

(

select vsm\_device\_name\_ems

from VSM\_SANE\_NAME\_EMS

where ems\_status <> 'Deprecated'

union

select vsm\_device\_name

from vsm\_devices\_domain\_map vddm

where vsm\_device\_name like '%EVDO'

union

select vsm\_device\_name

from CELL\_AVAIL\_ACTIVE ca

where ca.vsm\_device\_name like '%EVDO'

)

select ems.vsm\_device\_name\_ems

, case when ve.vsm\_device\_name\_ems is null then 'Missing in VSM\_SANE\_NAME\_EMS' else ve.vsm\_device\_name\_ems end in\_vsm\_name\_name\_ems, ve.EMS\_STATUS

, case when vddm.vsm\_device\_name is null then 'Missing in vddm' else vddm.vsm\_device\_name end in\_vddm

, case when ca.vsm\_device\_name is null then 'Missing in ca' else ca.vsm\_device\_name end in\_ca

from ALL\_EMS ems

left outer join VSM\_SANE\_NAME\_EMS ve

on ems.VSM\_device\_NAME\_EMS = ve.vsm\_device\_name\_ems

left outer join vsm\_devices\_domain\_map vddm

on ems.VSM\_device\_NAME\_EMS = vddm.vsm\_device\_name

left outer join cell\_avail\_active ca

on ems.VSM\_device\_NAME\_EMS = ca.vsm\_device\_name

where ve.vsm\_device\_name\_ems is null

or vddm.vsm\_device\_name is null

or ca.vsm\_device\_name is null

;

--------------------------------------------------------

-- DDL for View NT\_EVDO\_DCG\_Z\_SITE\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."NT\_EVDO\_DCG\_Z\_SITE\_V" ("VENDOR", "SERVICE", "VSM\_DEVICE\_NAME\_EMS", "VSM\_DEVICE\_NAME\_BSM", "RAW\_NAME\_BSM", "NE\_DCG\_NUMBER", "SITE\_HUM\_ID", "SITE\_INST\_ID", "COUNTY") AS

WITH

ALL\_NE\_DCG as

(

-- get CDMA only

select distinct NEXA.vsm\_device\_name\_ems, NEXA.vsm\_device\_name\_bsm, nexa.ne\_dcg\_number, vsrnb.RAW\_NAME\_BSM

from nortel\_ne\_vs\_xng\_audit nexa

join VSM\_SANE\_RAW\_NAME\_BSM vsrnb on nexa.vsm\_device\_name\_bsm = vsrnb.vsm\_device\_name\_bsm

where match\_code <> 'Xng Only'

and nexa.NE\_SPAN\_TYPE = 'DATA\_SPAN'

)

,

ALL\_DCG\_Z\_SIDE\_SITES as

(

--brings all distinct set of all ECP, ne\_bts, z\_site

select distinct NEXA.vsm\_device\_name\_ems, nexa.vsm\_device\_name\_bsm, RAW\_NAME\_BSM, nexa.NE\_DCG\_NUMBER, si.SITE\_HUM\_ID, si.SITE\_INST\_ID, si.county

from nortel\_ne\_vs\_xng\_audit nexa

join VSM\_SANE\_RAW\_NAME\_BSM vsrnb on nexa.vsm\_device\_name\_bsm = vsrnb.vsm\_device\_name\_bsm

join vzwnet.circ\_path\_inst cpi on nexa.CIRC\_PATH\_INST\_ID = cpi.CIRC\_PATH\_INST\_ID

join vzwnet.site\_inst si on cpi.Z\_SIDE\_SITE\_ID = si.SITE\_INST\_ID

where match\_code <> 'Xng Only'

and nexa.NE\_SPAN\_TYPE = 'DATA\_SPAN'

)

,

ALL\_NE\_ONLY\_BTS as

(

--brings all distinct set of all ECP, ne\_bts that dont have any paths

select vsm\_device\_name\_ems, vsm\_device\_name\_bsm, raw\_name\_bsm, ne\_dcg\_number

from ALL\_NE\_DCG nexa

minus

select vsm\_device\_name\_ems, vsm\_device\_name\_bsm, raw\_name\_bsm, ne\_dcg\_number

from ALL\_DCG\_Z\_SIDE\_SITES a

)

,

GET\_ALL as

(

select 'nortel' vendor, 'evdo' service, vsm\_device\_name\_ems, vsm\_device\_name\_bsm, raw\_name\_bsm, ne\_dcg\_number, site\_hum\_id, site\_inst\_id, county

from ALL\_DCG\_Z\_SIDE\_SITES

union

select 'nortel' vendor, 'evdo' service, vsm\_device\_name\_ems, vsm\_device\_name\_bsm, raw\_name\_bsm, ne\_dcg\_number, null, null, null

from ALL\_NE\_ONLY\_BTS

)

select vendor, service, vsm\_device\_name\_ems, vsm\_device\_name\_bsm, raw\_name\_bsm, ne\_dcg\_number, site\_hum\_id, site\_inst\_id, county

from GET\_ALL a

order by vsm\_device\_name\_ems, vsm\_device\_name\_bsm, raw\_name\_bsm, ne\_dcg\_number ;

COMMENT ON TABLE "XNG\_REPORTS"."NT\_EVDO\_DCG\_Z\_SITE\_V" IS 'Maps all EMS-BSM-DCG discovered via SANE/Datapro to Xng-Site. If one of the many spans to a BSM-DCG goes to a site, and others dont, still this view will map the whole BSM-DCG to the Xng-Site.'

;

--------------------------------------------------------

-- DDL for View NT\_Z\_SITE\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."NT\_Z\_SITE\_V" ("VSM\_DEVICE\_NAME\_BSM", "CELL\_NUMBER", "NE\_DCG\_NUMBER", "SITE\_HUM\_ID", "COUNTY") AS

WITH

NW\_CELLS\_IN\_SITE as

(

select distinct b.VSM\_DEVICE\_NAME\_BSM, b.CELL\_NUMBER, nvxa.ne\_dcg\_number ,site\_hum\_id, county

from BSM\_CELL\_DCG\_MAP b

join nortel\_ne\_vs\_xng\_audit nvxa

on nvxa.VSM\_DEVICE\_NAME\_BSM = b.VSM\_DEVICE\_NAME\_BSM

and nvxa.NE\_DCG\_NUMBER = b.DCG\_NUMBER

join vzwnet.circ\_path\_inst cpi

on nvxa.CIRC\_PATH\_INST\_ID = cpi.CIRC\_PATH\_INST\_ID

join vzwnet.site\_inst si

on si.SITE\_INST\_ID = cpi.Z\_SIDE\_SITE\_ID

where nvxa.match\_code <> 'Xng Only'

)

select distinct b.VSM\_DEVICE\_NAME\_BSM, b.CELL\_NUMBER, nvxa.ne\_dcg\_number, nvxa.site\_hum\_id, county

from BSM\_CELL\_DCG\_MAP b

left outer join NW\_CELLS\_IN\_SITE nvxa

on nvxa.VSM\_DEVICE\_NAME\_BSM = b.VSM\_DEVICE\_NAME\_BSM

and nvxa.CELL\_NUMBER = b.CELL\_NUMBER

order by VSM\_DEVICE\_NAME\_BSM, b.CELL\_NUMBER, nvxa.ne\_dcg\_number, nvxa.site\_hum\_id ;

COMMENT ON TABLE "XNG\_REPORTS"."NT\_Z\_SITE\_V" IS 'Maps all distinct BSM-DCG discovered via SANE to Xng-Site. If one of the many spans to a DCG goes to a site, and others dont, still this view will map the whole DCG to the Xng-Site.'

;

--------------------------------------------------------

-- DDL for View OPSTRACKER\_TASK\_QUERY

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."OPSTRACKER\_TASK\_QUERY" ("WO\_NAME", "TASK\_NUMBER", "TASK\_NAME", "STATUS", "SITE\_INST\_ID", "NAURU\_SITE\_ID", "WO\_DESCRIPTION", "TASK\_URL", "USER\_NAME") AS

SELECT WOI.WO\_NAME,

WTI.TASK\_NUMBER,

WTI.TASK\_NAME,

VTS.STATUS\_NAME STATUS,

SI.SITE\_INST\_ID,

XSM.NAURU\_SITE\_ID,

WOI.DESCRIPTION WO\_DESCRIPTION,

'http://xng.vh.vzwnet.com/pls/vzwnet/xweb\_works.xperworks\_show\_task?taskinstid='

|| WTI.TASK\_INST\_ID

TASK\_URL,

QV.USER\_NAME

FROM vzwnet.wo\_task\_inst wti,

vzwnet.val\_task\_status vts,

vzwnet.work\_order\_inst woi,

vzwnet.queue\_user\_privilege\_view qv,

XNG\_APPS.NAURU\_XNG\_SITE\_MAP xsm,

vzwnet.site\_inst si

WHERE WTI.WO\_INST\_ID = WOI.WO\_INST\_ID

AND WTI.ELEMENT\_TYPE = 'I'

AND WTI.ELEMENT\_INST\_ID = SI.SITE\_INST\_ID

AND WTI.QUEUE\_INST\_ID = QV.QUEUE\_INST\_ID

AND QV.PRIV\_MASK LIKE '\_1%'

AND VTS.STAT\_CODE = WTI.STATUS\_CODE

AND VTS.STATUS\_NAME IN ('READY', 'IN-PROCESS')

AND SI.SITE\_INST\_ID = XSM.SITE\_INST\_ID(+)

ORDER BY 1, 2

;

--------------------------------------------------------

-- DDL for View SAME\_IP\_MULT\_VSM\_DEV\_ISSUES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."SAME\_IP\_MULT\_VSM\_DEV\_ISSUES\_V" ("VSM\_IP", "VSM\_DEVICE\_NAME", "VSM\_CLASS", "AS\_OF\_DATE", "ALLTEL\_DO\_NOT\_REMOVE", "LEAF\_DOMAIN\_INST\_ID") AS

WITH all\_devices\_w\_ip AS

(SELECT ip.vsm\_ip, vddm.\*

FROM vsm\_devices\_ipaddresses ip JOIN vsm\_devices\_domain\_map vddm

ON ip.vsm\_device\_name = vddm.vsm\_device\_name

WHERE vddm.vsm\_class LIKE '%MSPP'),

ip\_mult\_names AS

(SELECT ip.vsm\_ip, COUNT (1)

FROM all\_devices\_w\_ip ip

GROUP BY ip.vsm\_ip

HAVING COUNT (1) > 1)

SELECT vdip."VSM\_IP",vdip."VSM\_DEVICE\_NAME",vdip."VSM\_CLASS",vdip."AS\_OF\_DATE",vdip."ALLTEL\_DO\_NOT\_REMOVE",vdip."LEAF\_DOMAIN\_INST\_ID"

FROM all\_devices\_w\_ip vdip JOIN ip\_mult\_names ip

ON ip.vsm\_ip = vdip.vsm\_ip

ORDER BY vdip.vsm\_ip, vdip.vsm\_device\_name

;

--------------------------------------------------------

-- DDL for View SAME\_VSM\_DEV\_MULT\_IP\_ISSUES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."SAME\_VSM\_DEV\_MULT\_IP\_ISSUES\_V" ("VSM\_IP", "VSM\_DEVICE\_NAME", "VSM\_CLASS", "AS\_OF\_DATE", "ALLTEL\_DO\_NOT\_REMOVE", "LEAF\_DOMAIN\_INST\_ID") AS

WITH all\_devices\_w\_ip AS

(SELECT ip.vsm\_ip, vddm.\*

FROM vsm\_devices\_ipaddresses ip JOIN vsm\_devices\_domain\_map vddm

ON ip.vsm\_device\_name = vddm.vsm\_device\_name

WHERE vddm.vsm\_class LIKE '%MSPP'),

name\_mult\_ip AS

(SELECT ip.vsm\_device\_name, COUNT (1)

FROM all\_devices\_w\_ip ip

GROUP BY ip.vsm\_device\_name

HAVING COUNT (1) > 1)

SELECT vdip."VSM\_IP",vdip."VSM\_DEVICE\_NAME",vdip."VSM\_CLASS",vdip."AS\_OF\_DATE",vdip."ALLTEL\_DO\_NOT\_REMOVE",vdip."LEAF\_DOMAIN\_INST\_ID"

FROM all\_devices\_w\_ip vdip JOIN name\_mult\_ip ip

ON ip.vsm\_device\_name = vdip.vsm\_device\_name

ORDER BY vdip.vsm\_device\_name, vdip.vsm\_ip

;

--------------------------------------------------------

-- DDL for View SAM\_NCM\_AGGREGATE\_VIEW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."SAM\_NCM\_AGGREGATE\_VIEW" ("AREA", "REGION", "EH\_CSRS", "SAM\_CSRS", "BRIX\_CSRS", "GI\_L\_CSRS", "GI\_NL\_CSRS", "EH\_GI\_L\_MATCH", "EH\_GI\_NL\_MATCH", "EH\_GI\_MIS\_MATCH", "EH\_GI\_PER", "SAM\_GI\_L\_MATCH", "SAM\_GI\_NL\_MATCH", "SAM\_GI\_MIS\_MATCH", "SAM\_GI\_PER", "BRIX\_GI\_L\_MATCH", "BRIX\_GI\_NL\_MATCH", "BRIX\_GI\_MIS\_MATCH", "BRIX\_GI\_PER", "GI\_CSRS", "EH\_GI\_MATCH", "BRIX\_GI\_MATCH", "SAM\_GI\_MATCH", "SAM\_EH\_GI\_ISSUES", "HPOV\_CSRS", "NCM\_HPOV\_ISSUES", "NCM\_HPOV\_MATCH", "NCM\_HPOV\_PER", "HPOV\_GI\_MATCH", "HPOV\_GI\_L\_MATCH", "HPOV\_GI\_NL\_MATCH", "HPOV\_GI\_ISSUES", "HPOV\_GI\_PER") AS

select ncm.area,ncm.region,

sum(nvl(sam.EH\_CSRS,0)+nvl(ncm.EH\_CSRS,0)) as EH\_CSRS,

sum(nvl(sam.SAM\_CSRS,0)+nvl(ncm.NCM\_CSRS,0)) as SAM\_CSRS,

sum(nvl(sam.BRIX\_CSRS,0)+nvl(ncm.BRIX\_CSRS,0)) as BRIX\_CSRS,

sum(nvl(sam.GI\_L\_CSRS,0)+nvl(ncm.GI\_L\_CSRS,0)) as GI\_L\_CSRS,

sum( nvl(sam.GI\_NL\_CSRS,0) +nvl(ncm.GI\_NL\_CSRS,0)) as GI\_NL\_CSRS,

sum( nvl(sam.EH\_GI\_L\_MATCH,0) + nvl(ncm.EH\_GI\_L\_MATCH,0)) as EH\_GI\_L\_MATCH,

sum( nvl(sam.EH\_GI\_NL\_MATCH,0) + nvl(ncm.EH\_GI\_NL\_MATCH,0) ) as EH\_GI\_NL\_MATCH,

sum( nvl(sam.EH\_GI\_MIS\_MATCH,0) + nvl(ncm.EH\_GI\_ISSUES,0)) as EH\_GI\_MIS\_MATCH,

round ((((sum( nvl(sam.EH\_GI\_L\_MATCH,0)+nvl(ncm.EH\_GI\_L\_MATCH,0)))\*2)/(sum(nvl(sam.GI\_L\_CSRS,0)+nvl(ncm.GI\_L\_CSRS,0)+

nvl(sam.EH\_CSRS,0)+nvl(ncm.EH\_CSRS,0)))\*100),2) as EH\_GI\_PER,

sum( nvl(sam.SAM\_GI\_L\_MATCH,0) + nvl(ncm.NCM\_GI\_L\_MATCH,0)) as SAM\_GI\_L\_MATCH,

sum( nvl(sam.SAM\_GI\_NL\_MATCH,0) + nvl(ncm.NCM\_GI\_NL\_MATCH,0)) as SAM\_GI\_NL\_MATCH,

sum( nvl(sam.SAM\_GI\_MIS\_MATCH,0)+ nvl(ncm.NCM\_GI\_ISSUES,0)) as SAM\_GI\_MIS\_MATCH,

round( (((sum(nvl(sam.SAM\_GI\_L\_MATCH,0)+nvl(ncm.NCM\_GI\_L\_MATCH,0)))\*2)/(sum( nvl(sam.GI\_L\_CSRS,0)+nvl(ncm.GI\_L\_CSRS,0)+

nvl(sam.SAM\_CSRS,0)+nvl(ncm.NCM\_CSRS,0)))\*100),2) as SAM\_GI\_PER,

sum( nvl(sam.BRIX\_GI\_L\_MATCH,0)+ nvl(ncm.BRIX\_GI\_L\_MATCH,0)) as BRIX\_GI\_L\_MATCH,

sum( nvl(sam.BRIX\_GI\_NL\_MATCH,0) + nvl(ncm.BRIX\_GI\_NL\_MATCH,0)) as BRIX\_GI\_NL\_MATCH,

sum( nvl(sam.BRIX\_GI\_MIS\_MATCH,0) +nvl(ncm.BRIX\_GI\_ISSUES,0)) as BRIX\_GI\_MIS\_MATCH,

round( (((sum( nvl(sam.BRIX\_GI\_L\_MATCH,0)+nvl(ncm.BRIX\_GI\_L\_MATCH,0)))\*2)/(sum( nvl(sam.GI\_L\_CSRS,0)+ nvl(ncm.GI\_L\_CSRS,0)+

nvl(sam.BRIX\_CSRS,0)+nvl(ncm.BRIX\_CSRS,0)))\*100),2) as BRIX\_GI\_PER,

sum( nvl(sam.GI\_CSRS,0)+ nvl(ncm.GI\_CSRS,0)) as GI\_CSRS,

sum( nvl(sam.EH\_GI\_MATCH,0)+ nvl(ncm.EH\_GI\_MATCH,0)) as EH\_GI\_MATCH,

sum(sam.BRIX\_GI\_MATCH+ncm.BRIX\_GI\_MATCH) as BRIX\_GI\_MATCH,

sum( nvl(sam.SAM\_GI\_MATCH,0)+ nvl(ncm.NCM\_GI\_MATCH,0)) as SAM\_GI\_MATCH,

sum(sam.SAM\_EH\_GI\_ISSUES+ nvl(ncm.EH\_GI\_ISSUES,0)) as SAM\_EH\_GI\_ISSUES,

sum(ncm.HPOV\_CSRS+0) as HPOV\_CSRS,

sum(ncm.NCM\_HPOV\_ISSUES+0) as NCM\_HPOV\_ISSUES,

sum(ncm.ncm\_hpov\_match+0) as ncm\_hpov\_match,

ROUND (

( ( (SUM (

NVL (ncm.NCM\_HPOV\_MATCH, 0)))

\* 2)

/ (SUM (

NVL (sam.GI\_L\_CSRS, 0)

+ NVL (ncm.GI\_L\_CSRS, 0)

+ NVL (ncm.HPOV\_CSRS, 0)))

\* 100),

2)

AS NCM\_HPOV\_PER,

sum(ncm.hpov\_gi\_match+0) as hpov\_gi\_match,

sum(ncm.hpov\_gi\_l\_match+0) as hpov\_gi\_l\_match,

sum(ncm.hpov\_gi\_nl\_match+0) as hpov\_gi\_nl\_match,

sum(ncm.hpov\_gi\_issues+0) as hpov\_gi\_issues,

ROUND (

( ( (SUM (

NVL (ncm.HPOV\_GI\_L\_MATCH, 0)))

\* 2)

/ (SUM (

NVL (sam.GI\_L\_CSRS, 0)

+ NVL (ncm.GI\_L\_CSRS, 0)

+ NVL (ncm.HPOV\_CSRS, 0)))

\* 100),

2)

AS HPOV\_GI\_PER

from NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM ncm, SAM\_EH\_GI\_CSR\_REG\_SUMM sam

where ncm.region=sam.region and ncm.area=sam.area

group by rollup ( ncm.area,ncm.region)

order by NCM.area,ncm.region

;

--------------------------------------------------------

-- DDL for View SAM\_NCM\_GI\_EH\_BRIX\_BY\_CLLI\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."SAM\_NCM\_GI\_EH\_BRIX\_BY\_CLLI\_VW" ("CSR\_VENDOR", "CSR\_DEVICE\_NAME", "SAM\_CSR\_DEVICE\_NAME", "SAM\_REACHABILITY", "SAM\_RESYNC\_STATUS", "SAM\_RESYNC\_STATE", "SAM\_SERVER\_NAME", "SAM\_LAST\_RESYNC\_START", "SAM\_LAST\_RESYNC\_END", "GI\_DEVICE\_NAME", "LIVE\_IN\_XNG", "EQ\_STATUS", "MATCH\_CODE", "MATCH\_STATUS", "EH\_CSR\_HOSTNAME", "EH\_CSR\_IPADDRESS", "EQUIP\_INST\_ID", "AREA", "REGION", "MARKET", "LEAF\_DOMAIN\_INST\_ID", "CLLI", "BRIX\_CSR\_SITE", "SAM\_CSR\_DEVICE\_IP", "NCM\_CSR\_VLAN\_PARTITION", "NCM\_CSR\_VLAN\_DEVICE\_IP", "HP\_CSR\_HOSTNAME", "HP\_CSR\_IP", "DESCR") AS

select

csr\_vendor

,csr\_device\_name

,sam\_csr\_device\_name

,sam\_reachability

,sam\_resync\_status

,sam\_resync\_state

,sam\_server\_name

,sam\_last\_resync\_start

,sam\_last\_resync\_end

,gi\_device\_name

,live\_in\_xng

,eq\_status

,match\_code

,match\_status

,eh\_csr\_hostname

,eh\_csr\_ipaddress

,equip\_inst\_id

,area

,region

,market

,leaf\_domain\_inst\_id

,clli

,brix\_csr\_site

,SAM\_CSR\_device\_IP

,ncm\_csr\_vlan\_partition

,ncm\_csr\_vlan\_device\_ip

,hp\_csr\_hostname

,hp\_csr\_ip

,descr

from ( select distinct decode(n.MATCH\_CODE, 'Xng Only', 'Granite Only', n.MATCH\_CODE) MATCH\_CODE1, n.\*, SER.NODE\_ID SAM\_CSR\_DEVICE\_IP, I.DESCRIPTION as issues,

'' ncm\_csr\_vlan\_partition,

'' ncm\_csr\_vlan\_device\_ip,

'' hp\_csr\_hostname,

'' hp\_csr\_ip,

'' descr

from xng\_reports.SAM\_GI\_AL\_CSR\_AUDIT n

left outer join xng\_reports.sam\_ebh\_rtr ser on N.SAM\_CSR\_DEVICE\_NAME = SER.NODE\_NAME

left outer join xng\_reports.csr\_device\_audit\_issues d

on N.CSR\_DEVICE\_NAME = d.CSR\_DEVICE\_NAME

left outer join xng\_reports.csr\_issue

i on I.CSR\_ISSUE\_ID = d.CSR\_ISSUE\_ID )

union

select

csr\_vendor

,csr\_device\_name

,ncm\_csr\_hostname

,'' sam\_reachability

,'' sam\_resync\_status

,'' sam\_resync\_state

,'' sam\_server\_name

,null sam\_last\_resync\_start

,null sam\_last\_resync\_end

,gi\_device\_name

,live\_in\_xng

,eq\_status

,match\_code

,match\_status

,eh\_csr\_hostname

,eh\_csr\_ipaddress

,equip\_inst\_id

,area

,region

,market

,leaf\_domain\_inst\_id

,clli

,brix\_csr\_site

,NCM\_CSR\_device\_IP

,ncm\_csr\_vlan\_partition

,ncm\_csr\_vlan\_device\_ip

,hp\_csr\_hostname

,hp\_csr\_ip

,descr

from

(

select distinct CDM.AREA, CDM.REGION, CDM.MARKET, CDM.LEAF\_DOMAIN\_INST\_ID, cdm.clli, decode(w.MATCH\_CODE, 'Xng Only', 'Granite Only', w.MATCH\_CODE) MATCH\_CODE1, EI.STATUS eq\_status, w.\*, ci.description match\_status

from xng\_reports.NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT w

left outer join vzwnet.equip\_inst ei on W.EQUIP\_INST\_ID = EI.EQUIP\_INST\_ID

left outer join xng\_reports.clli\_domain\_map\_v cdm on cdm.clli = substr(W.CSR\_DEVICE\_NAME, 1,8)

left outer join xng\_reports.csr\_device\_audit\_issues cdi on w.csr\_device\_name = cdi.csr\_device\_name

left outer join xng\_reports.csr\_issue ci

on CI.CSR\_ISSUE\_ID= CDI.CSR\_ISSUE\_ID

)

;

--------------------------------------------------------

-- DDL for View SAM\_NCM\_GI\_EH\_BRIX\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."SAM\_NCM\_GI\_EH\_BRIX\_VW" ("CSR\_VENDOR", "CSR\_DEVICE\_NAME", "SAM\_CSR\_DEVICE\_NAME", "SAM\_REACHABILITY", "SAM\_RESYNC\_STATUS", "SAM\_RESYNC\_STATE", "SAM\_SERVER\_NAME", "SAM\_LAST\_RESYNC\_START", "SAM\_LAST\_RESYNC\_END", "GI\_DEVICE\_NAME", "LIVE\_IN\_XNG", "EQ\_STATUS", "MATCH\_CODE", "MATCH\_STATUS", "EH\_CSR\_HOSTNAME", "EH\_CSR\_IPADDRESS", "EQUIP\_INST\_ID", "AREA", "REGION", "MARKET", "LEAF\_DOMAIN\_INST\_ID", "CLLI", "BRIX\_CSR\_SITE", "SAM\_CSR\_DEVICE\_IP", "NCM\_CSR\_VLAN\_PARTITION", "NCM\_CSR\_VLAN\_DEVICE\_IP", "HP\_CSR\_HOSTNAME", "HP\_CSR\_IP", "DESCR") AS

select

csr\_vendor

,csr\_device\_name

,sam\_csr\_device\_name

,sam\_reachability

,sam\_resync\_status

,sam\_resync\_state

,sam\_server\_name

,sam\_last\_resync\_start

,sam\_last\_resync\_end

,gi\_device\_name

,live\_in\_xng

,eq\_status

,match\_code

,match\_status

,eh\_csr\_hostname

,eh\_csr\_ipaddress

,equip\_inst\_id

,area

,region

,market

,leaf\_domain\_inst\_id

,clli

,brix\_csr\_site

,SAM\_CSR\_device\_IP

,ncm\_csr\_vlan\_partition

,ncm\_csr\_vlan\_device\_ip

,hp\_csr\_hostname

,hp\_csr\_ip

,descr

from ( select distinct decode(n.MATCH\_CODE, 'Xng Only', 'Granite Only', n.MATCH\_CODE) MATCH\_CODE1, n.\*, SER.NODE\_ID SAM\_CSR\_DEVICE\_IP, I.DESCRIPTION as issues,

'' ncm\_csr\_vlan\_partition,

'' ncm\_csr\_vlan\_device\_ip,

'' hp\_csr\_hostname,

'' hp\_csr\_ip,

'' descr

from xng\_reports.SAM\_GI\_AL\_CSR\_AUDIT n

left outer join xng\_reports.sam\_ebh\_rtr ser on N.SAM\_CSR\_DEVICE\_NAME = SER.NODE\_NAME

left outer join xng\_reports.csr\_device\_audit\_issues d

on N.CSR\_DEVICE\_NAME = d.CSR\_DEVICE\_NAME

left outer join xng\_reports.csr\_issue

i on I.CSR\_ISSUE\_ID = d.CSR\_ISSUE\_ID where N.REGION= 'Great Plains' and N.CLLI = 'DESMIA' )

union

select

csr\_vendor

,csr\_device\_name

,ncm\_csr\_hostname

,'' sam\_reachability

,'' sam\_resync\_status

,'' sam\_resync\_state

,'' sam\_server\_name

,null sam\_last\_resync\_start

,null sam\_last\_resync\_end

,gi\_device\_name

,live\_in\_xng

,eq\_status

,match\_code

,match\_status

,eh\_csr\_hostname

,eh\_csr\_ipaddress

,equip\_inst\_id

,area

,region

,market

,leaf\_domain\_inst\_id

,clli

,brix\_csr\_site

,NCM\_CSR\_device\_IP

,ncm\_csr\_vlan\_partition

,ncm\_csr\_vlan\_device\_ip

,hp\_csr\_hostname

,hp\_csr\_ip

,descr

from

(

select distinct CDM.AREA, CDM.REGION, CDM.MARKET, CDM.LEAF\_DOMAIN\_INST\_ID, cdm.clli, decode(w.MATCH\_CODE, 'Xng Only', 'Granite Only', w.MATCH\_CODE) MATCH\_CODE1, EI.STATUS eq\_status, w.\*, ci.description match\_status

from xng\_reports.NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT w

left outer join vzwnet.equip\_inst ei on W.EQUIP\_INST\_ID = EI.EQUIP\_INST\_ID

left outer join xng\_reports.clli\_domain\_map\_v cdm on cdm.clli = substr(W.CSR\_DEVICE\_NAME, 1,8)

left outer join xng\_reports.csr\_device\_audit\_issues cdi on w.csr\_device\_name = cdi.csr\_device\_name

left outer join xng\_reports.csr\_issue ci

on CI.CSR\_ISSUE\_ID= CDI.CSR\_ISSUE\_ID

)

;

--------------------------------------------------------

-- DDL for View SAM\_NCM\_REGION\_AGGREGATE\_VIEW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."SAM\_NCM\_REGION\_AGGREGATE\_VIEW" ("AREA", "MARKET", "REGION", "CLLI\_6", "EH\_CSRS", "SAM\_CSRS", "BRIX\_CSRS", "GI\_L\_CSRS", "GI\_NL\_CSRS", "EH\_GI\_L\_MATCH", "EH\_GI\_NL\_MATCH", "EH\_GI\_MIS\_MATCH", "EH\_GI\_PER", "SAM\_GI\_L\_MATCH", "SAM\_GI\_NL\_MATCH", "SAM\_GI\_MIS\_MATCH", "SAM\_GI\_PER", "BRIX\_GI\_L\_MATCH", "BRIX\_GI\_NL\_MATCH", "BRIX\_GI\_MIS\_MATCH", "BRIX\_GI\_PER", "GI\_CSRS", "EH\_GI\_MATCH", "BRIX\_GI\_MATCH", "SAM\_GI\_MATCH", "SAM\_EH\_GI\_ISSUES", "HPOV\_CSRS", "NCM\_HPOV\_ISSUES", "NCM\_HPOV\_MATCH", "NCM\_HPOV\_PER", "HPOV\_GI\_MATCH", "HPOV\_GI\_L\_MATCH", "HPOV\_GI\_NL\_MATCH", "HPOV\_GI\_ISSUES", "HPOV\_GI\_PER") AS

(

(select sam.area,sam.MARKET,sam.region,sam.CLLI\_6,sam.EH\_CSRS,sam.SAM\_CSRS,sam.BRIX\_CSRS,sam.GI\_L\_CSRS,sam.GI\_NL\_CSRS,sam.EH\_GI\_L\_MATCH,sam.EH\_GI\_NL\_MATCH,sam.EH\_GI\_MIS\_MATCH,sam.EH\_GI\_PER,sam.SAM\_GI\_L\_MATCH,sam.SAM\_GI\_NL\_MATCH,sam.SAM\_GI\_MIS\_MATCH,sam.SAM\_GI\_PER,sam.BRIX\_GI\_L\_MATCH,sam.BRIX\_GI\_NL\_MATCH,sam.BRIX\_GI\_MIS\_MATCH,sam.BRIX\_GI\_PER,sam.GI\_CSRS,sam.EH\_GI\_MATCH,sam.BRIX\_GI\_MATCH,sam.SAM\_GI\_MATCH,sam.SAM\_EH\_GI\_ISSUES,0 as HPOV\_CSRS,0 as NCM\_HPOV\_ISSUES,0 as ncm\_hpov\_match,0 as NCM\_HPOV\_PER,0 as hpov\_gi\_match,0 as hpov\_gi\_l\_match,0 as hpov\_gi\_nl\_match,0 as hpov\_gi\_issues,0 as hpov\_gi\_per from xng\_reports.SAM\_EH\_GI\_CSR\_CLLI\_SUMM sam

UNION

select ncm.area,ncm.MARKET,ncm.region,ncm.CLLI\_6,ncm.EH\_CSRS,ncm.NCM\_CSRS as SAM\_CSRS,ncm.BRIX\_CSRS,ncm.GI\_L\_CSRS,ncm.GI\_NL\_CSRS,ncm.EH\_GI\_L\_MATCH,ncm.EH\_GI\_NL\_MATCH,ncm.EH\_GI\_ISSUES as EH\_GI\_MIS\_MATCH,ncm.EH\_GI\_PER,ncm.NCM\_GI\_L\_MATCH,ncm.NCM\_GI\_NL\_MATCH,ncm.NCM\_GI\_ISSUES,ncm.NCM\_GI\_PER,ncm.BRIX\_GI\_L\_MATCH,ncm.BRIX\_GI\_NL\_MATCH,ncm.BRIX\_GI\_ISSUES,ncm.BRIX\_GI\_PER,ncm.GI\_CSRS,ncm.EH\_GI\_MATCH,ncm.BRIX\_GI\_MATCH,ncm.NCM\_GI\_MATCH,0 as SAM\_EH\_GI\_ISSUES,NCM.HPOV\_CSRS, NCM.NCM\_HPOV\_ISSUES, ncm.ncm\_hpov\_match, NCM.NCM\_HPOV\_PER,

ncm.hpov\_gi\_match,ncm.hpov\_gi\_l\_match,ncm.hpov\_gi\_nl\_match,ncm.hpov\_gi\_issues,ncm.hpov\_gi\_per from xng\_reports.NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM ncm

UNION

select ncm.area,ncm.market,ncm.region,ncm.clli\_6,

sum(nvl(sam.EH\_CSRS,0)+nvl(ncm.EH\_CSRS,0)) as EH\_CSRS,

sum(nvl(sam.SAM\_CSRS,0)+nvl(ncm.NCM\_CSRS,0)) as SAM\_CSRS,

sum(nvl(sam.BRIX\_CSRS,0)+nvl(ncm.BRIX\_CSRS,0)) as BRIX\_CSRS,

sum(nvl(sam.GI\_L\_CSRS,0)+nvl(ncm.GI\_L\_CSRS,0)) as GI\_L\_CSRS,

sum( nvl(sam.GI\_NL\_CSRS,0) +nvl(ncm.GI\_NL\_CSRS,0)) as GI\_NL\_CSRS,

sum( nvl(sam.EH\_GI\_L\_MATCH,0) + nvl(ncm.EH\_GI\_L\_MATCH,0)) as EH\_GI\_L\_MATCH,

sum( nvl(sam.EH\_GI\_NL\_MATCH,0) + nvl(ncm.EH\_GI\_NL\_MATCH,0) ) as EH\_GI\_NL\_MATCH,

sum( nvl(sam.EH\_GI\_MIS\_MATCH,0) + nvl(ncm.EH\_GI\_ISSUES,0)) as EH\_GI\_MIS\_MATCH,

round ((((sum( nvl(sam.EH\_GI\_L\_MATCH,0)+nvl(ncm.EH\_GI\_L\_MATCH,0)))\*2)/(sum(nvl(sam.GI\_L\_CSRS,0)+nvl(ncm.GI\_L\_CSRS,0)+

nvl(sam.EH\_CSRS,0)+nvl(ncm.EH\_CSRS,0)))\*100),2) as EH\_GI\_PER,

sum( nvl(sam.SAM\_GI\_L\_MATCH,0) + nvl(ncm.NCM\_GI\_L\_MATCH,0)) as SAM\_GI\_L\_MATCH,

sum( nvl(sam.SAM\_GI\_NL\_MATCH,0) + nvl(ncm.NCM\_GI\_NL\_MATCH,0)) as SAM\_GI\_NL\_MATCH,

sum( nvl(sam.SAM\_GI\_MIS\_MATCH,0)+ nvl(ncm.NCM\_GI\_ISSUES,0)) as SAM\_GI\_MIS\_MATCH,

round( (((sum(nvl(sam.SAM\_GI\_L\_MATCH,0)+nvl(ncm.NCM\_GI\_L\_MATCH,0)))\*2)/(sum( nvl(sam.GI\_L\_CSRS,0)+nvl(ncm.GI\_L\_CSRS,0)+

nvl(sam.SAM\_CSRS,0)+nvl(ncm.NCM\_CSRS,0)))\*100),2) as SAM\_GI\_PER,

sum( nvl(sam.BRIX\_GI\_L\_MATCH,0)+ nvl(ncm.BRIX\_GI\_L\_MATCH,0)) as BRIX\_GI\_L\_MATCH,

sum( nvl(sam.BRIX\_GI\_NL\_MATCH,0) + nvl(ncm.BRIX\_GI\_NL\_MATCH,0)) as BRIX\_GI\_NL\_MATCH,

sum( nvl(sam.BRIX\_GI\_MIS\_MATCH,0) +nvl(ncm.BRIX\_GI\_ISSUES,0)) as BRIX\_GI\_MIS\_MATCH,

round( (((sum( nvl(sam.BRIX\_GI\_L\_MATCH,0)+nvl(ncm.BRIX\_GI\_L\_MATCH,0)))\*2)/(sum( nvl(sam.GI\_L\_CSRS,0)+ nvl(ncm.GI\_L\_CSRS,0)+

nvl(sam.BRIX\_CSRS,0)+nvl(ncm.BRIX\_CSRS,0)))\*100),2) as BRIX\_GI\_PER,

sum( nvl(sam.GI\_CSRS,0)+ nvl(ncm.GI\_CSRS,0)) as GI\_CSRS,

sum( nvl(sam.EH\_GI\_MATCH,0)+ nvl(ncm.EH\_GI\_MATCH,0)) as EH\_GI\_MATCH,

sum(sam.BRIX\_GI\_MATCH+ncm.BRIX\_GI\_MATCH) as BRIX\_GI\_MATCH,

sum( nvl(sam.SAM\_GI\_MATCH,0)+ nvl(ncm.NCM\_GI\_MATCH,0)) as SAM\_GI\_MATCH,

sum(sam.SAM\_EH\_GI\_ISSUES+0) as SAM\_EH\_GI\_ISSUES,

0,0,0,0,0,0,0,0,0

from NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM ncm, SAM\_EH\_GI\_CSR\_CLLI\_SUMM sam

where ncm.market=sam.market and sam.clli\_6 = ncm.clli\_6 GROUP BY ncm.area,ncm.market,ncm.region,ncm.clli\_6

)

)

MINUS

(select sam.area,sam.MARKET,sam.region,sam.CLLI\_6,sam.EH\_CSRS,sam.SAM\_CSRS,sam.BRIX\_CSRS,sam.GI\_L\_CSRS,sam.GI\_NL\_CSRS,sam.EH\_GI\_L\_MATCH,sam.EH\_GI\_NL\_MATCH,sam.EH\_GI\_MIS\_MATCH,sam.EH\_GI\_PER,sam.SAM\_GI\_L\_MATCH,sam.SAM\_GI\_NL\_MATCH,sam.SAM\_GI\_MIS\_MATCH,sam.SAM\_GI\_PER,sam.BRIX\_GI\_L\_MATCH,sam.BRIX\_GI\_NL\_MATCH,sam.BRIX\_GI\_MIS\_MATCH,sam.BRIX\_GI\_PER,sam.GI\_CSRS,sam.EH\_GI\_MATCH,sam.BRIX\_GI\_MATCH,sam.SAM\_GI\_MATCH,sam.SAM\_EH\_GI\_ISSUES,0,0,0,0,0,0,0,0,0 from xng\_reports.SAM\_EH\_GI\_CSR\_CLLI\_SUMM sam,xng\_reports.NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM ncm

where sam.clli\_6=ncm.clli\_6)

MINUS

(select ncm.area,ncm.MARKET,ncm.region,ncm.CLLI\_6,ncm.EH\_CSRS,ncm.NCM\_CSRS,ncm.BRIX\_CSRS,ncm.GI\_L\_CSRS,ncm.GI\_NL\_CSRS,ncm.EH\_GI\_L\_MATCH,ncm.EH\_GI\_NL\_MATCH,ncm.EH\_GI\_ISSUES,ncm.EH\_GI\_PER,ncm.NCM\_GI\_L\_MATCH,ncm.NCM\_GI\_NL\_MATCH,ncm.NCM\_GI\_ISSUES,ncm.NCM\_GI\_PER,ncm.BRIX\_GI\_L\_MATCH,ncm.BRIX\_GI\_NL\_MATCH,ncm.BRIX\_GI\_ISSUES,ncm.BRIX\_GI\_PER,ncm.GI\_CSRS,ncm.EH\_GI\_MATCH,ncm.BRIX\_GI\_MATCH,ncm.NCM\_GI\_MATCH,0,0,0,0,0,0,0,0,0,0 as SAM\_EH\_GI\_ISSUES from xng\_reports.SAM\_EH\_GI\_CSR\_CLLI\_SUMM sam,xng\_reports.NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM ncm

where sam.clli\_6=ncm.clli\_6)

order by area,MARKET, CLLI\_6

;

--------------------------------------------------------

-- DDL for View SWITCH\_VENDOR\_MAP

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."SWITCH\_VENDOR\_MAP" ("SWITCH\_ID", "DOMAIN\_NAME", "VENDOR") AS

WITH all\_ref\_switches AS

(SELECT DISTINCT di.domain\_name, di.domain\_inst\_id,

SUBSTR (vvxx.vsm\_device\_name, 1, 6)

|| SUBSTR (vvxx.vsm\_device\_name, 11, 2) AS switch\_id,

'NORTEL' AS vendor

FROM xng\_reports.vzw\_vsm\_xng\_xref vvxx,

xng\_reports.vzw\_network\_org vno,

vzwnet.domain\_inst di,

xng\_reports.bsm\_mtx\_map bm

WHERE vno.area = vvxx.area

AND vno.region = vvxx.region

AND di.domain\_name = vno.region\_domain\_name

AND bm.vsm\_device\_name\_mtx = vvxx.vsm\_device\_name

AND bm.bsm\_status = 'Live'

AND bm.mtx\_status = 'Live'

AND bm.vsm\_device\_name\_bsm LIKE '%BSM'

UNION

SELECT di.domain\_name, di.domain\_inst\_id, esm.switch\_id,

'LUCENT' AS vendor

FROM xng\_reports.ecp\_switch\_map esm, vzwnet.domain\_inst di

WHERE esm.xng\_domain\_inst\_id = di.domain\_inst\_id

UNION

SELECT DISTINCT di.domain\_name, di.domain\_inst\_id,

SUBSTR (vvxx.vsm\_device\_name, 1, 6)

|| SUBSTR (vvxx.vsm\_device\_name, 11, 2) AS switch\_id,

'MOTOROLA' AS vendor

FROM xng\_reports.vzw\_vsm\_xng\_xref vvxx,

xng\_reports.vzw\_network\_org vno,

vzwnet.domain\_inst di,

xng\_reports.bsm\_mtx\_map bmm

WHERE vno.area = vvxx.area

AND vno.region = vvxx.region

AND di.domain\_name = vno.region\_domain\_name

AND SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 16) =

SUBSTR (vvxx.vsm\_device\_name, 1, 16)

AND bmm.bsm\_status = 'Live'

AND bmm.mtx\_status = 'Live'

AND bmm.vsm\_device\_name\_bsm LIKE '%OMCR'

UNION

SELECT DISTINCT di.domain\_name, di.domain\_inst\_id,

a.vsm\_device\_name\_aems AS switch\_id,

'MOTOROLA' AS vendor

FROM xng\_reports.moto\_vsm\_name\_aems a,

vzwnet.domain\_inst di,

xng\_reports.vzw\_network\_org vno,

xng\_reports.vzw\_vsm\_xng\_xref vvxx

WHERE SUBSTR (a.vsm\_device\_name\_aems, 1, 8) =

SUBSTR (vvxx.vsm\_device\_name, 1, 8)

AND SUBSTR (vvxx.vsm\_device\_name, 1, 16) LIKE

'\_\_\_\_\_\_\_\_\_\_\_\_%MTX%'

AND vno.region = vvxx.region

AND di.domain\_name = vno.region\_domain\_name

UNION

SELECT DISTINCT di.domain\_name, di.domain\_inst\_id,

a.vsm\_device\_name\_ems AS switch\_id,

'NORTEL' AS vendor

FROM xng\_reports.vsm\_sane\_name\_ems a,

vzwnet.domain\_inst di,

xng\_reports.vzw\_network\_org vno,

xng\_reports.vzw\_vsm\_xng\_xref vvxx

WHERE SUBSTR (a.vsm\_device\_name\_ems, 1, 8) =

SUBSTR (vvxx.vsm\_device\_name, 1, 8)

AND vvxx.vsm\_device\_name LIKE '%MTX'

AND vno.region = vvxx.region

AND di.domain\_name = vno.region\_domain\_name)

SELECT afs.switch\_id, afs.domain\_name, afs.vendor

FROM all\_ref\_switches afs

ORDER BY afs.domain\_name, afs.switch\_id

;

--------------------------------------------------------

-- DDL for View T1\_MULTIPLE\_TERMINATIONS

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."T1\_MULTIPLE\_TERMINATIONS" ("SWITCH\_PATH\_T1\_TESTABILITY\_ID", "SWITCH\_PATH\_ISSUE\_ID") AS

WITH multiple\_a\_termination\_points AS

(SELECT sptt.device\_name AS switch\_id,

lnvxa.xng\_cell\_num AS cell\_number,

COUNT (DISTINCT cpi.a\_side\_site\_id) AS a\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.lucent\_ne\_vs\_xng\_audit lnvxa

ON sptt.circ\_path\_inst\_id = lnvxa.circ\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'ALU' AND lnvxa.xng\_cell\_num IS NOT NULL

GROUP BY sptt.device\_name, lnvxa.xng\_cell\_num

HAVING COUNT (DISTINCT cpi.a\_side\_site\_id) > 1

UNION

SELECT nnvxa.vsm\_device\_name\_mtx AS switch\_id,

nnvxa.xng\_dcg\_number AS cell\_number,

COUNT (DISTINCT cpi.a\_side\_site\_id) AS a\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

ON sptt.circ\_path\_inst\_id = nnvxa.circ\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'NT'

AND nnvxa.ne\_span\_type = 'VOICE\_SPAN'

AND nnvxa.xng\_dcg\_number IS NOT NULL

GROUP BY nnvxa.vsm\_device\_name\_mtx, nnvxa.xng\_dcg\_number

HAVING COUNT (DISTINCT cpi.a\_side\_site\_id) > 1

UNION

SELECT nnvxa.vsm\_device\_name\_ems AS switch\_id,

nnvxa.xng\_dcg\_number AS cell\_number,

COUNT (DISTINCT cpi.a\_side\_site\_id) AS a\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

ON sptt.circ\_path\_inst\_id = nnvxa.circ\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'NT'

AND nnvxa.ne\_span\_type = 'DATA\_SPAN'

AND nnvxa.xng\_dcg\_number IS NOT NULL

GROUP BY nnvxa.vsm\_device\_name\_ems, nnvxa.xng\_dcg\_number

HAVING COUNT (DISTINCT cpi.a\_side\_site\_id) > 1

UNION

SELECT mcnvxa.mtx AS switch\_id,

TO\_NUMBER (mcnvxa.xng\_bts\_number) AS cell\_number,

COUNT (DISTINCT cpi.a\_side\_site\_id) AS a\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.moto\_cdma\_ne\_vs\_xng\_audit mcnvxa

ON sptt.circ\_path\_inst\_id = mcnvxa.xng\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'MOTO' AND mcnvxa.xng\_bts\_number IS NOT NULL

GROUP BY mcnvxa.mtx, TO\_NUMBER (mcnvxa.xng\_bts\_number)

HAVING COUNT (DISTINCT cpi.a\_side\_site\_id) > 1

UNION

SELECT menvxa.aems || menvxa.xng\_ipbscdo\_number AS switch\_id,

TO\_NUMBER (menvxa.xng\_mccdo\_identifier) AS cell\_number,

COUNT (DISTINCT cpi.a\_side\_site\_id) AS a\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit menvxa

ON sptt.circ\_path\_inst\_id = menvxa.xng\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'MOTO'

AND menvxa.xng\_mccdo\_identifier IS NOT NULL

GROUP BY menvxa.aems || menvxa.xng\_ipbscdo\_number,

TO\_NUMBER (menvxa.xng\_mccdo\_identifier)

HAVING COUNT (DISTINCT cpi.a\_side\_site\_id) > 1),

multiple\_z\_termination\_points AS

(SELECT sptt.device\_name AS switch\_id,

lnvxa.xng\_cell\_num AS cell\_number,

COUNT (DISTINCT cpi.z\_side\_site\_id) AS z\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.lucent\_ne\_vs\_xng\_audit lnvxa

ON sptt.circ\_path\_inst\_id = lnvxa.circ\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'ALU' AND lnvxa.xng\_cell\_num IS NOT NULL

GROUP BY sptt.device\_name, lnvxa.xng\_cell\_num

HAVING COUNT (DISTINCT cpi.z\_side\_site\_id) > 1

UNION

SELECT nnvxa.vsm\_device\_name\_mtx AS switch\_id,

nnvxa.xng\_dcg\_number AS cell\_number,

COUNT (DISTINCT cpi.z\_side\_site\_id) AS z\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

ON sptt.circ\_path\_inst\_id = nnvxa.circ\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'NT'

AND nnvxa.ne\_span\_type = 'VOICE\_SPAN'

AND nnvxa.xng\_dcg\_number IS NOT NULL

GROUP BY nnvxa.vsm\_device\_name\_mtx, nnvxa.xng\_dcg\_number

HAVING COUNT (DISTINCT cpi.z\_side\_site\_id) > 1

UNION

SELECT nnvxa.vsm\_device\_name\_ems AS switch\_id,

nnvxa.xng\_dcg\_number AS cell\_number,

COUNT (DISTINCT cpi.z\_side\_site\_id) AS z\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

ON sptt.circ\_path\_inst\_id = nnvxa.circ\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'NT'

AND nnvxa.ne\_span\_type = 'DATA\_SPAN'

AND nnvxa.xng\_dcg\_number IS NOT NULL

GROUP BY nnvxa.vsm\_device\_name\_ems, nnvxa.xng\_dcg\_number

HAVING COUNT (DISTINCT cpi.z\_side\_site\_id) > 1

UNION

SELECT mcnvxa.mtx AS switch\_id,

TO\_NUMBER (mcnvxa.xng\_bts\_number) AS cell\_number,

COUNT (DISTINCT cpi.z\_side\_site\_id) AS z\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.moto\_cdma\_ne\_vs\_xng\_audit mcnvxa

ON sptt.circ\_path\_inst\_id = mcnvxa.xng\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'MOTO' AND mcnvxa.xng\_bts\_number IS NOT NULL

GROUP BY mcnvxa.mtx, TO\_NUMBER (mcnvxa.xng\_bts\_number)

HAVING COUNT (DISTINCT cpi.z\_side\_site\_id) > 1

UNION

SELECT menvxa.aems || menvxa.xng\_ipbscdo\_number AS switch\_id,

TO\_NUMBER (menvxa.xng\_mccdo\_identifier) AS cell\_number,

COUNT (DISTINCT cpi.z\_side\_site\_id) AS z\_side\_site\_count

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit menvxa

ON sptt.circ\_path\_inst\_id = menvxa.xng\_path\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON sptt.circ\_path\_inst\_id = cpi.circ\_path\_inst\_id

WHERE sptt.vendor = 'MOTO'

AND menvxa.xng\_mccdo\_identifier IS NOT NULL

GROUP BY menvxa.aems || menvxa.xng\_ipbscdo\_number,

TO\_NUMBER (menvxa.xng\_mccdo\_identifier)

HAVING COUNT (DISTINCT cpi.z\_side\_site\_id) > 1)

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.lucent\_ne\_vs\_xng\_audit lnvxa

ON sptt.circ\_path\_inst\_id = lnvxa.circ\_path\_inst\_id

JOIN multiple\_a\_termination\_points matp

ON sptt.device\_name = matp.switch\_id

AND lnvxa.xng\_cell\_num = matp.cell\_number

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different a-side'

WHERE sptt.vendor = 'ALU'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.lucent\_ne\_vs\_xng\_audit lnvxa

ON sptt.circ\_path\_inst\_id = lnvxa.circ\_path\_inst\_id

JOIN multiple\_z\_termination\_points mztp

ON sptt.device\_name = mztp.switch\_id

AND lnvxa.xng\_cell\_num = mztp.cell\_number

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different z-side'

WHERE sptt.vendor = 'ALU'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

ON sptt.circ\_path\_inst\_id = nnvxa.circ\_path\_inst\_id

JOIN multiple\_a\_termination\_points matp

ON nnvxa.vsm\_device\_name\_mtx = matp.switch\_id

AND nnvxa.xng\_dcg\_number = matp.cell\_number

AND nnvxa.ne\_span\_type = 'VOICE\_SPAN'

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different a-side'

WHERE sptt.vendor = 'NT'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

ON sptt.circ\_path\_inst\_id = nnvxa.circ\_path\_inst\_id

JOIN multiple\_z\_termination\_points mztp

ON nnvxa.vsm\_device\_name\_mtx = mztp.switch\_id

AND nnvxa.xng\_dcg\_number = mztp.cell\_number

AND nnvxa.ne\_span\_type = 'VOICE\_SPAN'

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different a-side'

WHERE sptt.vendor = 'NT'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

ON sptt.circ\_path\_inst\_id = nnvxa.circ\_path\_inst\_id

JOIN multiple\_a\_termination\_points matp

ON nnvxa.vsm\_device\_name\_ems = matp.switch\_id

AND nnvxa.xng\_dcg\_number = matp.cell\_number

AND nnvxa.ne\_span\_type = 'DATA\_SPAN'

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different a-side'

WHERE sptt.vendor = 'NT'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.nortel\_ne\_vs\_xng\_audit nnvxa

ON sptt.circ\_path\_inst\_id = nnvxa.circ\_path\_inst\_id

JOIN multiple\_z\_termination\_points mztp

ON nnvxa.vsm\_device\_name\_ems = mztp.switch\_id

AND nnvxa.xng\_dcg\_number = mztp.cell\_number

AND nnvxa.ne\_span\_type = 'DATA\_SPAN'

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different a-side'

WHERE sptt.vendor = 'NT'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.moto\_cdma\_ne\_vs\_xng\_audit mcnvxa

ON sptt.circ\_path\_inst\_id = mcnvxa.xng\_path\_inst\_id

JOIN multiple\_a\_termination\_points matp

ON mcnvxa.mtx = matp.switch\_id

AND mcnvxa.xng\_bts\_number = matp.cell\_number

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different a-side'

WHERE sptt.vendor = 'MOTO'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.moto\_cdma\_ne\_vs\_xng\_audit mcnvxa

ON sptt.circ\_path\_inst\_id = mcnvxa.xng\_path\_inst\_id

JOIN multiple\_z\_termination\_points mztp

ON mcnvxa.mtx = mztp.switch\_id

AND mcnvxa.xng\_bts\_number = mztp.cell\_number

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different z-side'

WHERE sptt.vendor = 'MOTO'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit menvxa

ON sptt.circ\_path\_inst\_id = menvxa.xng\_path\_inst\_id

JOIN multiple\_a\_termination\_points matp

ON menvxa.aems || menvxa.xng\_ipbscdo\_number = matp.switch\_id

AND menvxa.xng\_mccdo\_identifier = matp.cell\_number

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different a-side'

WHERE sptt.vendor = 'MOTO'

UNION

SELECT sptt.switch\_path\_t1\_testability\_id, spi.switch\_path\_issue\_id

FROM xng\_reports.switch\_path\_t1\_testability\_wk sptt JOIN xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit menvxa

ON sptt.circ\_path\_inst\_id = menvxa.xng\_path\_inst\_id

JOIN multiple\_z\_termination\_points mztp

ON menvxa.aems || menvxa.xng\_ipbscdo\_number = mztp.switch\_id

AND menvxa.xng\_mccdo\_identifier = mztp.cell\_number

JOIN xng\_reports.switch\_path\_issue spi

ON spi.description =

'Other paths in switch/cell have different a-side'

WHERE sptt.vendor = 'MOTO'

;

--------------------------------------------------------

-- DDL for View USER\_INST\_NAME\_PARSE

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."USER\_INST\_NAME\_PARSE" ("XNG\_USER\_ID", "USER\_INST\_ID", "DEPARTMENT", "EMAIL\_ADDR", "FIRSTNAME", "MIDDLENAME", "LASTNAME", "FIRSTLEN", "MIDDLELEN", "LASTLEN") AS

SELECT xng\_user\_id,

user\_inst\_id,

department,

email\_addr,

name1 AS firstname,

CASE WHEN len3 = 0 THEN NULL ELSE name3 END middlename,

CASE WHEN len3 = 0 THEN name2 ELSE name4 END lastname,

len1 AS firstlen,

CASE WHEN len3 = 0 THEN 0 ELSE len3 END middlelen,

CASE WHEN len3 = 0 THEN len2 ELSE len4 END lastlen

FROM (SELECT xng\_user\_id,

user\_inst\_id,

department,

email\_addr,

name1,

name2,

name3,

name4,

NVL (len1, 0) len1,

NVL (len2, 0) len2,

NVL (len3, 0) len3,

NVL (len4, 0) len4

FROM (SELECT xng\_user\_id,

user\_inst\_id,

department,

email\_addr,

name1,

name2,

name3,

name4,

LENGTH (name1) len1,

LENGTH (name2) len2,

LENGTH (name3) len3,

LENGTH (name4) len4

FROM (SELECT xng\_user\_id,

user\_inst\_id,

department,

email\_addr,

LOWER(SUBSTR (xng\_fullname,

1,

INSTR (xng\_fullname,

' ',

1,

1)

- 1))

name1,

LOWER(SUBSTR (xng\_fullname,

INSTR (xng\_fullname,

' ',

1,

1)

+ 1))

name2,

LOWER(SUBSTR (

xng\_fullname,

INSTR (xng\_fullname,

' ',

1,

1)

+ 1,

( INSTR (xng\_fullname,

' ',

1,

2)

- INSTR (xng\_fullname,

' ',

1,

1)

- 1)

))

name3,

LOWER(SUBSTR (xng\_fullname,

INSTR (xng\_fullname,

' ',

1,

2)

+ 1))

name4

FROM (SELECT LOWER(RTRIM(LTRIM (

user\_name

)))

xng\_user\_id,

user\_inst\_id,

department,

email\_addr,

REPLACE (

REPLACE (

REPLACE (

REPLACE (

full\_name,

'.',

''

),

',',

''

),

'''',

''

),

'"'

)

AS xng\_fullname

FROM xperadmin.user\_inst))))

;

--------------------------------------------------------

-- DDL for View VLAN\_SUMM\_ALL

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."VLAN\_SUMM\_ALL" ("AREA", "REGION", "LIVE", "RPTD", "NCM\_NGMLS\_COMPLIANCE", "SAM\_NGMLS\_COMPLIANCE", "SAM\_CSR\_COMPLIANCE", "NCM\_CSR\_COMPLIANCE", "OVERALL\_COMPLIANCE") AS

WITH regions

AS (SELECT area, region

FROM domains\_leaf\_reporting dlr

WHERE area NOT IN ('OSS', 'NNO')

UNION

SELECT DISTINCT area, NULL

FROM domains\_leaf\_reporting dlr

WHERE area <> 'NNO' AND area <> 'OSS'

UNION

SELECT 'unknown', 'unknown' FROM DUAL

UNION

SELECT 'unknown', NULL FROM DUAL

UNION

SELECT 'zEnterprise', NULL FROM DUAL),

nnv

AS ( SELECT CASE

WHEN area IS NULL AND region IS NULL THEN 'zEnterprise' --= 'Grand Total'

ELSE DECODE (area, 'Unknown', 'unknown', area)

END

area,

DECODE (region, 'Unknown', 'unknown', region) region,

SUM (live) live,

SUM (rptd) rptd,

ROUND (AVG (vlan\_compliance), 2) vlan\_compliance

FROM (SELECT area,

region,

ne\_total\_ngmls\_vlan - matched\_nl\_ngmls\_vlan live,

matched\_l\_ngmls\_vlan rptd,

vlan\_compliance

FROM NCM\_NGMLS\_VLAN\_REGION\_SUMM\_WK)

GROUP BY area, region),

snv

AS ( SELECT CASE

WHEN area IS NULL AND region IS NULL THEN 'zEnterprise' --= 'Grand Total'

ELSE DECODE (area, 'Unknown', 'unknown', area)

END

area,

DECODE (region, 'Unknown', 'unknown', region) region,

SUM (live) live,

SUM (rptd) rptd,

ROUND (AVG (vlan\_compliance), 2) vlan\_compliance

FROM (SELECT area,

region,

ne\_total\_ngmls\_vlan - matched\_nl\_ngmls\_vlan live,

matched\_l\_ngmls\_vlan rptd,

vlan\_compliance

FROM SAM\_NGMLS\_VLAN\_REGION\_SUMM\_WK)

GROUP BY area, region),

scv

AS ( SELECT CASE

WHEN area IS NULL AND region IS NULL THEN 'zEnterprise' --= 'Grand Total'

ELSE DECODE (area, 'Unknown', 'unknown', area)

END

area,

DECODE (region, 'Unknown', 'unknown', region) region,

SUM (live) live,

SUM (rptd) rptd,

ROUND (AVG (vlan\_compliance), 2) vlan\_compliance

FROM (SELECT area,

region,

ne\_total\_csr\_vlan - matched\_nl\_csr\_vlan live,

matched\_l\_csr\_vlan rptd,

vlan\_compliance

FROM SAM\_CSR\_VLAN\_REGION\_SUMM\_WK)

GROUP BY area, region),

ncv

AS ( SELECT CASE

WHEN area IS NULL AND region IS NULL THEN 'zEnterprise' --= 'Grand Total'

ELSE DECODE (area, 'Unknown', 'unknown', area)

END

area,

DECODE (region, 'Unknown', 'unknown', region) region,

SUM (live) live,

SUM (rptd) rptd,

ROUND (AVG (vlan\_compliance), 2) vlan\_compliance

FROM (SELECT area,

region,

ne\_total\_csr\_vlan - matched\_nl\_csr\_vlan live,

matched\_l\_csr\_vlan rptd,

vlan\_compliance

FROM NCM\_CSR\_VLAN\_REGION\_SUMM\_WK)

GROUP BY area, region),

calc\_summ

AS (SELECT dlr.area,

dlr.region,

NVL (nn.live, 0)

+ NVL (sn.live, 0)

+ NVL (sc.live, 0)

+ NVL (nc.live, 0)

live,

NVL (nn.rptd, 0)

+ NVL (sn.rptd, 0)

+ NVL (sc.rptd, 0)

+ NVL (nc.rptd, 0)

rptd,

NVL (nn.vlan\_compliance, 0) ncm\_ngmls\_compliance,

NVL (sn.vlan\_compliance, 0) sam\_ngmls\_compliance,

NVL (sc.vlan\_compliance, 0) sam\_csr\_compliance,

NVL (nc.vlan\_compliance, 0) ncm\_csr\_compliance,

CASE

WHEN ( NVL (nn.rptd, 0)

+ NVL (sn.rptd, 0)

+ NVL (sc.rptd, 0)

+ NVL (nc.rptd, 0)) = 0

THEN

0

ELSE

ROUND (

( NVL (nn.rptd, 0)

+ NVL (sn.rptd, 0)

+ NVL (sc.rptd, 0)

+ NVL (nc.rptd, 0))

/ ( NVL (nn.live, 0)

+ NVL (sn.live, 0)

+ NVL (sc.live, 0)

+ NVL (nc.live, 0))

\* 100,

2)

END

overall\_compliance

FROM regions dlr

LEFT OUTER JOIN

nnv nn

ON ( ( UPPER (nn.area) = UPPER (dlr.area)

AND UPPER (nn.region) = UPPER (dlr.region))

OR ( UPPER (nn.area) = UPPER (dlr.area)

AND nn.region IS NULL

AND dlr.region IS NULL))

LEFT OUTER JOIN

snv sn

ON ( ( UPPER (sn.area) = UPPER (dlr.area)

AND UPPER (sn.region) = UPPER (dlr.region))

OR ( UPPER (sn.area) = UPPER (dlr.area)

AND sn.region IS NULL

AND dlr.region IS NULL))

LEFT OUTER JOIN

scv sc

ON ( ( UPPER (sc.area) = UPPER (dlr.area)

AND UPPER (sc.region) = UPPER (dlr.region))

OR ( UPPER (sc.area) = UPPER (dlr.area)

AND sc.region IS NULL

AND dlr.region IS NULL))

LEFT OUTER JOIN

ncv nc

ON ( ( UPPER (nc.area) = UPPER (dlr.area)

AND UPPER (nc.region) = UPPER (dlr.region))

OR ( UPPER (nc.area) = UPPER (dlr.area)

AND nc.region IS NULL

AND dlr.region IS NULL)))

SELECT "AREA",

"REGION",

"LIVE",

"RPTD",

"NCM\_NGMLS\_COMPLIANCE",

"SAM\_NGMLS\_COMPLIANCE",

"SAM\_CSR\_COMPLIANCE",

"NCM\_CSR\_COMPLIANCE",

"OVERALL\_COMPLIANCE"

FROM calc\_summ dlr

ORDER BY dlr.area, dlr.region

;

--------------------------------------------------------

-- DDL for View VPI\_GI\_AUDIT\_RPT\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."VPI\_GI\_AUDIT\_RPT\_V" ("TERRITORIES", "MARKETS", "SUB\_MARKETS", "CMA\_NO", "CMA\_NAME", "UACE", "UACE\_NAME", "SITE\_NAME", "NWID", "TOP\_CMA\_UACE", "GI\_SITE\_ID", "GI\_SITE\_NAME", "GI\_SITE\_TYPE", "GI\_SITE\_STATUS", "GI\_LOC\_CAT") AS

SELECT DISTINCT territories, markets, sub\_markets, cma\_no, cma\_name, uace, uace\_name, site\_name, nwid, top\_cma\_uace, gi\_site\_id, gi\_site\_name,

gi\_site\_type, gi\_site\_status, gi\_loc\_cat

FROM xng\_reports.vpi\_gi\_audit\_unq

;

--------------------------------------------------------

-- DDL for View VSM\_DEVICES\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."VSM\_DEVICES\_V" ("VSM\_DEVICE\_NAME", "VSM\_MOD\_DEVICE\_NAME", "VSM\_CLASS", "LOAD\_DATE", "VSM\_GATEWAY\_NAME", "VSM\_IP") AS

select distinct vd."VSM\_DEVICE\_NAME",vd."VSM\_MOD\_DEVICE\_NAME",vd."VSM\_CLASS",vd."LOAD\_DATE",vd."VSM\_GATEWAY\_NAME", vdi.VSM\_IP

from

xng\_reports.vsm\_Devices vd

LEFT OUTER JOIN

xng\_reports.vsm\_devices\_ipaddresses vdi

on vd.VSM\_DEVICE\_NAME = vdi.VSM\_DEVICE\_NAME

;

--------------------------------------------------------

-- DDL for View VVXX\_CLLI\_ISSUES

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."VVXX\_CLLI\_ISSUES" ("CLLI\_6", "AREA", "REGION") AS

WITH unq\_clli AS

(SELECT DISTINCT SUBSTR (vsm\_device\_name, 1, 6) clli\_6, area, region

FROM vzw\_vsm\_xng\_xref vvxx),

multi\_clli\_region AS

(SELECT clli\_6, COUNT (1)

FROM unq\_clli c

GROUP BY clli\_6

HAVING COUNT (1) > 1)

SELECT m.clli\_6, area, region

FROM multi\_clli\_region m JOIN unq\_clli c ON m.clli\_6 = c.clli\_6

ORDER BY m.clli\_6

;

--------------------------------------------------------

-- DDL for View VZW\_DOMAIN\_LEAVES

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."VZW\_DOMAIN\_LEAVES" ("DOMAIN\_INST\_ID") AS

select DOMAIN\_INST\_ID

from

(

-- all

select DOMAIN\_INST\_ID as DOMAIN\_INST\_ID

from vzwnet.domain\_inst

MINUS

-- minus the parents

select PARENT\_DOMAIN\_INST\_ID as DOMAIN\_INST\_ID

from vzwnet.domain\_inst

)

where DOMAIN\_INST\_ID not in

(

1000 -- VZWNET\_DOMAIN

,1011 -- LEC\_DOMAIN

,1013 -- OTN\_ASP\_DOMAIN

,1014 -- IAE\_DOMAIN

,1015 -- CDS\_DOMIAN

,1016 -- NSS\_DOMAIN

,1017 -- VAP\_DOMIAN

)

union

-- don't forget IAN\_DOMAIN

select DOMAIN\_INST\_ID as DOMAIN\_INST\_ID

from vzwnet.domain\_inst

where domain\_name = 'IAN\_DOMAIN'

;

--------------------------------------------------------

-- DDL for View VZW\_VSM\_XNG\_XREF

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."VZW\_VSM\_XNG\_XREF" ("AREA", "REGION", "VSM\_DEVICE\_NAME", "VSM\_CLASS", "ALLTEL\_DO\_NOT\_REMOVE", "AS\_OF\_DATE", "LEAF\_DOMAIN\_NAME", "LEAF\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_INST\_ID", "CITY", "COUNTY", "DESCR", "DOMAIN\_INST\_ID", "DOMAIN\_NAME", "EQUIP\_INST\_ID", "MARKET", "MARKET\_INST\_ID", "OLD\_SITE\_INST\_ID", "SITE\_INST\_ID", "SITE\_HUM\_ID", "STATE", "STATUS", "VENDOR", "VSM\_CITY", "VSM\_STATE", "VSM\_MOD\_DEVICE\_NAME", "XNG\_DEVICE\_NAME", "XNG\_MOD\_DEVICE\_NAME", "SCHEMA") AS

SELECT dlr.area, dlr.region, vddm.vsm\_device\_name, vddm.vsm\_class,

vddm.alltel\_do\_not\_remove, vddm.as\_of\_date, dlr.domain\_name,

leaf\_domain\_inst\_id, parent\_domain\_inst\_id, null , null,null , null,null , null,null , null,null , null,null , null,null , null,null , null,null , null,null , null

FROM domains\_leaf\_reporting dlr JOIN vsm\_devices\_domain\_map vddm

ON dlr.domain\_inst\_id = vddm.leaf\_domain\_inst\_id

;

--------------------------------------------------------

-- DDL for View VZW\_VSM\_XNG\_XREF\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."VZW\_VSM\_XNG\_XREF\_V" ("AREA", "REGION", "VSM\_DEVICE\_NAME", "VSM\_CLASS", "ALLTEL\_DO\_NOT\_REMOVE", "AS\_OF\_DATE", "LEAF\_DOMAIN\_NAME", "LEAF\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_INST\_ID") AS

SELECT dlr.area, dlr.region, vddm.vsm\_device\_name, vddm.vsm\_class,

vddm.alltel\_do\_not\_remove,

vddm.as\_of\_date, dlr.domain\_name, leaf\_domain\_inst\_id, parent\_domain\_inst\_id

FROM domains\_leaf\_reporting dlr

JOIN vsm\_devices\_domain\_map vddm

ON dlr.domain\_inst\_id = vddm.leaf\_domain\_inst\_id

;

--------------------------------------------------------

-- DDL for View VZW\_VSM\_XNG\_XREF\_V\_BK

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."VZW\_VSM\_XNG\_XREF\_V\_BK" ("AREA", "REGION", "VSM\_DEVICE\_NAME", "VSM\_CLASS", "ALLTEL\_DO\_NOT\_REMOVE", "AS\_OF\_DATE", "LEAF\_DOMAIN\_NAME", "LEAF\_DOMAIN\_INST\_ID", "PARENT\_DOMAIN\_INST\_ID") AS

SELECT dlr.area, dlr.region, vddm.vsm\_device\_name, vddm.vsm\_class,

vddm.alltel\_do\_not\_remove, vddm.as\_of\_date, dlr.domain\_name,

leaf\_domain\_inst\_id, parent\_domain\_inst\_id

FROM domains\_leaf\_reporting dlr JOIN vsm\_devices\_domain\_map vddm

ON dlr.domain\_inst\_id = vddm.leaf\_domain\_inst\_id

;

--------------------------------------------------------

-- DDL for View WATCHDOG\_V

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."WATCHDOG\_V" ("PADDED\_PROCESS", "PADDED\_DS\_PROCESS\_NAME", "PROCESS\_NAME", "PROCESS\_STATUS", "DS\_PROCESS\_NAME", "DS\_PROCESS\_STATUS", "PROCESS\_ID", "START\_TIME", "END\_TIME", "DS\_PROCESS\_ID", "DS\_IS\_LEAF", "LEVEL\_IN\_HIERARCHY", "REL\_ORDER") AS

WITH topmost\_upstream\_process AS

(SELECT p.process\_id

FROM xng\_reports.process\_dependency\_matrix p

MINUS

SELECT ds\_process\_id

FROM xng\_reports.process\_dependency\_matrix),

proc AS

(SELECT m.process\_id, pp.process\_name process\_name,

pp.exec\_status process\_status, m.ds\_process\_id,

cp.process\_name ds\_process\_name,

cp.exec\_status ds\_process\_status,

TO\_CHAR (pp.start\_time, 'MM/DD/YYYY HH24:MI:SS') start\_time,

TO\_CHAR (pp.end\_time, 'MM/DD/YYYY HH24:MI:SS') end\_time

FROM xng\_reports.process\_dependency\_matrix m LEFT OUTER JOIN xng\_reports.all\_processes pp

ON pp.process\_id = m.process\_id

LEFT OUTER JOIN xng\_reports.all\_processes cp

ON cp.process\_id = m.ds\_process\_id

)

SELECT LPAD (' ', (LEVEL - 1) \* 3) || m.process\_name padded\_process,

LPAD (' ', LEVEL \* 3)

|| m.ds\_process\_name padded\_ds\_process\_name,

m.process\_name, m.process\_status, m.ds\_process\_name,

m.ds\_process\_status, m.process\_id, m.start\_time, m.end\_time,

m.ds\_process\_id, CONNECT\_BY\_ISLEAF ds\_is\_leaf,

LEVEL level\_in\_hierarchy, ROWNUM rel\_order

FROM proc m

start with m.process\_id in (select process\_id from topmost\_upstream\_process)

connect by m.process\_id = prior m.ds\_process\_id

;

--------------------------------------------------------

-- DDL for View XNGSTD\_MARKET\_DOMAIN\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XNGSTD\_MARKET\_DOMAIN\_VW" ("AREA", "REGION", "MARKET\_DOMAIN\_NAME") AS

SELECT DISTINCT AREA, REGION, MARKET\_DOMAIN\_NAME

FROM XNG\_REPORTS.VZW\_NETWORK\_ORG

WHERE MARKET\_DOMAIN\_NAME IN

(SELECT DOMAIN\_NAME FROM XNG\_REPORTS.XNGSTD\_DOMAIN\_REGION)

;

--------------------------------------------------------

-- DDL for View XNGSTD\_REGION\_DOMAIN\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XNGSTD\_REGION\_DOMAIN\_VW" ("AREA", "REGION", "REGION\_DOMAIN\_NAME") AS

SELECT DISTINCT AREA, REGION, REGION\_DOMAIN\_NAME

FROM XNG\_REPORTS.VZW\_NETWORK\_ORG

;

--------------------------------------------------------

-- DDL for View XNGSTD\_SUM\_EQUIP\_ALL\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XNGSTD\_SUM\_EQUIP\_ALL\_VW" ("EQUIP\_INST\_ID", "CATEGORY", "DESCR", "DOMAIN\_NAME", "DOMAIN\_INST\_ID", "VENDOR", "MODEL", "SITE\_INST\_ID", "CLLI", "CLEI", "EQ\_CLASS", "STATUS\_NAME", "SITE\_HUM\_ID", "SITE\_STATE") AS

WITH LEAF\_DOMAINS\_MINUS\_NNO

AS (SELECT pdi.\*

FROM vzwnet.domain\_inst pdi, vzwnet.domain\_inst cdi

WHERE pdi.DOMAIN\_INST\_ID = cdi.PARENT\_DOMAIN\_INST\_ID(+)

AND cdi.DOMAIN\_INST\_ID IS NULL

AND pdi.DOMAIN\_NAME NOT IN

('NNO\_DOMAIN', 'SLTX\_LAB\_DOMAIN', 'ORPHAN\_DOMAIN')),

active\_status

AS (SELECT status\_name

FROM vzwnet.val\_deploy\_status

WHERE root\_status = 'A' AND appl\_class\_mask LIKE '%Q%'),

all\_equip\_in\_NNO\_w\_prefix

AS (SELECT DISTINCT ei.equip\_inst\_id

FROM vzwnet.equip\_DOMAIN\_MAP edm,

vzwnet.equip\_INST ei,

vzwnet.domain\_inst di,

active\_status status

WHERE ei.equip\_inst\_id = edm.equip\_inst\_id

AND di.DOMAIN\_INST\_ID = edm.DOMAIN\_INST\_ID

AND di.DOMAIN\_NAME = 'NNO\_DOMAIN'

AND ei.descr LIKE 'NNO %'

AND status.status\_name = ei.status),

all\_equip\_in\_NNO\_w\_no\_prefix

AS (SELECT DISTINCT ei.equip\_inst\_id

FROM vzwnet.equip\_DOMAIN\_MAP edm,

vzwnet.equip\_INST ei,

vzwnet.domain\_inst di,

active\_status status

WHERE ei.equip\_inst\_id = edm.equip\_inst\_id

AND di.DOMAIN\_INST\_ID = edm.DOMAIN\_INST\_ID

AND di.DOMAIN\_NAME = 'NNO\_DOMAIN'

AND ei.descr NOT LIKE 'NNO %'

AND status.status\_name = ei.status),

not\_exists\_in\_other\_domains

AS (SELECT DISTINCT equip\_inst\_id FROM all\_equip\_in\_NNO\_w\_no\_prefix

MINUS

SELECT DISTINCT edm.equip\_inst\_id

FROM vzwnet.equip\_DOMAIN\_MAP edm, LEAF\_DOMAINS\_MINUS\_NNO di

WHERE edm.domain\_inst\_id = di.DOMAIN\_INST\_ID),

ALL\_EQUIPS

AS (SELECT DISTINCT ei.equip\_inst\_id

FROM vzwnet.equip\_inst ei, active\_status status

WHERE status.status\_name = ei.status),

all\_nno\_equips

AS (SELECT DISTINCT equip\_inst\_id FROM all\_equip\_in\_NNO\_w\_prefix

UNION

SELECT DISTINCT equip\_inst\_id FROM not\_exists\_in\_other\_domains),

OTHER\_DOMAIN\_EQUIPS

AS (SELECT equip\_inst\_id FROM ALL\_EQUIPS

MINUS

SELECT equip\_inst\_id FROM all\_nno\_equips),

ALL\_EQUIPS\_PD

AS (SELECT n.\*, di.domain\_name, di.domain\_inst\_id

FROM all\_nno\_equips n, vzwnet.domain\_inst di

WHERE di.domain\_name = 'NNO\_DOMAIN'

UNION

SELECT n.\*, di.domain\_name, di.domain\_inst\_id

FROM OTHER\_DOMAIN\_EQUIPS n,

LEAF\_DOMAINS\_MINUS\_NNO di,

vzwnet.equip\_domain\_map edm

WHERE n.equip\_inst\_id = edm.equip\_inst\_id

AND edm.domain\_inst\_id = di.domain\_inst\_id),

ALL\_EQUIPS\_WITH\_DETAILS

AS (SELECT ei.equip\_inst\_id,

ei.TYPE,

ei.descr,

a.domain\_name,

a.domain\_inst\_id,

ei.vendor,

ei.model,

ei.site\_inst\_id,

ei.clli,

ei.clei,

ei.eq\_class,

stat.status\_name,

si.SITE\_HUM\_ID,

si.state\_prov site\_state

FROM active\_status stat,

vzwnet.site\_inst si,

vzwnet.equip\_inst ei,

ALL\_EQUIPS\_PD a

WHERE ei.equip\_inst\_id = a.equip\_inst\_id

AND si.site\_inst\_id = ei.site\_inst\_id

AND ei.status = stat.status\_name)

SELECT ei."EQUIP\_INST\_ID",

ei."TYPE" CATEGORY,

ei."DESCR",

ei."DOMAIN\_NAME",

ei."DOMAIN\_INST\_ID",

ei."VENDOR",

ei."MODEL",

ei."SITE\_INST\_ID",

ei."CLLI",

ei."CLEI",

ei."EQ\_CLASS",

ei."STATUS\_NAME",

ei."SITE\_HUM\_ID",

ei."SITE\_STATE"

FROM ALL\_EQUIPS\_WITH\_DETAILS ei

;

--------------------------------------------------------

-- DDL for View XNGSTD\_SUM\_PATH\_ALL\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XNGSTD\_SUM\_PATH\_ALL\_VW" ("CIRC\_PATH\_HUM\_ID", "CIRC\_PATH\_REV\_NBR", "TYPE\_NAME", "DOMAIN\_NAME", "DOMAIN\_INST\_ID", "BANDWIDTH", "STATUS", "A\_SIDE\_SITE\_ID", "Z\_SIDE\_SITE\_ID", "NBR\_CHANNELS", "NBR\_CHAN\_ASSIGNED", "TOPOLOGY", "IS\_TRUNK\_GROUP", "CIRC\_PATH\_INST\_ID", "A\_SIDE\_SITE\_STATE", "DOM\_IP") AS

WITH LEAF\_DOMAINS\_MINUS\_NNO

AS (SELECT pdi.\*

FROM vzwnet.domain\_inst pdi, vzwnet.domain\_inst cdi

WHERE pdi.DOMAIN\_INST\_ID = cdi.PARENT\_DOMAIN\_INST\_ID(+)

AND cdi.DOMAIN\_INST\_ID IS NULL

AND pdi.DOMAIN\_NAME NOT IN

('NNO\_DOMAIN', 'SLTX\_LAB\_DOMAIN', 'ORPHAN\_DOMAIN')),

active\_status

AS (SELECT status\_name

FROM vzwnet.val\_deploy\_status

WHERE root\_status = 'A' AND appl\_class\_mask LIKE '%P%'),

all\_path\_in\_NNO\_w\_prefix

AS (SELECT DISTINCT si.circ\_path\_inst\_id

FROM vzwnet.path\_DOMAIN\_MAP pdm,

vzwnet.circ\_path\_INST si,

vzwnet.domain\_inst di,

active\_status status

WHERE si.circ\_path\_inst\_id = pdm.circ\_path\_inst\_id

AND di.DOMAIN\_INST\_ID = pdm.DOMAIN\_INST\_ID

AND di.DOMAIN\_NAME = 'NNO\_DOMAIN'

AND si.circ\_path\_hum\_id LIKE 'NNO %'

AND status.status\_name = si.status),

all\_path\_in\_NNO\_w\_no\_prefix

AS (SELECT DISTINCT si.circ\_path\_inst\_id

FROM vzwnet.path\_DOMAIN\_MAP pdm,

vzwnet.circ\_path\_INST si,

vzwnet.domain\_inst di,

active\_status status

WHERE si.circ\_path\_inst\_id = pdm.circ\_path\_inst\_id

AND di.DOMAIN\_INST\_ID = pdm.DOMAIN\_INST\_ID

AND di.DOMAIN\_NAME = 'NNO\_DOMAIN'

AND si.circ\_path\_hum\_id NOT LIKE 'NNO %'

AND status.status\_name = si.status),

not\_exists\_in\_other\_domains

AS (SELECT DISTINCT circ\_path\_inst\_id

FROM all\_path\_in\_NNO\_w\_no\_prefix

MINUS

SELECT DISTINCT pdm.circ\_path\_inst\_id

FROM vzwnet.path\_DOMAIN\_MAP pdm, LEAF\_DOMAINS\_MINUS\_NNO di

WHERE pdm.domain\_inst\_id = di.DOMAIN\_INST\_ID),

ALL\_CIRC\_PATHS

AS (SELECT DISTINCT si.circ\_path\_inst\_id

FROM vzwnet.circ\_path\_inst si, active\_status status

WHERE status.status\_name = si.status),

all\_nno\_circ\_paths

AS (SELECT DISTINCT circ\_path\_inst\_id FROM all\_path\_in\_NNO\_w\_prefix

UNION

SELECT DISTINCT circ\_path\_inst\_id

FROM not\_exists\_in\_other\_domains),

OTHER\_DOMAIN\_PATHS

AS (SELECT circ\_path\_inst\_id FROM ALL\_CIRC\_PATHS

MINUS

SELECT circ\_path\_inst\_id FROM all\_nno\_circ\_paths),

ALL\_CIRC\_PATHS\_PD

AS (SELECT n.\*, di.domain\_name, di.domain\_inst\_id

FROM all\_nno\_circ\_paths n, vzwnet.domain\_inst di

WHERE di.domain\_name = 'NNO\_DOMAIN'

UNION

SELECT n.\*, di.domain\_name, di.domain\_inst\_id

FROM OTHER\_DOMAIN\_PATHS n,

LEAF\_DOMAINS\_MINUS\_NNO di,

vzwnet.path\_DOMAIN\_MAP pdm

WHERE n.circ\_path\_inst\_id = pdm.circ\_path\_inst\_id

AND pdm.domain\_inst\_id = di.domain\_inst\_id),

ALL\_PATHS\_WITH\_DETAILS

AS (SELECT cpi.circ\_path\_hum\_id,

cpi.circ\_path\_rev\_nbr,

ct.type\_name,

a.domain\_name,

a.domain\_inst\_id,

cpi.bandwidth,

cpi.status,

cpi.a\_side\_site\_id,

cpi.z\_side\_site\_id,

cpi.nbr\_channels,

cpi.nbr\_chan\_assigned,

cpi.topology,

ct.is\_trunk\_group,

cpi.circ\_path\_inst\_id,

si.state\_prov a\_side\_site\_state

FROM active\_status stat,

vzwnet.circ\_types ct,

vzwnet.site\_inst si,

vzwnet.circ\_path\_inst cpi,

ALL\_CIRC\_PATHS\_PD a

WHERE cpi.circ\_path\_inst\_id = a.circ\_path\_inst\_id

AND si.site\_inst\_id = cpi.a\_side\_site\_id

AND cpi.TYPE = ct.type\_name

AND cpi.status = stat.status\_name),

dom\_node\_ip

AS (SELECT DISTINCT

n.circ\_path\_inst\_id,

NVL (n.xng\_dom\_ip, 'not present') dom\_ip

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit n,

ALL\_PATHS\_WITH\_DETAILS p

WHERE n.circ\_path\_inst\_id = p.circ\_path\_inst\_id

AND n.circ\_type IN ('EVDO', '3G'))

SELECT p."CIRC\_PATH\_HUM\_ID",p."CIRC\_PATH\_REV\_NBR",p."TYPE\_NAME",p."DOMAIN\_NAME",p."DOMAIN\_INST\_ID",p."BANDWIDTH",p."STATUS",p."A\_SIDE\_SITE\_ID",p."Z\_SIDE\_SITE\_ID",p."NBR\_CHANNELS",p."NBR\_CHAN\_ASSIGNED",p."TOPOLOGY",p."IS\_TRUNK\_GROUP",p."CIRC\_PATH\_INST\_ID",p."A\_SIDE\_SITE\_STATE", dn.dom\_ip

FROM ALL\_PATHS\_WITH\_DETAILS p, dom\_node\_ip dn

WHERE p.circ\_path\_inst\_id = dn.circ\_path\_inst\_id(+)

;

--------------------------------------------------------

-- DDL for View XNGSTD\_SUM\_SGMT\_ALL\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XNGSTD\_SUM\_SGMT\_ALL\_VW" ("CIRC\_INST\_ID", "CIRC\_HUM\_ID", "DOMAIN\_NAME", "XNG\_TYPE", "VENDOR", "XNG\_STATUS", "XNG\_BANDWIDTH", "CHARGE\_AMOUNT", "DISCONNECT\_PON", "DISCONNECT\_ORDER\_DATE", "DISCONNECT\_ORDER\_NUMBER", "DISCONNECT\_DATE", "BILL\_DATE\_MONTH", "BILL\_DATE\_YEAR", "ASIDESITESTATE") AS

WITH domain

AS ( SELECT DISTINCT area, region, region\_domain\_name

FROM xng\_reports.vzw\_network\_org

ORDER BY area, region),

all\_seg

AS (SELECT DISTINCT drf.circ\_inst\_id,

drf.circ\_hum\_id,

d.region\_domain\_name domain\_name,

bxa.xng\_type,

drf.vendor,

bxa.xng\_status,

bxa.xng\_bandwidth,

drf.charge\_amount,

drf.disconnect\_pon,

drf.disconnect\_order\_date,

drf.disconnect\_order\_number,

drf.disconnect\_date,

bxa.bill\_date\_month,

bxa.bill\_date\_year,

si.state\_prov aSideSiteState

FROM xng\_reports.decomm\_reqd\_fields drf,

xng\_reports.btp\_xng\_audit bxa,

vzwnet.site\_inst si,

vzwnet.circ\_inst ci,

domain d

WHERE bxa.xng\_status = 'Decommissioned'

AND ci.circ\_inst\_id = drf.circ\_inst\_id

AND si.site\_inst\_id = ci.a\_site\_id

AND bxa.xng\_circ\_inst\_id = drf.circ\_inst\_id

AND d.area = drf.area

AND d.region = drf.region)

SELECT seg."CIRC\_INST\_ID",seg."CIRC\_HUM\_ID",seg."DOMAIN\_NAME",seg."XNG\_TYPE",seg."VENDOR",seg."XNG\_STATUS",seg."XNG\_BANDWIDTH",seg."CHARGE\_AMOUNT",seg."DISCONNECT\_PON",seg."DISCONNECT\_ORDER\_DATE",seg."DISCONNECT\_ORDER\_NUMBER",seg."DISCONNECT\_DATE",seg."BILL\_DATE\_MONTH",seg."BILL\_DATE\_YEAR",seg."ASIDESITESTATE"

FROM all\_seg seg

;

--------------------------------------------------------

-- DDL for View XNGSTD\_SUM\_SITE\_ALL\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XNGSTD\_SUM\_SITE\_ALL\_VW" ("SITE\_INST\_ID", "TYPE", "NAME", "DOMAINNAME", "DOMAIN\_INST\_ID", "CLLI", "LATITUDE", "LONGITUDE", "STREET", "CITY", "STATE", "POSTALCODE", "POSTALCODE2", "FLOOR", "NPANXX", "COUNTY", "COUNTRY", "RESTRICTIONS", "DIRECTIONS", "STATUS", "FIRE\_NUMBER", "HVAC\_COMPANY", "HVAC\_PHONE\_NUMBER", "LOCAL\_POLICE\_NUMBER", "POWER\_ACCOUNT\_NUMBER", "POWER\_COMPANY", "POWER\_PHONE\_NUMBER") AS

WITH LEAF\_DOMAINS\_MINUS\_NNO

AS (SELECT pdi.\*

FROM vzwnet.domain\_inst pdi, vzwnet.domain\_inst cdi

WHERE pdi.DOMAIN\_INST\_ID = cdi.PARENT\_DOMAIN\_INST\_ID(+)

AND cdi.DOMAIN\_INST\_ID IS NULL

AND pdi.DOMAIN\_NAME NOT IN

('NNO\_DOMAIN', 'SLTX\_LAB\_DOMAIN', 'ORPHAN\_DOMAIN')),

active\_status

AS (SELECT status\_name

FROM vzwnet.val\_deploy\_status

WHERE root\_status = 'A' AND appl\_class\_mask LIKE '%I%'),

all\_site\_in\_NNO\_w\_prefix

AS (SELECT DISTINCT si.site\_inst\_id

FROM vzwnet.SITE\_DOMAIN\_MAP sdm,

vzwnet.SITE\_INST si,

vzwnet.domain\_inst di,

active\_status status

WHERE si.site\_inst\_id = sdm.site\_inst\_id

AND di.DOMAIN\_INST\_ID = sdm.DOMAIN\_INST\_ID

AND di.DOMAIN\_NAME = 'NNO\_DOMAIN'

AND si.site\_hum\_id LIKE 'NNO %'

AND status.status\_name = si.status),

all\_site\_in\_NNO\_w\_no\_prefix

AS (SELECT DISTINCT si.site\_inst\_id

FROM vzwnet.SITE\_DOMAIN\_MAP sdm,

vzwnet.SITE\_INST si,

vzwnet.domain\_inst di,

active\_status status

WHERE si.site\_inst\_id = sdm.site\_inst\_id

AND di.DOMAIN\_INST\_ID = sdm.DOMAIN\_INST\_ID

AND di.DOMAIN\_NAME = 'NNO\_DOMAIN'

AND si.site\_hum\_id NOT LIKE 'NNO %'

AND status.status\_name = si.status),

not\_exists\_in\_other\_domains

AS (SELECT DISTINCT site\_inst\_id FROM all\_site\_in\_NNO\_w\_no\_prefix

MINUS

SELECT DISTINCT sdm.site\_inst\_id

FROM vzwnet.SITE\_DOMAIN\_MAP sdm, LEAF\_DOMAINS\_MINUS\_NNO di

WHERE sdm.domain\_inst\_id = di.DOMAIN\_INST\_ID),

ALL\_SITES

AS (SELECT DISTINCT si.site\_inst\_id

FROM vzwnet.site\_inst si, active\_status status

WHERE status.status\_name = si.status),

all\_nno\_sites

AS (SELECT DISTINCT site\_inst\_id FROM all\_site\_in\_NNO\_w\_prefix

UNION

SELECT DISTINCT site\_inst\_id FROM not\_exists\_in\_other\_domains),

OTHER\_DOMAIN\_SITES

AS (SELECT site\_inst\_id FROM ALL\_SITES

MINUS

SELECT site\_inst\_id FROM all\_nno\_sites),

ALL\_SITES\_PD

AS (SELECT n.\*, di.domain\_name, di.domain\_inst\_id

FROM all\_nno\_sites n, vzwnet.domain\_inst di

WHERE di.domain\_name = 'NNO\_DOMAIN'

UNION

SELECT n.\*, di.domain\_name, di.domain\_inst\_id

FROM OTHER\_DOMAIN\_SITES n,

LEAF\_DOMAINS\_MINUS\_NNO di,

vzwnet.site\_domain\_map sdm

WHERE n.site\_inst\_id = sdm.site\_inst\_id

AND sdm.domain\_inst\_id = di.domain\_inst\_id),

ALL\_SITES\_WITH\_DETAILS

AS (SELECT si.site\_inst\_id,

si.num TYPE,

si.site\_hum\_id NAME,

a.domain\_name domainname,

a.domain\_inst\_id,

si.clli,

si.latitude,

si.longitude,

si.address street,

si.city,

si.state\_prov state,

si.post\_code\_1 postalcode,

si.post\_code\_2 postalcode2,

si.FLOOR,

si.npa\_nxx npanxx,

si.county,

si.country,

si.restrictions,

si.contacts directions,

status

FROM active\_status stat, vzwnet.site\_inst si, ALL\_SITES\_PD a

WHERE si.site\_inst\_id = a.site\_inst\_id

AND si.status = stat.status\_name),

hvac\_company

AS (SELECT DISTINCT van.val\_attr\_inst\_id,

van.attr\_name,

attr\_value hvac\_company,

si.site\_inst\_id

FROM vzwnet.val\_attr\_name van,

vzwnet.site\_attr\_settings sas,

ALL\_SITES\_WITH\_DETAILS si

WHERE van.group\_name = 'Emergency Contacts'

AND van.attr\_name = 'HVAC Company'

AND si.site\_inst\_id = sas.site\_inst\_id

AND van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id),

hvac\_phone\_number

AS (SELECT DISTINCT van.val\_attr\_inst\_id,

van.attr\_name,

attr\_value hvac\_phone\_number,

si.site\_inst\_id

FROM vzwnet.val\_attr\_name van,

vzwnet.site\_attr\_settings sas,

ALL\_SITES\_WITH\_DETAILS si

WHERE van.group\_name = 'Emergency Contacts'

AND van.attr\_name = 'HVAC Phone Number'

AND si.site\_inst\_id = sas.site\_inst\_id

AND van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id),

local\_police\_number

AS (SELECT DISTINCT van.val\_attr\_inst\_id,

van.attr\_name,

attr\_value local\_police\_number,

si.site\_inst\_id

FROM vzwnet.val\_attr\_name van,

vzwnet.site\_attr\_settings sas,

ALL\_SITES\_WITH\_DETAILS si

WHERE van.group\_name = 'Emergency Contacts'

AND van.attr\_name = 'Local Police Number'

AND si.site\_inst\_id = sas.site\_inst\_id

AND van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id),

power\_account\_number

AS (SELECT DISTINCT van.val\_attr\_inst\_id,

van.attr\_name,

attr\_value power\_account\_number,

si.site\_inst\_id

FROM vzwnet.val\_attr\_name van,

vzwnet.site\_attr\_settings sas,

ALL\_SITES\_WITH\_DETAILS si

WHERE van.group\_name = 'Emergency Contacts'

AND van.attr\_name = 'Power Account Number'

AND si.site\_inst\_id = sas.site\_inst\_id

AND van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id),

power\_company

AS (SELECT DISTINCT van.val\_attr\_inst\_id,

van.attr\_name,

attr\_value power\_company,

si.site\_inst\_id

FROM vzwnet.val\_attr\_name van,

vzwnet.site\_attr\_settings sas,

ALL\_SITES\_WITH\_DETAILS si

WHERE van.group\_name = 'Emergency Contacts'

AND van.attr\_name = 'Power Company'

AND si.site\_inst\_id = sas.site\_inst\_id

AND van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id),

power\_phone\_number

AS (SELECT DISTINCT van.val\_attr\_inst\_id,

van.attr\_name,

attr\_value power\_phone\_number,

si.site\_inst\_id

FROM vzwnet.val\_attr\_name van,

vzwnet.site\_attr\_settings sas,

ALL\_SITES\_WITH\_DETAILS si

WHERE van.group\_name = 'Emergency Contacts'

AND van.attr\_name = 'Power Phone Number'

AND si.site\_inst\_id = sas.site\_inst\_id

AND van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id),

fire\_number

AS (SELECT DISTINCT van.val\_attr\_inst\_id,

van.attr\_name,

attr\_value fire\_number,

si.site\_inst\_id

FROM vzwnet.val\_attr\_name van,

vzwnet.site\_attr\_settings sas,

ALL\_SITES\_WITH\_DETAILS si

WHERE van.group\_name = 'Emergency Contacts'

AND van.attr\_name = 'Local Fire Number'

AND sas.site\_inst\_id = si.site\_inst\_id

AND van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id)

SELECT si."SITE\_INST\_ID",si."TYPE",si."NAME",si."DOMAINNAME",si."DOMAIN\_INST\_ID",si."CLLI",si."LATITUDE",si."LONGITUDE",si."STREET",si."CITY",si."STATE",si."POSTALCODE",si."POSTALCODE2",si."FLOOR",si."NPANXX",si."COUNTY",si."COUNTRY",si."RESTRICTIONS",si."DIRECTIONS",si."STATUS",

fn.fire\_number,

hc.hvac\_company,

hpn.hvac\_phone\_number,

lpn.local\_police\_number,

pan.power\_account\_number,

pc.power\_company,

ppn.power\_phone\_number

FROM ALL\_SITES\_WITH\_DETAILS si,

fire\_number fn,

hvac\_company hc,

hvac\_phone\_number hpn,

local\_police\_number lpn,

power\_account\_number pan,

power\_company pc,

power\_phone\_number ppn

WHERE si.site\_inst\_id = fn.site\_inst\_id(+)

AND si.site\_inst\_id = hc.site\_inst\_id(+)

AND si.site\_inst\_id = hpn.site\_inst\_id(+)

AND si.site\_inst\_id = lpn.site\_inst\_id(+)

AND si.site\_inst\_id = pan.site\_inst\_id(+)

AND si.site\_inst\_id = pc.site\_inst\_id(+)

AND si.site\_inst\_id = ppn.site\_inst\_id(+)

;

--------------------------------------------------------

-- DDL for View XNGSTD\_SWITCH\_VENDOR\_VW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XNGSTD\_SWITCH\_VENDOR\_VW" ("SWITCH\_ID", "VENDOR") AS

SELECT DISTINCT

SUBSTR (bm.VSM\_DEVICE\_NAME\_MTX, 1, 6)

|| SUBSTR (bm.VSM\_DEVICE\_NAME\_MTX, 11, 2)

switch\_id,

'NORTEL' vendor

FROM xng\_reports.bsm\_mtx\_map bm

WHERE bm.VSM\_DEVICE\_NAME\_BSM LIKE '%\_BSM'

AND SUBSTR (bm.VSM\_DEVICE\_NAME\_MTX, 1, 6)

|| SUBSTR (bm.VSM\_DEVICE\_NAME\_MTX, 11, 2) NOT IN

('CHTNSC01', 'RLGHNC01')

UNION

SELECT DISTINCT

SUBSTR (bm.VSM\_DEVICE\_NAME\_MTX, 1, 6)

|| SUBSTR (bm.VSM\_DEVICE\_NAME\_MTX, 11, 2)

switch\_id,

'MOTOROLA' vendor

FROM xng\_reports.bsm\_mtx\_map bm

WHERE bm.VSM\_DEVICE\_NAME\_BSM LIKE '%\_OMCR%'

UNION

SELECT DISTINCT

SUBSTR (bm.VSM\_DEVICE\_NAME\_MTX, 1, 6) switch\_id, 'MOTOROLA' vendor

FROM xng\_reports.bsm\_mtx\_map bm

WHERE bm.VSM\_DEVICE\_NAME\_BSM LIKE '%\_OMCR%'

UNION

SELECT switch\_id, 'LUCENT' vendor FROM xng\_reports.ECP\_SWITCH\_MAP

;

--------------------------------------------------------

-- DDL for View XOR\_MENUS\_VIEW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XOR\_MENUS\_VIEW" ("MENU\_ID", "MENU\_NAME", "MENU\_REL\_ORDER", "PARENT\_MENU\_ID", "PARENT\_MENU\_NAME", "ROOT\_MENU\_ID", "ROOT\_MENU\_NAME", "MENU\_PATH") AS

SELECT DISTINCT

M.MENU\_ID,

M.MENU\_NAME,

M.REL\_ORDER MENU\_REL\_ORDER,

M2.MENU\_ID AS PARENT\_MENU\_ID,

M2.MENU\_NAME AS PARENT\_MENU\_NAME,

TRIM(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH (DECODE (LEVEL,1, M.MENU\_ID, ''), '/')) ROOT\_MENU\_ID,

TRIM(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH (DECODE (LEVEL,1, M.MENU\_NAME, ''), '/')) ROOT\_MENU\_NAME,

TRIM(LEADING '>' FROM SYS\_CONNECT\_BY\_PATH (M.MENU\_NAME, ' > ')) MENU\_PATH

FROM

XOR\_MENUS M,

XOR\_MENUS M2

WHERE

(LEVEL = 1 OR M.PARENT\_MENU\_ID IS NOT NULL)

AND M.PARENT\_MENU\_ID = M2.MENU\_ID(+)

START WITH

M.PARENT\_MENU\_ID IS NULL

CONNECT BY

PRIOR M.MENU\_ID=M.PARENT\_MENU\_ID

;

--------------------------------------------------------

-- DDL for View XOR\_REPORTS\_VIEW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XOR\_REPORTS\_VIEW" ("TAB\_ID", "MENU\_ID", "MENU\_NAME", "MENU\_REL\_ORDER", "REPORT\_ID", "REPORT\_NAME", "REPORT\_REL\_ORDER", "OPEN\_OPTION", "PROJECT\_NAME", "HOSTED\_APPLICATION", "HOSTED\_SERVER", "RELATIVE\_URL", "URL", "CAPTION", "NAVIGATION\_PATH") AS

SELECT DISTINCT tm.tab\_id, m.menu\_id, m.menu\_name, m.menu\_rel\_order,

r.report\_id, r.report\_name,

r.rel\_order AS report\_rel\_order, r.open\_option,

r.project\_name, r.hosted\_application, r.hosted\_server,

r.relative\_url,

DECODE (r.permanent\_rewrite\_name,

NULL, 'http://'

|| r.hosted\_server

|| DECODE (r.hosted\_port,

NULL, NULL,

':' || r.hosted\_port

)

|| r.relative\_url

|| DECODE (r.next\_page,

NULL, NULL,

'?p\_report=' || r.report\_name

)

|| DECODE (r.next\_page,

NULL, NULL,

'&' || 'p\_nextpage=' || r.next\_page

)

|| DECODE (r.rep\_level,

NULL, NULL,

'&' || 'p\_replevel=' || r.rep\_level

),

'http://'

|| r.hosted\_server

|| '/'

|| r.permanent\_rewrite\_name

) AS url,

R.DESCRIPTION AS CAPTION,

-- CONCAT

-- (tm.tab\_name,

-- CONCAT (' > ',

CONCAT (M.MENU\_PATH,

CONCAT (' > ', R.REPORT\_NAME)

-- )

-- )

) navigation\_path

FROM xor\_menus\_view m,

xor\_reports r,

xor\_menus\_reports\_link mr,

xor\_tabs\_menus\_view tm

WHERE m.menu\_id = mr.menu\_id

AND mr.report\_id = r.report\_id

AND M.MENU\_ID = TM.MENU\_ID(+)

;

--------------------------------------------------------

-- DDL for View XOR\_TABS\_MENUS\_VIEW

--------------------------------------------------------

CREATE OR REPLACE FORCE VIEW "XNG\_REPORTS"."XOR\_TABS\_MENUS\_VIEW" ("TAB\_ID", "TAB\_NAME", "TAB\_REL\_ORDER", "MENU\_ID", "MENU\_NAME", "MENU\_REL\_ORDER", "PARENT\_MENU\_ID", "PARENT\_MENU\_NAME", "ROOT\_MENU\_ID", "ROOT\_MENU\_NAME") AS

select distinct t.tab\_id, t.tab\_name, t.REL\_ORDER tab\_rel\_order, m.menu\_id, m.menu\_name,

m.menu\_rel\_order, m.parent\_menu\_id, m.parent\_menu\_name, m.ROOT\_MENU\_ID, m.ROOT\_MENU\_NAME

from XOR\_TABS t, XOR\_Menus\_View m,XOR\_TABS\_MENUS\_LINK tm

where t.TAB\_ID = tm.TAB\_ID and tm.MENU\_ID = m.ROOT\_MENU\_ID

;

--------------------------------------------------------

-- DDL for Index NE\_VS\_XNG\_CSR\_1\_IDA

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NE\_VS\_XNG\_CSR\_1\_IDA" ON "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT" ("XNG\_CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NCM\_CSR\_VLAN\_NXA\_XNG\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NCM\_CSR\_VLAN\_NXA\_XNG\_IDX" ON "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDIT\_WK" ("CSR\_VENDOR", "XNG\_VLAN\_NUMBER", "VLAN\_STATUS", "MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NGNA\_MATCH\_CODE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NGNA\_MATCH\_CODE\_IDX" ON "XNG\_REPORTS"."NCM\_GI\_CI\_NGMLS\_AUDIT\_WK" ("MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PK\_SWITCH\_PATH\_TESTABILITY\_WK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."PK\_SWITCH\_PATH\_TESTABILITY\_WK" ON "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" ("SWITCH\_PATH\_T1\_TESTABILITY\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NORTEL\_NE\_VS\_XNG\_AUDIT\_IDX3

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_IDX3" ON "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT" ("VSM\_DEVICE\_NAME\_BSM", "NE\_DCG\_NUMBER", "NE\_SLOT\_NUMBER", "NE\_SPAN\_NUMBER", "VSM\_DEVICE\_NAME\_MTX")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_SLOT\_TID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_SLOT\_TID\_IDX" ON "XNG\_REPORTS"."INC\_MSPP\_SLOT" ("TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_EQPT\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."NETSMART\_EQPT\_PK" ON "XNG\_REPORTS"."NETSMART\_EQPT" ("INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NS\_XNG\_PORTS\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NS\_XNG\_PORTS\_IDX1" ON "XNG\_REPORTS"."NETSMART\_XNG\_PORTS" ("TID", "PORT\_HUM\_ID", "SLOT"||'-'||"PORT\_HUM\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SEG\_MIS\_EST\_COST\_DOMAIN\_INDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SEG\_MIS\_EST\_COST\_DOMAIN\_INDX" ON "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST" ("DOMAIN")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MOTO\_EVDO\_AUDIT\_WRK\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MOTO\_EVDO\_AUDIT\_WRK\_IDX2" ON "XNG\_REPORTS"."MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT\_WRK" ("XNG\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_EBH\_RTR\_NODE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_EBH\_RTR\_NODE\_IDX" ON "XNG\_REPORTS"."SAM\_EBH\_RTR" ("NODE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MOTO\_EVDO\_ETHERNET\_INV\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MOTO\_EVDO\_ETHERNET\_INV\_IDX" ON "XNG\_REPORTS"."MOTO\_EVDO\_ETHERNET\_INV" ("VSM\_DEVICE\_NAME\_AEMS", "IPBSCDO\_NUMBER", "MCCDO\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index LUCENT\_NE\_VS\_XNG\_AUDT\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDT\_IDX" ON "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDIT" ("SYS\_ID", "ECP\_SID", "NE\_CELL\_NUM", "NE\_URC\_NUM", "NE\_DS1\_NUM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_SNCNE\_AUDIT\_WK\_INDEX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT\_WK\_INDEX1" ON "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT\_WK" ("TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_JOB\_STATUS\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_JOB\_STATUS\_PK" ON "XNG\_REPORTS"."XNG\_JOB\_STATUS" ("JOB\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVP\_HUB\_SPOKE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVP\_HUB\_SPOKE\_IDX" ON "XNG\_REPORTS"."XNG\_VLAN\_PARENTS\_WK" ("SPOKE\_TO\_HUB")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_DCOM\_PATH\_DETAIL\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL\_PK" ON "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" ("REPORT\_ID", "ROW\_NUM", "AREA", "REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_CSR\_VLAN\_NXA\_XNG\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_CSR\_VLAN\_NXA\_XNG\_IDX" ON "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT\_WK" ("CSR\_VENDOR", "XNG\_VLAN\_NUMBER", "VLAN\_STATUS", "MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_CIRC\_EQUIP\_INST

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_CIRC\_EQUIP\_INST" ON "XNG\_REPORTS"."XCONN\_CIRC\_PATH\_DETAILS" ("XNG\_EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVM\_PATH\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVM\_PATH\_IDX" ON "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_MAP" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_DATA\_FOR\_UPD\_OLD

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_OLD" ON "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_OLD" ("STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_VLAN\_NXA\_NE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_VLAN\_NXA\_NE\_IDX" ON "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT\_WK" ("NGMLS\_VENDOR", "NE\_HOSTNAME", "NE\_VLAN\_NUMBER", "MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SEGMENTS\_MISSING\_EST\_COST\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST\_PK" ON "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST" ("CIRC\_INST\_ID", "DOMAIN")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EMB\_IPBSC\_BTS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EMB\_IPBSC\_BTS\_IDX" ON "XNG\_REPORTS"."MT\_EMH\_MCCDO\_BTS" ("IPBSCDO\_NUMBER", "BTS\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_OPTICAL\_LINK\_ATID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_OPTICAL\_LINK\_ATID\_IDX" ON "XNG\_REPORTS"."INC\_MSPP\_OPTICAL\_LINK" ("A\_TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_ISSUE\_ID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_ISSUE\_ID\_IDX" ON "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE" ("CSR\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BSM\_DCG\_DS1\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BSM\_DCG\_DS1\_IDX" ON "XNG\_REPORTS"."BSM\_DCG\_SPAN\_VOICE\_MAP" ("VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER", "SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_VLAN\_NGMLS\_NE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_VLAN\_NGMLS\_NE\_IDX" ON "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT\_WK" ("NGMLS\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVM\_PATH\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVM\_PATH\_IDX2" ON "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_MAP\_WK" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EDATE\_BSM\_ETHERNET\_INV\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EDATE\_BSM\_ETHERNET\_INV\_IDX" ON "XNG\_REPORTS"."BSM\_ETHERNET\_INV" ("EXTRACT\_DATE", "VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_CSR\_VLAN\_NXA\_NE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_CSR\_VLAN\_NXA\_NE\_IDX" ON "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT\_WK" ("CSR\_VENDOR", "NE\_HOSTNAME", "NE\_VLAN\_NUMBER", "MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OLD\_STATS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."OLD\_STATS" ON "XNG\_REPORTS"."OLD\_STATS" ("STATID", "TYPE", "C5", "C1", "C2", "C3", "C4", "VERSION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_CEBH\_CLLI\_EQUIP

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_CEBH\_CLLI\_EQUIP" ON "XNG\_REPORTS"."COMP\_EBH\_DETAILS\_BY\_CLLI" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_VLAN\_PARSED\_WK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XNG\_VLAN\_PARSED\_WK\_IDX" ON "XNG\_REPORTS"."XNG\_VLAN\_PARSED\_WK" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EDATE\_BSM\_METRO\_INV\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EDATE\_BSM\_METRO\_INV\_IDX" ON "XNG\_REPORTS"."BSM\_METRO\_INV" ("EXTRACT\_DATE", "VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER", "BCKHTYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_CIRC\_PATH\_INST

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_CIRC\_PATH\_INST" ON "XNG\_REPORTS"."XCONN\_CIRC\_PATH\_DETAILS" ("XNG\_CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_XREF\_WK

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_XREF\_WK" ON "XNG\_REPORTS"."BTP\_DATA\_WRK" ("BAN\_CATEGORY\_1")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EC\_INS\_DATE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EC\_INS\_DATE\_IDX" ON "XNG\_REPORTS"."EBH\_CKTS" ("IN\_SERVICE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index VSM\_OMCR\_UNQ\_IDX

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."VSM\_OMCR\_UNQ\_IDX" ON "XNG\_REPORTS"."MT\_VSM\_SANE\_RAW\_OMCR\_NAME" ("VSM\_DEVICE\_NAME\_OMCR")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVNG\_PATH\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVNG\_PATH\_IDX2" ON "XNG\_REPORTS"."XNG\_VLAN\_NGMLS\_MAP\_WK" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index WO\_Q\_QUEUE\_AUDIT\_WO\_T\_NVAL\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."WO\_Q\_QUEUE\_AUDIT\_WO\_T\_NVAL\_IDX" ON "XNG\_REPORTS"."WO\_BY\_Q\_TEMP\_AUDIT" ("INST\_ID", "SUBCLASS\_INST\_ID", "NEW\_VALUE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVM\_EQ\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVM\_EQ\_IDX2" ON "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_MAP\_WK" ("EQUIP\_INST\_ID", "CARD\_INST\_ID", "PORT\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index WD\_ERROR\_CODE\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."WD\_ERROR\_CODE\_PK" ON "XNG\_REPORTS"."WD\_ERROR\_CODE" ("ERROR\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNGSTD\_FAULT\_EQUIP\_REGION

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XNGSTD\_FAULT\_EQUIP\_REGION" ON "XNG\_REPORTS"."XNGSTD\_FAULT\_EQUIP\_DETAIL" ("REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_EBH\_HEALTH\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_EBH\_HEALTH\_IDX" ON "XNG\_REPORTS"."SAM\_EBH\_HEALTH" ("SITE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_REPORTS\_WITH\_VH\_U01

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_REPORTS\_WITH\_VH\_U01" ON "XNG\_REPORTS"."XOR\_REPORTS\_WITH\_VH" ("REPORT\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_TABS\_MENUS\_LINK\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_TABS\_MENUS\_LINK\_PK" ON "XNG\_REPORTS"."XOR\_TABS\_MENUS\_LINK" ("MENU\_ID", "TAB\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_VZW\_SEVONE\_TRAFFIC

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_VZW\_SEVONE\_TRAFFIC" ON "XNG\_REPORTS"."VZW\_SEVONE\_TRAFFIC" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_CSR\_PARSED\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XNG\_CSR\_PARSED\_IDX" ON "XNG\_REPORTS"."XNG\_CSR\_PARSED" ("CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NCM\_CSR\_VLAN\_NXA\_NE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NCM\_CSR\_VLAN\_NXA\_NE\_IDX" ON "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDIT\_WK" ("CSR\_VENDOR", "NE\_HOSTNAME", "NE\_VLAN\_NUMBER", "MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index WO\_TASK\_AGING\_DETAIL\_BACKUP

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL\_BACKUP" ON "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL\_BACKUP" ("WO\_INST\_ID", "TASK\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 16777216 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_UPD\_XREF

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_UPD\_XREF" ON "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD" ("BAN\_CATEGORY\_1")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NGNA\_HOST\_VENDOR\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NGNA\_HOST\_VENDOR\_IDX" ON "XNG\_REPORTS"."NCM\_GI\_CI\_NGMLS\_AUDIT\_WK" ("NGMLS\_DEVICE\_NAME", "NGMLS\_VENDOR")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NORTEL\_NE\_VS\_XNG\_AUDIT\_IDX4

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_IDX4" ON "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT" ("NE\_DOM\_IP", "NE\_STATUS", "NE\_DCG\_NUMBER", "NE\_SLOT\_NUMBER", "NE\_SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index LU\_ETHER\_DISCO\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."LU\_ETHER\_DISCO\_IDX" ON "XNG\_REPORTS"."PROS\_ECP\_CELL\_EBH\_MAP" ("SYS\_ID", "ECP\_SID", "CELL\_NUMBER", "URC\_NUM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_EBH\_HEALTH\_DEV\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_EBH\_HEALTH\_DEV\_IDX" ON "XNG\_REPORTS"."SAM\_EBH\_HEALTH" ("DEVICE\_TYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_GIGE\_PARENTS\_GINST

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_GIGE\_PARENTS\_GINST" ON "XNG\_REPORTS"."XNG\_GIGE\_PARENTS\_WK" ("GIGE\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index JUNIPER\_DEVICES\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."JUNIPER\_DEVICES\_PK" ON "XNG\_REPORTS"."JUNIPER\_DEVICES" ("HOST\_NAME", "DEVICE\_IP", "DEVICE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NE\_VS\_XNG\_CSR\_5\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NE\_VS\_XNG\_CSR\_5\_IDX" ON "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XMD\_NE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XMD\_NE\_IDX" ON "XNG\_REPORTS"."XNG\_MSPP\_DEVICES" ("NE\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_CONNECTION\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_CONNECTION\_IDX1" ON "XNG\_REPORTS"."INC\_MSPP\_CONNECTION" ("ALOGICALPORT")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_XREF

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_XREF" ON "XNG\_REPORTS"."BTP\_DATA" ("BAN\_CATEGORY\_1")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MNVXA\_IDX\_2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MNVXA\_IDX\_2" ON "XNG\_REPORTS"."MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT" ("MTX", "MATCH\_CODE", "MATCH\_STATUS", "TERMINATION\_TYPE", "XNG\_PATH\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SITE\_AUDIT\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."SITE\_AUDIT\_PK" ON "XNG\_REPORTS"."SITE\_AUDIT" ("AREA", "REGION", "ATTR\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XLCP\_PT\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XLCP\_PT\_IDX" ON "XNG\_REPORTS"."VZW\_LUCENT\_CELL\_PATHS" ("SWITCH\_ID", "CELL\_NUMBER", "URC\_CRC", "DS1\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_DATA\_PP

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_DATA\_PP" ON "XNG\_REPORTS"."BTP\_DATA\_PP" ("STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MWR\_PATH\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MWR\_PATH\_IDX" ON "XNG\_REPORTS"."XNG\_MWR\_PATHS\_WK" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NDI\_EMS\_IP\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NDI\_EMS\_IP\_IDX" ON "XNG\_REPORTS"."NORTEL\_DOM\_INVENTORY" ("EMS\_NAME", "DOM\_IP")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MNVXA\_IDX\_1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MNVXA\_IDX\_1" ON "XNG\_REPORTS"."MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT" ("MTX", "NE\_OMCR", "NE\_BTS\_NUMBER", "NE\_CLUSTER\_NUMBER", "NE\_SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_TABS\_U01

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_TABS\_U01" ON "XNG\_REPORTS"."XOR\_TABS" ("TAB\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OSS\_RC\_ENB\_WRK\_NODE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."OSS\_RC\_ENB\_WRK\_NODE\_IDX" ON "XNG\_REPORTS"."OSS\_RC\_ENB\_WRK" (SUBSTR("NODE",1,6))

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MOTO\_EVDO\_AUDIT\_WRK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MOTO\_EVDO\_AUDIT\_WRK\_IDX" ON "XNG\_REPORTS"."MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT\_WRK" ("AEMS", "NE\_IPBSCDO\_NUMBER", "NE\_MCCDO\_IDENTIFIER", "NE\_SPAN\_NUMBER", "MATCH\_STATUS", "MATCH\_CODE", "TERMINATION\_TYPE", "XNG\_PATH\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EDATE\_BSBM\_DCG\_CELL\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EDATE\_BSBM\_DCG\_CELL\_IDX" ON "XNG\_REPORTS"."BSM\_CELL\_DCG\_MAP" ("EXTRACT\_DATE", "VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER", "CELL\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XEP\_ENB\_NUM\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XEP\_ENB\_NUM\_IDX" ON "XNG\_REPORTS"."XNG\_ENB\_PARSED" ("ENB\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_REPORTS\_WITH\_VH\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_REPORTS\_WITH\_VH\_PK" ON "XNG\_REPORTS"."XOR\_REPORTS\_WITH\_VH" ("REPORT\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_CSR\_PARSED\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_CSR\_PARSED\_PK" ON "XNG\_REPORTS"."XNG\_CSR\_PARSED" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_ENB\_NODE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_ENB\_NODE\_IDX" ON "XNG\_REPORTS"."SAM\_ENB" (SUBSTR("NODE",1,6))

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index GIGE\_VLAN\_MAP\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."GIGE\_VLAN\_MAP\_IDX1" ON "XNG\_REPORTS"."XNG\_GIGE\_VLAN\_MAP" ("GIGE\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index LNVXA\_WRK\_PATHINST\_STA\_TYP\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."LNVXA\_WRK\_PATHINST\_STA\_TYP\_IDX" ON "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDIT\_WRK" ("CIRC\_PATH\_INST\_ID", "XNG\_PATH\_STATUS", "XNG\_PATH\_TYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CLLI\_DOMAIN\_MAP\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP\_PK" ON "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP" ("CLLI", "LEAF\_DOMAIN\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NNVW\_HOST\_VLAN\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NNVW\_HOST\_VLAN\_IDX" ON "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_WK" ("HOSTNAME", "VLAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index GIGE\_VLAN\_ISSUE\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."GIGE\_VLAN\_ISSUE\_IDX1" ON "XNG\_REPORTS"."GIGE\_VLAN\_AUDIT\_ISSUE" ("GIGE\_VLAN\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CDW\_CSR\_DEVICE\_NAME\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CDW\_CSR\_DEVICE\_NAME\_IDX" ON "XNG\_REPORTS"."CSR\_DEVICES\_WK" ("CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_CSR\_IDX3

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XNG\_CSR\_IDX3" ON "XNG\_REPORTS"."XNG\_CSR\_GIGE\_PATHS" ("NEXT\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS NOLOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index JUNK\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."JUNK\_IDX1" ON "XNG\_REPORTS"."SEGS\_UPGRADED\_EBH" ("SITE\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index VVDM\_UNQ\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."VVDM\_UNQ\_IDX" ON "XNG\_REPORTS"."VSM\_DEVICES\_DOMAIN\_MAP" ("VSM\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CIRC\_SEG\_NEW\_DECOM\_DETAIL\_IDX4

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_IDX4" ON "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" ("BANDWIDTH", "SEGMENT\_TYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PK\_SWITCH\_PATH\_ISSUE

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."PK\_SWITCH\_PATH\_ISSUE" ON "XNG\_REPORTS"."SWITCH\_PATH\_ISSUE" ("SWITCH\_PATH\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_SPTE\_PATH\_INST

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_SPTE\_PATH\_INST" ON "XNG\_REPORTS"."SWITCH\_PATH\_T1\_ENDPOINTS" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_ID\_WK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_ID\_WK\_IDX" ON "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE\_WK" ("CSR\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_SNCNE\_INDEX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NETSMART\_SNCNE\_INDEX2" ON "XNG\_REPORTS"."NETSMART\_SNCNE" ("TID\_TYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_EQPT\_INDEX1

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."NETSMART\_EQPT\_INDEX1" ON "XNG\_REPORTS"."NETSMART\_EQPT" ("SNCNE\_ID", "COMPONENT\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_VZW\_EHEALTH\_TRAFFIC

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_VZW\_EHEALTH\_TRAFFIC" ON "XNG\_REPORTS"."VZW\_EHEALTH\_TRAFFIC" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NG\_VLAN\_NXA\_XNG\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NG\_VLAN\_NXA\_XNG\_IDX" ON "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_AUDIT\_WK" ("NGMLS\_VENDOR", "NE\_INST\_ID", "XNG\_VLAN\_NUMBER", "VLAN\_STATUS", "MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PID\_EID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."PID\_EID\_IDX" ON "XNG\_REPORTS"."WD\_ERROR\_DETAILS" ("PROCESS\_ID", "ERROR\_ID", "LOG\_DATE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_XCONN\_EQUIP\_INST\_ID

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_XCONN\_EQUIP\_INST\_ID" ON "XNG\_REPORTS"."XCONN\_AUDIT\_DETAILS" ("XNG\_EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index LNVXA\_PATHINST\_STATUS\_TYPE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."LNVXA\_PATHINST\_STATUS\_TYPE\_IDX" ON "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDIT" ("CIRC\_PATH\_INST\_ID", "XNG\_PATH\_STATUS", "XNG\_PATH\_TYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SEGMENTS\_TERM\_EXPIRATION\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION\_PK" ON "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION" ("CIRC\_INST\_ID", "DOMAIN")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_GIGE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_GIGE\_IDX" ON "XNG\_REPORTS"."XNG\_CSR\_GIGE\_PATHS" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XLCP\_INST\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XLCP\_INST\_IDX" ON "XNG\_REPORTS"."VZW\_LUCENT\_CELL\_PATHS" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_CONNECTION\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_CONNECTION\_IDX2" ON "XNG\_REPORTS"."INC\_MSPP\_CONNECTION" ("ZLOGICALPORT")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_PROTECTIONPORT\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_PROTECTIONPORT\_IDX2" ON "XNG\_REPORTS"."INC\_MSPP\_PROTECTPORT" ("WORKING\_PORT")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index TDM\_CKT\_TO\_EBH\_SITE\_STAT\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."TDM\_CKT\_TO\_EBH\_SITE\_STAT\_IDX" ON "XNG\_REPORTS"."TDM\_CKTS\_TO\_EBH\_SITES" ("STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_EBH\_PORT\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_EBH\_PORT\_IDX" ON "XNG\_REPORTS"."SAM\_EBH\_PORT" ("SITE\_NAME", "PORT\_NAME", "DEVICE\_TYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EBH\_DISCONNECT\_NDF\_PATH\_UDASPK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."EBH\_DISCONNECT\_NDF\_PATH\_UDASPK" ON "XNG\_REPORTS"."LEASED\_SEG\_DF\_NDF\_PATH\_UDAS\_WK" ("NDF\_PATH\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MOTO\_BTS\_LOCATION\_MAP\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MOTO\_BTS\_LOCATION\_MAP\_IDX1" ON "XNG\_REPORTS"."MOTO\_BTS\_LOCATION\_MAP" ("VSM\_DEVICE\_NAME\_OMCR", "BTS\_NUMBER", "BTS\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_GV\_MN\_QT\_MAP

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_GV\_MN\_QT\_MAP" ON "XNG\_REPORTS"."GIGE\_VLAN\_MSPP\_NGMLS\_QT\_MAP\_LR" ("GIGE\_INST\_ID", "VLAN\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index TDM\_CKT\_TO\_EBH\_SITE\_INS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."TDM\_CKT\_TO\_EBH\_SITE\_INS\_IDX" ON "XNG\_REPORTS"."TDM\_CKTS\_TO\_EBH\_SITES" ("EBH\_IN\_SERVICE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NGNA\_AREA\_REG\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NGNA\_AREA\_REG\_IDX" ON "XNG\_REPORTS"."NCM\_GI\_CI\_NGMLS\_AUDIT\_WK" ("AREA", "REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NT\_AUDIT\_WRK\_IDX3

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NT\_AUDIT\_WRK\_IDX3" ON "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK" ("VSM\_DEVICE\_NAME\_BSM", "NE\_DCG\_NUMBER", "NE\_SLOT\_NUMBER", "NE\_SPAN\_NUMBER", "VSM\_DEVICE\_NAME\_MTX")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AI\_VSM\_AEMS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AI\_VSM\_AEMS\_IDX" ON "XNG\_REPORTS"."MT\_AEMS\_IPBSCDO" ("VSM\_DEVICE\_NAME\_AEMS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BSM\_DCG\_DS1\_WRK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BSM\_DCG\_DS1\_WRK\_IDX" ON "XNG\_REPORTS"."BSM\_DCG\_SPAN\_VOICE\_MAP\_WRK" ("VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER", "SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_OPTICAL\_LINK\_ZTID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_OPTICAL\_LINK\_ZTID\_IDX" ON "XNG\_REPORTS"."INC\_MSPP\_OPTICAL\_LINK" ("Z\_TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MWR\_PATH\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MWR\_PATH\_IDX2" ON "XNG\_REPORTS"."XNG\_MWR\_PATHS\_WK" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index GIGE\_VLAN\_AUDIT\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."GIGE\_VLAN\_AUDIT\_IDX2" ON "XNG\_REPORTS"."GIGE\_VLAN\_AUDIT\_ISSUE" ("GIGE\_VLAN\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_DATA\_WK

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_DATA\_WK" ON "XNG\_REPORTS"."BTP\_DATA\_WRK" ("STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_REPORTS\_U01

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_REPORTS\_U01" ON "XNG\_REPORTS"."XOR\_REPORTS" ("REPORT\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NPI\_SEQNO\_PORT\_TYPE\_SPAN\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NPI\_SEQNO\_PORT\_TYPE\_SPAN\_IDX" ON "XNG\_REPORTS"."NORTEL\_PORT\_INVENTORY" ("DOM\_SEQ\_NUM", "PORT\_TYPE", "PORT\_NUM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CAI\_EQ\_WK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CAI\_EQ\_WK\_IDX" ON "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE\_WK" ("XNG\_EQUIP\_INST\_ID", "CSR\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CKT\_DISCO\_AGING\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CKT\_DISCO\_AGING\_IDX" ON "XNG\_REPORTS"."DISCONNECT\_AGING" ("CIRC\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_SNCNE\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."NETSMART\_SNCNE\_PK" ON "XNG\_REPORTS"."NETSMART\_SNCNE" ("INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNGSTD\_DOMAIN\_REGION\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XNGSTD\_DOMAIN\_REGION\_IDX" ON "XNG\_REPORTS"."XNGSTD\_DOMAIN\_REGION" ("REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NT\_AUDIT\_WRK\_IDX4

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NT\_AUDIT\_WRK\_IDX4" ON "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK" ("NE\_DOM\_IP", "NE\_STATUS", "NE\_DCG\_NUMBER", "NE\_SLOT\_NUMBER", "NE\_SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_CSR\_WK\_IDX3

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XNG\_CSR\_WK\_IDX3" ON "XNG\_REPORTS"."XNG\_CSR\_GIGE\_PATHS\_WK" ("NEXT\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_XCONN\_CIRC\_PATH\_INST

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_XCONN\_CIRC\_PATH\_INST" ON "XNG\_REPORTS"."XCONN\_AUDIT\_DETAILS" ("XNG\_CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVP\_VLAN\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVP\_VLAN\_IDX" ON "XNG\_REPORTS"."XNG\_VLAN\_PARENTS" ("VLAN\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NFE\_EQUIP\_INST\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NFE\_EQUIP\_INST\_IDX" ON "XNG\_REPORTS"."NETSMART\_FUJITSU\_EQUIPMENT" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SANE\_OMCR\_UNQ\_IDX

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."SANE\_OMCR\_UNQ\_IDX" ON "XNG\_REPORTS"."MT\_VSM\_SANE\_RAW\_OMCR\_NAME" ("SANE\_NAME\_OMCR")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_PROTECTIONPORT\_IDX3

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_PROTECTIONPORT\_IDX3" ON "XNG\_REPORTS"."INC\_MSPP\_PROTECTPORT" ("PROTECT\_PORT")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index WO\_TASK\_AGING\_DETAIL\_AUDIT

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL\_AUDIT" ON "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL\_AUDIT" ("WO\_INST\_ID", "TASK\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 16777216 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MWR\_PATH\_IDX3

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MWR\_PATH\_IDX3" ON "XNG\_REPORTS"."XNG\_MWR\_PATHS\_WK" ("NEXT\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XC\_TECH\_NAME\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XC\_TECH\_NAME\_IDX" ON "XNG\_REPORTS"."XNG\_CONTACTS" ("TECH\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_NE\_U01

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."INC\_MSPP\_NE\_U01" ON "XNG\_REPORTS"."INC\_MSPP\_NE" ("TID", "GNE\_TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_DCOM\_PATH\_DETAIL\_2IX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL\_2IX" ON "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" ("REPORT\_ID", "CHANGED\_BY", "DECOMMISSION\_DATE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_DCOM\_PATH\_SUMMARY\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_SUMMARY\_PK" ON "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_SUMMARY" ("REPORT\_ID", "ROW\_NUM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MOTO\_CDMA\_EI\_OMCR\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MOTO\_CDMA\_EI\_OMCR\_IDX" ON "XNG\_REPORTS"."MOTO\_CDMA\_ETHERNET\_INV" ("VSM\_DEVICE\_NAME\_OMCR")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NS\_ADM\_XNG\_PORTS\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NS\_ADM\_XNG\_PORTS\_IDX1" ON "XNG\_REPORTS"."NETSMART\_ADM\_XNG\_PORTS" ("TID", "PORT\_HUM\_ID", "SLOT"||'-'||"PORT\_HUM\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_SPTD

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."IDX\_SPTD" ON "XNG\_REPORTS"."SWITCH\_PATH\_T1\_DACSPORTS" ("SWITCH\_PATH\_T1\_TESTABILITY\_ID", "CIRC\_PATH\_INST\_ID", "EQUIP\_INST\_ID", "PORT\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_DEVICES\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."CSR\_DEVICES\_PK" ON "XNG\_REPORTS"."CSR\_DEVICES" ("CSR\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index GIGE\_VLAN\_MAP\_WK\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."GIGE\_VLAN\_MAP\_WK\_IDX1" ON "XNG\_REPORTS"."XNG\_GIGE\_VLAN\_MAP\_WK" ("GIGE\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_SS7\_PARSED\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_SS7\_PARSED\_PK" ON "XNG\_REPORTS"."XNG\_SS7\_PATHS\_PARSED" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_SEVONE\_PARSED\_DEVNAME

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_SEVONE\_PARSED\_DEVNAME" ON "XNG\_REPORTS"."SEVONE\_PERFORMANCE\_DATA" ("DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index GIGE\_VLAN\_MAP\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."GIGE\_VLAN\_MAP\_IDX2" ON "XNG\_REPORTS"."XNG\_GIGE\_VLAN\_MAP" ("VLAN\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_MENUS\_REPORTS\_LINK\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_MENUS\_REPORTS\_LINK\_PK" ON "XNG\_REPORTS"."XOR\_MENUS\_REPORTS\_LINK" ("REPORT\_ID", "MENU\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_SUMMIT\_SEG\_EXTRACT\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT\_PK" ON "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT" ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "CIRC\_INST\_ID", "NAURU\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_CSR\_PARSED\_WK\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_CSR\_PARSED\_WK\_PK" ON "XNG\_REPORTS"."XNG\_CSR\_PARSED\_WK" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EC\_Z\_SITE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EC\_Z\_SITE\_IDX" ON "XNG\_REPORTS"."EBH\_CKTS" ("Z\_SITE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_SUMMIT\_EXTRACT\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_SUMMIT\_EXTRACT\_PK" ON "XNG\_REPORTS"."XNG\_SUMMIT\_EXTRACT" ("SITE\_INST\_ID", "SITE\_DESCRIPTION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index VZW\_NT\_VOICE\_CELL\_PATHS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."VZW\_NT\_VOICE\_CELL\_PATHS\_IDX" ON "XNG\_REPORTS"."VZW\_NT\_VOICE\_CELL\_PATHS" ("SWITCH\_ID", "DCG\_NUMBER", "SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_DCOM\_PATH\_NOTIF\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIF\_PK" ON "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" ("USER\_NAME", "CIRC\_PATH\_INST\_ID", "AREA", "REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_CONNECTION\_TID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_CONNECTION\_TID\_IDX" ON "XNG\_REPORTS"."INC\_MSPP\_CONNECTION" ("TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVP\_VLAN\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVP\_VLAN\_IDX2" ON "XNG\_REPORTS"."XNG\_VLAN\_PARENTS\_WK" ("VLAN\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index LU\_T1\_DISCO\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."LU\_T1\_DISCO\_IDX" ON "XNG\_REPORTS"."PROS\_ECP\_CELL\_DS1\_MAP" ("SYS\_ID", "ECP\_SID", "CELL\_NUMBER", "URC\_NUM", "DS1\_NUM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_UPD\_XREF\_OLD

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_UPD\_XREF\_OLD" ON "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_OLD" ("BAN\_CATEGORY\_1")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MONTHLY\_MRC\_BY\_REGION\_22

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION\_22" ON "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION" ("STRIPPED\_EC\_CIRCUIT\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_DCOM\_PATH\_DETAIL\_1IX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL\_1IX" ON "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" ("AREA", "REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index LNVXA\_WRK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."LNVXA\_WRK\_IDX" ON "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDIT\_WRK" ("SYS\_ID", "ECP\_SID", "NE\_CELL\_NUM", "NE\_URC\_NUM", "NE\_DS1\_NUM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OU\_IDDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."OU\_IDDX2" ON "XNG\_REPORTS"."OPSTRACKER\_USERS" ("FNAME", "LNAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_CIRC\_DETAILS\_STATUS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_CIRC\_DETAILS\_STATUS" ON "XNG\_REPORTS"."XCONN\_CIRC\_PATH\_DETAILS" ("PATH\_DETAILS\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BSM\_MTX\_MAP\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BSM\_MTX\_MAP\_IDX1" ON "XNG\_REPORTS"."BSM\_MTX\_MAP" ("VSM\_DEVICE\_NAME\_BSM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MONTHLY\_MRC\_BY\_REGION\_1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION\_1" ON "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION\_OLD" ("INVENTORY\_VENDOR\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XCONN\_AUDIT\_DETAILS\_STATUS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XCONN\_AUDIT\_DETAILS\_STATUS" ON "XNG\_REPORTS"."XCONN\_AUDIT\_DETAILS" ("DETAILS\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SEGMENT\_PATH\_AUDIT\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SEGMENT\_PATH\_AUDIT\_IDX2" ON "XNG\_REPORTS"."INCOMPLETE\_PATH" ("CIRC\_HUM\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INST\_AUD\_WINDOW\_ELEM\_TYPE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INST\_AUD\_WINDOW\_ELEM\_TYPE\_IDX" ON "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW" ("ELEMENT\_TYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_SPTT\_PATH\_INST

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_SPTT\_PATH\_INST" ON "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_REPORTS\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_REPORTS\_PK" ON "XNG\_REPORTS"."XOR\_REPORTS" ("REPORT\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index ECL\_HOST\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."ECL\_HOST\_IDX" ON "XNG\_REPORTS"."EHEALTH\_CSR\_LIST\_SAVE" ("HOSTNAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index ORDER\_REQD\_FIELDS\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."ORDER\_REQD\_FIELDS\_IDX2" ON "XNG\_REPORTS"."ORDER\_REQD\_FIELDS" ("CIRC\_HUM\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_MENUS\_U01

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_MENUS\_U01" ON "XNG\_REPORTS"."XOR\_MENUS" ("MENU\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OU\_IDX3

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."OU\_IDX3" ON "XNG\_REPORTS"."OPSTRACKER\_USERS" ("LNAME", "FNAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NCM\_CSR\_VLAN\_CSR\_NE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NCM\_CSR\_VLAN\_CSR\_NE\_IDX" ON "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDIT\_WK" ("CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MNVXA\_WRK\_IDX\_2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MNVXA\_WRK\_IDX\_2" ON "XNG\_REPORTS"."MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK" ("MTX", "MATCH\_CODE", "MATCH\_STATUS", "TERMINATION\_TYPE", "XNG\_PATH\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NT\_AUDIT\_WRK\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NT\_AUDIT\_WRK\_IDX2" ON "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK" ("TERMINATION\_TYPE", "NE\_SPAN\_TYPE", "MATCH\_CODE", "MATCH\_STATUS", "XNG\_PATH\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_CIRC\_CONTAINER\_ID

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_CIRC\_CONTAINER\_ID" ON "XNG\_REPORTS"."XCONN\_CIRC\_PATH\_DETAILS" ("XNG\_CONTAINER\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OBS\_SANE\_BTS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."OBS\_SANE\_BTS\_IDX" ON "XNG\_REPORTS"."MT\_OMCR\_BTS\_STATUS" ("SANE\_NAME\_OMCR", "BTS\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_SUMMIT\_SITE\_EXTRACT\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT\_PK" ON "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT" ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "NAURU\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PROCESS\_DEPENDENCY\_MATRIX\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."PROCESS\_DEPENDENCY\_MATRIX\_PK" ON "XNG\_REPORTS"."PROCESS\_DEPENDENCY\_MATRIX" ("PROCESS\_ID", "DS\_PROCESS\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_CSR\_PARSED\_WK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XNG\_CSR\_PARSED\_WK\_IDX" ON "XNG\_REPORTS"."XNG\_CSR\_PARSED\_WK" ("CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index TEST\_HEAD\_XREF\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."TEST\_HEAD\_XREF\_PK" ON "XNG\_REPORTS"."TEST\_HEAD\_XREF" ("TEST\_HEAD\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_TABS\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_TABS\_PK" ON "XNG\_REPORTS"."XOR\_TABS" ("TAB\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XOR\_MENUS\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XOR\_MENUS\_PK" ON "XNG\_REPORTS"."XOR\_MENUS" ("MENU\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_DCOM\_PATH\_NOTIF\_1IX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIF\_1IX" ON "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" ("STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_XCONN\_AUDIT\_SUMMA\_U01

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_SUMMA\_U01" ON "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_SUMMARY" ("TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_SUMMIT\_ANT\_EXTRACT\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT\_PK" ON "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT" ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "EQUIP\_INST\_ID", "NAURU\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INDX\_SEG\_MOD\_CIRC\_INST\_ID

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INDX\_SEG\_MOD\_CIRC\_INST\_ID" ON "XNG\_REPORTS"."SEGMENTS\_MODIFIED" ("CIRC\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CIRC\_SEG\_NEW\_DECOM\_DETAIL\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_IDX2" ON "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" ("CIRC\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NE\_VS\_XNG\_CSR\_3\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NE\_VS\_XNG\_CSR\_3\_IDX" ON "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT" ("MATCH\_CODE", "MATCH\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NE\_VS\_XNG\_CSR\_WK\_3\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NE\_VS\_XNG\_CSR\_WK\_3\_IDX" ON "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT\_WK" ("MATCH\_CODE", "MATCH\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_PHYSICALPORT\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_PHYSICALPORT\_IDX1" ON "XNG\_REPORTS"."INC\_MSPP\_PHYSICALPORT" ("PORTADDRESS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index ALL\_PROCESSES\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."ALL\_PROCESSES\_PK" ON "XNG\_REPORTS"."ALL\_PROCESSES" ("PROCESS\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_SNCNE\_INDEX1

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."NETSMART\_SNCNE\_INDEX1" ON "XNG\_REPORTS"."NETSMART\_SNCNE" ("TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_CIRC\_PATH\_ELEMENTS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_CIRC\_PATH\_ELEMENTS" ON "XNG\_REPORTS"."XCONN\_CIRC\_PATH\_DETAILS" ("XNG\_PATH\_ELEMENTS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OSS\_RC\_ENB\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."OSS\_RC\_ENB\_PK" ON "XNG\_REPORTS"."OSS\_RC\_ENB" ("NODE", "SERVER\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_XCONNECT\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."NETSMART\_XCONNECT\_PK" ON "XNG\_REPORTS"."NETSMART\_XCONNECT" ("INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NG\_VLAN\_NXA\_NE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NG\_VLAN\_NXA\_NE\_IDX" ON "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_AUDIT\_WK" ("NGMLS\_VENDOR", "NE\_HOSTNAME", "NE\_VLAN\_NUMBER", "MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NORTEL\_NE\_VS\_XNG\_AUDIT\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_IDX2" ON "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT" ("TERMINATION\_TYPE", "NE\_SPAN\_TYPE", "MATCH\_CODE", "MATCH\_STATUS", "XNG\_PATH\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INST\_AUD\_WINDOW\_CHG\_TS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INST\_AUD\_WINDOW\_CHG\_TS\_IDX" ON "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW" ("CHG\_TS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_LOGICALPORT\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT\_IDX1" ON "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT" ("LOGICALPORTADDRESS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_NCM\_NGMLS\_VLAN\_ISSUES1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_NCM\_NGMLS\_VLAN\_ISSUES1" ON "XNG\_REPORTS"."NGMLS\_VLAN\_AUDIT\_ISSUES" ("NGMLS\_DEVICE\_NAME", "VLAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EDATE\_BSM\_DCG\_SPAN\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EDATE\_BSM\_DCG\_SPAN\_IDX" ON "XNG\_REPORTS"."BSM\_DCG\_SPAN\_VOICE\_MAP" ("EXTRACT\_DATE", "VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER", "SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index COPY\_WK\_TABLES\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."COPY\_WK\_TABLES\_PK" ON "XNG\_REPORTS"."COPY\_WK\_TABLES" ("PROCESS\_ID", "SEQ\_NO")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OU\_DATA\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."OU\_DATA\_IDX1" ON "XNG\_REPORTS"."OPSTRACKER\_ALL\_DATA" ("TECH\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNG\_VLAN\_PARSED\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."XNG\_VLAN\_PARSED\_PK" ON "XNG\_REPORTS"."XNG\_VLAN\_PARSED" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MOTO\_CDMA\_SPANS\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."MOTO\_CDMA\_SPANS\_PK" ON "XNG\_REPORTS"."MOTO\_CDMA\_SPANS" ("ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_EBH\_HEAL\_SER\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_EBH\_HEAL\_SER\_IDX" ON "XNG\_REPORTS"."SAM\_EBH\_HEALTH" ("SITE\_NAME", "SERVER\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index STRIPPED\_XNG\_CIRC\_ID

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."STRIPPED\_XNG\_CIRC\_ID" ON "XNG\_REPORTS"."STRIPPED\_XNG\_CIRCUIT\_DATA" ("XNG\_CIRC\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NE\_VS\_XNG\_CSR\_WK\_1\_IDA

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NE\_VS\_XNG\_CSR\_WK\_1\_IDA" ON "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT\_WK" ("XNG\_CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XEP\_EQUIP\_INST\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XEP\_EQUIP\_INST\_IDX" ON "XNG\_REPORTS"."XNG\_ENB\_PARSED" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_PHYSICALPORT\_TID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_PHYSICALPORT\_TID\_IDX" ON "XNG\_REPORTS"."INC\_MSPP\_PHYSICALPORT" ("TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PK\_EMEMPTION\_ID

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."PK\_EMEMPTION\_ID" ON "XNG\_REPORTS"."T1\_EXEMPTIONS\_ID\_DEL" ("T1\_EXEMPTION\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NT\_AUDIT\_WRK\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NT\_AUDIT\_WRK\_IDX1" ON "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK" ("VSM\_DEVICE\_NAME\_EMS", "VSM\_DEVICE\_NAME\_MTX", "SWITCH\_ID", "VSM\_DEVICE\_NAME\_BSM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_XCONNECT\_INDEX1

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."NETSMART\_XCONNECT\_INDEX1" ON "XNG\_REPORTS"."NETSMART\_XCONNECT" ("SNCNE\_ID", "XCONNECT\_ID", "XCONNECT\_ASIDE", "XCONNECT\_ZSIDE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index STRIPPED\_XNG\_CIRC\_VND\_BAN

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."STRIPPED\_XNG\_CIRC\_VND\_BAN" ON "XNG\_REPORTS"."STRIPPED\_XNG\_CIRCUIT\_DATA" ("XNG\_CIRC\_STRIP", "XNG\_VENDOR\_STRIP", "XNG\_BAN\_STRIP")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 58720256 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index ENB\_AUD\_NE\_MKT\_ID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."ENB\_AUD\_NE\_MKT\_ID\_IDX" ON "XNG\_REPORTS"."NE\_VS\_XNG\_ENB\_AUDIT" ("NE\_MARKET\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_CONTROL\_2IX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."REPORT\_CONTROL\_2IX" ON "XNG\_REPORTS"."REPORT\_CONTROL" ("RUN\_DATE", "RUN\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_CSR\_VLAN\_CSR\_NE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_CSR\_VLAN\_CSR\_NE\_IDX" ON "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT\_WK" ("CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index ALL\_PROCESSES\_U01

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."ALL\_PROCESSES\_U01" ON "XNG\_REPORTS"."ALL\_PROCESSES" ("PROCESS\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_NE\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."INC\_MSPP\_NE\_PK" ON "XNG\_REPORTS"."INC\_MSPP\_NE" ("NEID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XMS\_NODE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XMS\_NODE\_IDX" ON "XNG\_REPORTS"."XMS\_ENB\_DEL" ("NODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index VZW\_NT\_EVDO\_CELL\_PATHS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."VZW\_NT\_EVDO\_CELL\_PATHS\_IDX" ON "XNG\_REPORTS"."VZW\_NT\_EVDO\_CELL\_PATHS" ("DOM\_NODE\_IP", "SPAN\_NUMBER", "BANDWIDTH")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index LNVXA\_WRK\_BTS\_URC\_DS1\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."LNVXA\_WRK\_BTS\_URC\_DS1\_IDX" ON "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDIT\_WRK" ("SWITCH\_ID", "XNG\_CELL\_NUM", "XNG\_URC\_NUM", "XNG\_DS1\_NUM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MONTHLY\_MRC\_BY\_REGION\_2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION\_2" ON "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION\_OLD" ("STRIPPED\_EC\_CIRCUIT\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_EBH\_RTR\_SER\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_EBH\_RTR\_SER\_IDX" ON "XNG\_REPORTS"."SAM\_EBH\_RTR" ("NODE\_NAME", "SERVER\_NAME", "OUT\_ENCAP\_VAL", "PORT\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVP\_PP\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVP\_PP\_IDX" ON "XNG\_REPORTS"."XNG\_VLAN\_PARENTS" ("PARENT\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index ENB\_AUD\_XNG\_MKT\_ID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."ENB\_AUD\_XNG\_MKT\_ID\_IDX" ON "XNG\_REPORTS"."NE\_VS\_XNG\_ENB\_AUDIT" ("XNG\_MARKET\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MT\_DO\_T1\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MT\_DO\_T1\_IDX" ON "XNG\_REPORTS"."MOTO\_EVDO\_T1\_INV" ("VSM\_DEVICE\_NAME\_AEMS", "IPBSCDO\_NUMBER", "MCCDO\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_LOGICALPORT\_TID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT\_TID\_IDX" ON "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT" ("TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EDATE\_BSM\_DCG\_SPAN\_WRK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EDATE\_BSM\_DCG\_SPAN\_WRK\_IDX" ON "XNG\_REPORTS"."BSM\_DCG\_SPAN\_VOICE\_MAP\_WRK" ("EXTRACT\_DATE", "VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER", "SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SEGMENT\_SUMMARY\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."SEGMENT\_SUMMARY\_PK" ON "XNG\_REPORTS"."SEGMENT\_SUMMARY" ("AREA", "REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVM\_EQ\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVM\_EQ\_IDX" ON "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_MAP" ("EQUIP\_INST\_ID", "CARD\_INST\_ID", "PORT\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVP\_PP\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVP\_PP\_IDX2" ON "XNG\_REPORTS"."XNG\_VLAN\_PARENTS\_WK" ("PARENT\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INDX\_SEG\_MODIFIED\_STATUS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INDX\_SEG\_MODIFIED\_STATUS" ON "XNG\_REPORTS"."SEGMENTS\_MODIFIED" ("NEW\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index GIGE\_VLAN\_MAP\_WK\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."GIGE\_VLAN\_MAP\_WK\_IDX2" ON "XNG\_REPORTS"."XNG\_GIGE\_VLAN\_MAP\_WK" ("VLAN\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_CONTROL\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."REPORT\_CONTROL\_PK" ON "XNG\_REPORTS"."REPORT\_CONTROL" ("REPORT\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_ISSUE\_ID\_WK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_ISSUE\_ID\_WK\_IDX" ON "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE\_WK" ("CSR\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NDI\_IPDCG\_SLOT\_SPAN\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NDI\_IPDCG\_SLOT\_SPAN\_IDX" ON "XNG\_REPORTS"."NORTEL\_DOM\_INVENTORY" ("DOM\_IP", "DCG\_NUMBER", "SLOT\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EDATE\_BSM\_ETHERNET\_INV\_WRK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EDATE\_BSM\_ETHERNET\_INV\_WRK\_IDX" ON "XNG\_REPORTS"."BSM\_ETHERNET\_INV\_WRK" ("EXTRACT\_DATE", "VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index LNVXA\_BTS\_URC\_DS1\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."LNVXA\_BTS\_URC\_DS1\_IDX" ON "XNG\_REPORTS"."LUCENT\_NE\_VS\_XNG\_AUDIT" ("SWITCH\_ID", "XNG\_CELL\_NUM", "XNG\_URC\_NUM", "XNG\_DS1\_NUM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index WO\_TASK\_AGING\_DETAIL

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL" ON "XNG\_REPORTS"."WO\_TASK\_AGING\_DETAIL" ("WO\_INST\_ID", "TASK\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EDATE\_BSBM\_DCG\_CELL\_WRK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EDATE\_BSBM\_DCG\_CELL\_WRK\_IDX" ON "XNG\_REPORTS"."BSM\_CELL\_DCG\_MAP\_WRK" ("EXTRACT\_DATE", "VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER", "CELL\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_GIGE\_WK\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_GIGE\_WK\_IDX2" ON "XNG\_REPORTS"."XNG\_CSR\_GIGE\_PATHS\_WK" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_LOGICALPORT\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT\_IDX2" ON "XNG\_REPORTS"."INC\_MSPP\_LOGICALPORT" ("PORTADDRESS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index TDM\_DISCONNECT\_DURATION\_INDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."TDM\_DISCONNECT\_DURATION\_INDX" ON "XNG\_REPORTS"."TDM\_DISCONNECT" ("DURATION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index ORDER\_REQD\_FIELDS\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."ORDER\_REQD\_FIELDS\_IDX1" ON "XNG\_REPORTS"."ORDER\_REQD\_FIELDS" ("AREA", "REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NDAI\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NDAI\_IDX" ON "XNG\_REPORTS"."NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK" ("NGMLS\_VENDOR", "NGMLS\_DEVICE\_NAME", "NGMLS\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVNG\_PATH\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVNG\_PATH\_IDX" ON "XNG\_REPORTS"."XNG\_VLAN\_NGMLS\_MAP" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_DATA

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_DATA" ON "XNG\_REPORTS"."BTP\_DATA" ("STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SITE\_SUMMARY\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."SITE\_SUMMARY\_PK" ON "XNG\_REPORTS"."SITE\_SUMMARY" ("AREA", "REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_XNG\_AUDIT

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_XNG\_AUDIT" ON "XNG\_REPORTS"."BTP\_XNG\_AUDIT" ("STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OBRGC\_SANE\_BTS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."OBRGC\_SANE\_BTS\_IDX" ON "XNG\_REPORTS"."MT\_OMCR\_BTS\_RTR\_GRP\_CONF" ("SANE\_NAME\_OMCR", "BTS\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XMD\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XMD\_IDX" ON "XNG\_REPORTS"."XNG\_MSPP\_DEVICES" ("INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_NCM\_CSR\_VLAN\_AUDITS1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_NCM\_CSR\_VLAN\_AUDITS1" ON "XNG\_REPORTS"."CSR\_VLAN\_AUDIT\_ISSUES" ("CSR\_DEVICE\_NAME", "VLAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index TEMP\_XNG\_CSR\_PORT\_IX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."TEMP\_XNG\_CSR\_PORT\_IX" ON "XNG\_REPORTS"."TEMP\_XNG\_CSR\_PORT" ("SITE\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MNVXA\_WRK\_IDX\_1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MNVXA\_WRK\_IDX\_1" ON "XNG\_REPORTS"."MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK" ("MTX", "NE\_OMCR", "NE\_BTS\_NUMBER", "NE\_CLUSTER\_NUMBER", "NE\_SPAN\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PROD\_STATS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."PROD\_STATS" ON "XNG\_REPORTS"."PROD\_STATS" ("STATID", "TYPE", "C5", "C1", "C2", "C3", "C4", "VERSION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 81920 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SYSAUX" ;

--------------------------------------------------------

-- DDL for Index NE\_VS\_XNG\_CSR\_2\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NE\_VS\_XNG\_CSR\_2\_IDX" ON "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT" ("NE\_CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index JUNK\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."JUNK\_IDX2" ON "XNG\_REPORTS"."SEGS\_UPGRADED\_EBH" ("CIRC\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EDATE\_BSM\_METRO\_WRK\_INV\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."EDATE\_BSM\_METRO\_WRK\_INV\_IDX" ON "XNG\_REPORTS"."BSM\_METRO\_INV\_WRK" ("EXTRACT\_DATE", "VSM\_DEVICE\_NAME\_BSM", "DCG\_NUMBER", "BCKHTYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_GIGE\_WK\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_GIGE\_WK\_IDX" ON "XNG\_REPORTS"."XNG\_CSR\_GIGE\_PATHS\_WK" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XNGSTD\_DOMAIN\_REGION\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XNGSTD\_DOMAIN\_REGION\_IDX2" ON "XNG\_REPORTS"."XNGSTD\_DOMAIN\_REGION" ("DOMAIN\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index REPORT\_CONTROL\_1IX

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."REPORT\_CONTROL\_1IX" ON "XNG\_REPORTS"."REPORT\_CONTROL" ("REPORT\_NAME", "REPORT\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT\_IDX" ON "XNG\_REPORTS"."MOTO\_EVDO\_NE\_VS\_XNG\_AUDIT" ("AEMS", "NE\_IPBSCDO\_NUMBER", "NE\_MCCDO\_IDENTIFIER", "NE\_SPAN\_NUMBER", "MATCH\_STATUS", "MATCH\_CODE", "TERMINATION\_TYPE", "XNG\_PATH\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index EQUIPMENT\_SUMMARY\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."EQUIPMENT\_SUMMARY\_PK" ON "XNG\_REPORTS"."EQUIPMENT\_SUMMARY" ("AREA", "REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MONTHLY\_MRC\_BY\_REGION\_11

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION\_11" ON "XNG\_REPORTS"."MONTHLY\_MRC\_BY\_REGION" ("INVENTORY\_VENDOR\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INDX\_LAST\_TEST\_SUMMARY

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INDX\_LAST\_TEST\_SUMMARY" ON "XNG\_REPORTS"."NOCC\_LAST\_TEST\_SUMMARY" ("RUN\_DATE" DESC, "DOMAIN\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_GIGE\_IDX2

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_GIGE\_IDX2" ON "XNG\_REPORTS"."XNG\_CSR\_GIGE\_PATHS" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS NOLOGGING

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index TEST\_HEAD\_QT\_XREF\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF\_PK" ON "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF" ("TEST\_HEAD\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_XCONN\_XCONN\_ELEMENTS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_XCONN\_XCONN\_ELEMENTS" ON "XNG\_REPORTS"."XCONN\_AUDIT\_DETAILS" ("DEVICE\_XCONN\_ELEMENTS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PK\_SWITCH\_PATH\_T1\_TESTABILITY

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."PK\_SWITCH\_PATH\_T1\_TESTABILITY" ON "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" ("SWITCH\_PATH\_T1\_TESTABILITY\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SEG\_MIS\_EST\_COST\_REGION\_INDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SEG\_MIS\_EST\_COST\_REGION\_INDX" ON "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST" ("REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NE\_VS\_XNG\_CSR\_WK\_2\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NE\_VS\_XNG\_CSR\_WK\_2\_IDX" ON "XNG\_REPORTS"."HPOV\_VS\_XNG\_CSR\_AUDIT\_WK" ("NE\_CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index AUDIT\_XCONN\_AUDIT\_STATUS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."AUDIT\_XCONN\_AUDIT\_STATUS" ON "XNG\_REPORTS"."XCONN\_AUDIT\_DETAILS" ("AUDIT\_STATUS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_EBH\_RTR\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_EBH\_RTR\_IDX" ON "XNG\_REPORTS"."SAM\_EBH\_RTR" ("OUT\_ENCAP\_VAL", "DEVICE\_TYPE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPPXCONNAUDITDETAILS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPPXCONNAUDITDETAILS\_IDX" ON "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_DETAILS" ("DEVICE\_XCONN\_ELEMENTS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_XNGSTD\_SWITCH\_VENDORS

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."IDX\_XNGSTD\_SWITCH\_VENDORS" ON "XNG\_REPORTS"."XNGSTD\_SWITCH\_VENDORS" ("SWITCH\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PK\_XCONN\_AUDIT

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."PK\_XCONN\_AUDIT" ON "XNG\_REPORTS"."XCONN\_AUDIT" ("XNG\_TARGET\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MT\_X\_DO\_CELL\_PATHS\_2\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MT\_X\_DO\_CELL\_PATHS\_2\_IDX" ON "XNG\_REPORTS"."VZW\_MOTO\_EVDO\_CELL\_PATHS" ("CIRC\_PATH\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_ID\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_ID\_IDX" ON "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE" ("CSR\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SITE\_PORTAL\_DOMAIN\_MAPPING\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."SITE\_PORTAL\_DOMAIN\_MAPPING\_PK" ON "XNG\_REPORTS"."SITE\_PORTAL\_DOMAIN\_MAPPING" ("SITE\_PORTAL\_DOMAIN\_MAPPING\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CSR\_DEVICE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CSR\_DEVICE\_IDX" ON "XNG\_REPORTS"."CSR\_DEVICES" ("CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CIRC\_SEG\_NEW\_DECOM\_DETAIL\_IDX3

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_IDX3" ON "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" ("CHG\_TS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NEWS\_TITLE\_INX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NEWS\_TITLE\_INX" ON "XNG\_REPORTS"."XNG\_NEWS" ("TITLE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_XNGSTD\_PROP\_NAME

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."IDX\_XNGSTD\_PROP\_NAME" ON "XNG\_REPORTS"."XNGSTD\_AUDIT\_PROPERTIES" ("NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index TEMP\_XNG\_ENODE\_B\_IX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B\_IX" ON "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B" ("SITE\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index XVP\_MSPP\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."XVP\_MSPP\_IDX" ON "XNG\_REPORTS"."XNG\_VLAN\_PARENTS\_WK" ("HAS\_MSPP")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_VLAN\_NXA\_XNG\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_VLAN\_NXA\_XNG\_IDX" ON "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT\_WK" ("NGMLS\_VENDOR", "NE\_INST\_ID", "XNG\_VLAN\_NUMBER", "VLAN\_STATUS", "MATCH\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MOTO\_CDMA\_T1\_INV\_OMCR\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MOTO\_CDMA\_T1\_INV\_OMCR\_IDX" ON "XNG\_REPORTS"."MOTO\_CDMA\_T1\_INV" ("VSM\_DEVICE\_NAME\_OMCR")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_SEVONE\_CSR\_DEV\_NAME

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_SEVONE\_CSR\_DEV\_NAME" ON "XNG\_REPORTS"."SEVONE\_CSR\_PARSED\_DATA\_WK" ("DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SEGMENT\_PATH\_AUDIT\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SEGMENT\_PATH\_AUDIT\_IDX1" ON "XNG\_REPORTS"."INCOMPLETE\_PATH" ("REGION")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_NHG\_CSR\_DEVICE

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."IDX\_NHG\_CSR\_DEVICE" ON "XNG\_REPORTS"."CSR\_DEVICE\_AUDIT\_ISSUES" ("CSR\_DEVICE\_NAME")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PK\_T1\_ISSUES\_ID

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."PK\_T1\_ISSUES\_ID" ON "XNG\_REPORTS"."T1\_ISSUES\_ID\_DEL" ("T1\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INST\_AUD\_WINDOW\_ELEM\_INST\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INST\_AUD\_WINDOW\_ELEM\_INST\_IDX" ON "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW" ("INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index PK\_TESTABILITY\_ID

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."PK\_TESTABILITY\_ID" ON "XNG\_REPORTS"."SWITCH\_PATH\_T1\_ENDPOINTS" ("SWITCH\_PATH\_T1\_TESTABILITY\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index TEMP\_XNG\_CSR\_IX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."TEMP\_XNG\_CSR\_IX" ON "XNG\_REPORTS"."TEMP\_XNG\_CSR" ("SITE\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index OU\_USWIN\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."OU\_USWIN\_IDX" ON "XNG\_REPORTS"."OPSTRACKER\_USERS" ("USERID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INC\_MSPP\_PROTECTIONPORT\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INC\_MSPP\_PROTECTIONPORT\_IDX1" ON "XNG\_REPORTS"."INC\_MSPP\_PROTECTPORT" ("TID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index IDX\_SPTI

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."IDX\_SPTI" ON "XNG\_REPORTS"."SWITCH\_PATH\_TESTABILITY\_ISSUE" ("SWITCH\_PATH\_T1\_TESTABILITY\_ID", "SWITCH\_PATH\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index INDX\_LAST\_TEST\_DETAILS

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."INDX\_LAST\_TEST\_DETAILS" ON "XNG\_REPORTS"."NOCC\_LAST\_TEST\_DETAILS" ("RUN\_DATE" DESC, "DOMAIN\_ID", "SITE\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BXA\_CIRC\_INST\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BXA\_CIRC\_INST\_IDX" ON "XNG\_REPORTS"."BTP\_XNG\_AUDIT" ("XNG\_CIRC\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CAI\_EQ\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CAI\_EQ\_IDX" ON "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE" ("XNG\_EQUIP\_INST\_ID", "CSR\_ISSUE\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index SAM\_ENB\_WK\_NODE\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."SAM\_ENB\_WK\_NODE\_IDX" ON "XNG\_REPORTS"."SAM\_ENB\_WK" (SUBSTR("NODE",1,6))

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NFE\_ADM\_EQUIP\_INST\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NFE\_ADM\_EQUIP\_INST\_IDX" ON "XNG\_REPORTS"."NETSMART\_ADM\_FUJITSU\_EQUIPMENT" ("EQUIP\_INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NORTEL\_NE\_VS\_XNG\_AUDIT\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT\_IDX1" ON "XNG\_REPORTS"."NORTEL\_NE\_VS\_XNG\_AUDIT" ("VSM\_DEVICE\_NAME\_EMS", "VSM\_DEVICE\_NAME\_MTX", "SWITCH\_ID", "VSM\_DEVICE\_NAME\_BSM")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index MT\_X\_DO\_CELL\_PATHS\_IDX

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."MT\_X\_DO\_CELL\_PATHS\_IDX" ON "XNG\_REPORTS"."VZW\_MOTO\_EVDO\_CELL\_PATHS" ("IPBSCDO\_NUMBER", "MCCDO\_NUMBER")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index BTP\_DATA\_FOR\_UPD

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD" ON "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD" ("STRIPPED\_EC\_CIRCUIT\_ID", "INVENTORY\_VENDOR\_CODE")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 25165824 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index NETSMART\_SNCNE\_AUDIT\_WK\_PK

--------------------------------------------------------

CREATE UNIQUE INDEX "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT\_WK\_PK" ON "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT\_WK" ("INST\_ID")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Index CIRC\_SEG\_NEW\_DECOM\_DETAIL\_IDX1

--------------------------------------------------------

CREATE INDEX "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_IDX1" ON "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" ("AREA", "REGION", "ACTION\_TYPE", "CHG\_TS")

PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ;

--------------------------------------------------------

-- DDL for Trigger AFTER\_LOGON\_TRG

--------------------------------------------------------

CREATE OR REPLACE TRIGGER "XNG\_REPORTS"."AFTER\_LOGON\_TRG" AFTER

LOGON ON xng\_reports.SCHEMA

BEGIN

EXECUTE IMMEDIATE 'ALTER SESSION SET cursor\_sharing=force';

END;

/

ALTER TRIGGER "XNG\_REPORTS"."AFTER\_LOGON\_TRG" DISABLE;

--------------------------------------------------------

-- DDL for Trigger MOTO\_CDMA\_SPANS\_INS

--------------------------------------------------------

CREATE OR REPLACE TRIGGER "XNG\_REPORTS"."MOTO\_CDMA\_SPANS\_INS"

BEFORE INSERT ON xng\_reports.MOTO\_CDMA\_SPANS

FOR EACH ROW

WHEN (new.id IS NULL) BEGIN

SELECT xng\_reports.moto\_cdma\_spans\_seq.NEXTVAL

INTO :new.id

FROM dual;

END;

/

ALTER TRIGGER "XNG\_REPORTS"."MOTO\_CDMA\_SPANS\_INS" ENABLE;

--------------------------------------------------------

-- DDL for Trigger PUT\_CSR\_DEVICE\_ID

--------------------------------------------------------

CREATE OR REPLACE TRIGGER "XNG\_REPORTS"."PUT\_CSR\_DEVICE\_ID"

BEFORE INSERT

ON CSR\_DEVICES\_WK REFERENCING NEW AS New OLD AS Old

FOR EACH ROW

DECLARE

tmpVar NUMBER;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: put\_csr\_device\_id

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/9/2010 1. Created this trigger.

NOTES:

Automatically available Auto Replace Keywords:

Object Name: put\_csr\_device\_id

Sysdate: 11/9/2010

Date and Time: 11/9/2010, 2:35:30 PM, and 11/9/2010 2:35:30 PM

Username: (set in TOAD Options, Proc Templates)

Table Name: CSR\_DEVICES (set in the "New PL/SQL Object" dialog)

Trigger Options: (set in the "New PL/SQL Object" dialog)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BEGIN

tmpVar := 0;

SELECT csr\_id\_seq.NEXTVAL INTO tmpVar FROM dual;

:NEW.csr\_id := tmpVar;

:NEW.load\_date := SYSDATE;

-- :NEW.CreatedUser := USER;

EXCEPTION

WHEN OTHERS THEN

-- Consider logging the error and then re-raise

RAISE;

END put\_csr\_device\_id;

/

ALTER TRIGGER "XNG\_REPORTS"."PUT\_CSR\_DEVICE\_ID" ENABLE;

--------------------------------------------------------

-- DDL for Procedure BTP\_XNG\_AUDIT\_CREATE\_VERSIONS

--------------------------------------------------------

set define off;

CREATE OR REPLACE PROCEDURE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_CREATE\_VERSIONS" (vtype IN VARCHAR2 default 'in progress') as

--

-- Purpose: Create historical copies of XNG\_REPORTS.BTP\_XNG\_AUDIT and

-- XNG\_REPORTS.BTP\_XNG\_AUDIT\_HISTORY

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- Cathy Sayre 12/10/10

-- --------- ------ -------------------------------------------

v\_btp\_upd char(1);

v\_date varchar2(8);

v\_tname varchar2(50);

v\_sname varchar2(50);

v\_hist\_tname varchar2(50);

v\_data\_tname varchar2(50);

v\_data\_wname varchar2(50);

v\_hist\_sname varchar2(50);

v\_pname varchar2(50);

v\_tcnt number;

v\_deltname varchar2(50);

v\_deltcnt number;

v\_rptd\_start xng\_reports.all\_processes.start\_time%type;

processName xng\_reports.all\_processes.process\_name%type;

sql\_stmt varchar2(32767);

type results is ref cursor;

dar results;

BEGIN

v\_date := to\_char(sysdate,'yyyymmdd');

v\_btp\_upd := 'Y';

v\_tcnt := 0;

v\_deltcnt := 0;

--check to see if btp\_data has been reloaded and if new month and the 3rd or later and get new history file name

select case

when a.btp\_extract\_date < b.btp\_extract\_date

then 'Y'

else 'N'

end btp\_upd,

'BTP\_XNG\_AUDIT\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_XNG\_AUDIT\_HIST\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_XNG\_AUDIT\_SUMMARY\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_XNG\_AUDIT\_SUM\_HST\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_DATA\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_DATA\_PROCESS\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_DATA\_WRK\_'||to\_char(sysdate,'yyyymmdd')

into v\_btp\_upd, v\_tname, v\_hist\_tname, v\_sname, v\_hist\_sname, v\_data\_tname, v\_pname, v\_data\_wname

from (select d1, btp\_extract\_date, to\_char(btp\_extract\_date,'mm') as btp\_extract\_mon

from (select 1 as d1, nvl(min(btp\_extract\_date),sysdate -1) as btp\_extract\_date

from xng\_reports.btp\_xng\_audit)) a,

(select 1 as d1, to\_date(btp\_extract\_date, 'yyyymmdd') as btp\_extract\_date, substr(btp\_extract\_date,5,2) as btp\_extract\_mon,

substr(btp\_extract\_date,7,2) as btp\_extract\_day

from xng\_reports.btp\_data where rownum = 1) b

where a.d1 = b.d1;

--added this so it runs everyday

v\_btp\_upd := 'Y';

if v\_btp\_upd <> 'N' then

--check if btp\_xng\_audit file already exists

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_tname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_tname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_tname||

'(BAN\_CATEGORY\_1 VARCHAR2(50 BYTE),

BAN\_CATEGORY\_2 VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_1 VARCHAR2(50 BYTE),

MASTER\_BAN VARCHAR2(50 BYTE),

BAN VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_4 VARCHAR2(50 BYTE),

STRIPPED\_EC\_CIRCUIT\_ID VARCHAR2(60 BYTE),

INVENTORY\_VENDOR\_CODE VARCHAR2(50 BYTE),

CHARGE\_AMOUNT NUMBER,

BILL\_DATE\_MONTH VARCHAR2(20 BYTE),

BILL\_DATE\_YEAR VARCHAR2(20 BYTE),

MANUAL\_ENTRY VARCHAR2(10 BYTE),

BTP\_EXTRACT\_DATE DATE,

BTP\_XNG\_MATCH VARCHAR2(1 BYTE),

XNG\_AREA VARCHAR2(50 BYTE),

XNG\_REGION VARCHAR2(50 BYTE),

XNG\_CIRC\_INST\_ID NUMBER,

XNG\_CIRC\_HUM\_ID VARCHAR2(60 BYTE),

XNG\_TYPE VARCHAR2(30 BYTE),

XNG\_VENDOR VARCHAR2(30 BYTE),

XNG\_BANDWIDTH VARCHAR2(30 BYTE),

XNG\_STATUS VARCHAR2(20 BYTE),

XNG\_CIRC\_PATH\_INST\_ID NUMBER,

XNG\_NEXT\_PATH\_INST\_ID NUMBER,

XNG\_EXTRACT\_DATE DATE,

UNIQUELY\_IDENTIFIED VARCHAR2(1 BYTE),

BILL\_DATE DATE,

BAN\_CUSTOM\_FIELD\_3 VARCHAR2(50 BYTE),

IN\_SERVICE\_DATE DATE,

MATCH\_CODE VARCHAR2(20 BYTE),

MATCH\_STATUS VARCHAR2(50 BYTE),

XNG\_BAN VARCHAR2(50 BYTE))';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_tname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_XNG\_AUDIT)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_AUDIT\_%'

and table\_name not like '%HIST%'

and table\_name not like '%SUM%'

and table\_name not like '%WRK%'

and table\_name <> 'BTP\_XNG\_AUDIT');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_AUDIT\_%'

and table\_name not like '%HIST%'

and table\_name not like '%SUM%'

and table\_name not like '%WRK%'

and table\_name <> 'BTP\_XNG\_AUDIT');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

v\_tcnt := 0;

v\_deltcnt := 0;

--check if btp\_xng\_audit\_summary file already exists

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_pname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_pname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_pname||

'(STRIPPED\_EC\_CIRCUIT\_ID VARCHAR2(60 BYTE),

BTP\_CIRC\_STRIP VARCHAR2(60 BYTE),

BAN\_CATEGORY\_1 VARCHAR2(50 BYTE),

BAN\_CATEGORY\_2 VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_1 VARCHAR2(50 BYTE),

MASTER\_BAN VARCHAR2(50 BYTE),

BAN VARCHAR2(50 BYTE),

BTP\_AMT NUMBER,

BTP\_BAN\_STRIP VARCHAR2(50 BYTE),

INVENTORY\_VENDOR\_CODE VARCHAR2(50 BYTE),

BTP\_BD DATE,

BTP\_BD\_CHAR VARCHAR2(50 BYTE),

BTP\_EXTRACT\_DATE DATE,

BAN\_CUSTOM\_FIELD\_4 VARCHAR2(50 BYTE),

BILL\_DATE\_MONTH VARCHAR2(20 BYTE),

BILL\_DATE\_YEAR VARCHAR2(20 BYTE),

MANUAL\_ENTRY VARCHAR2(10 BYTE),

BAN\_CUSTOM\_FIELD\_3 VARCHAR2(50 BYTE),

IN\_SERVICE\_DATE DATE,

SEQNO NUMBER,

PROCESSED CHAR(1 BYTE))';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_pname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_DATA\_PROCESS)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_PROCESS\_%'

and table\_name <> 'BTP\_DATA\_PROCESS');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_PROCESS\_%'

and table\_name <> 'BTP\_DATA\_PROCESS');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

v\_tcnt := 0;

v\_deltcnt := 0;

--check if btp\_xng\_audit\_summary file already exists

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_sname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_sname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_sname||

'(AREA VARCHAR2(50 BYTE),

REGION VARCHAR2(50 BYTE),

CKT\_COUNT NUMBER,

CKTS\_MATCHED NUMBER,

CKTS\_EXCP NUMBER,

CKTS\_MATCHED\_PCT\_STR VARCHAR2(15 BYTE),

CKTS\_MATCHED\_PCT\_NUM NUMBER,

EXTRACT\_DATE DATE)';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_sname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_AUDIT\_SUMMARY\_%'

and table\_name not like '%PRE3W%'

and table\_name <> 'BTP\_XNG\_AUDIT\_SUMMARY');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_AUDIT\_SUMMARY\_%'

and table\_name not like '%PRE3W%'

and table\_name <> 'BTP\_XNG\_AUDIT\_SUMMARY');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

v\_tcnt := 0;

v\_deltcnt := 0;

--check btp\_data if file already exists

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_data\_wname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_data\_wname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_data\_wname||

'( STRIPPED\_EC\_CIRCUIT\_ID VARCHAR2(60 BYTE),

BAN\_CUSTOM\_FIELD\_1 VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_3 VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_4 VARCHAR2(50 BYTE),

INVENTORY\_VENDOR\_CODE VARCHAR2(50 BYTE),

BAN\_CATEGORY\_1 VARCHAR2(50 BYTE),

BAN\_CATEGORY\_2 VARCHAR2(50 BYTE),

SERV\_ESTAB\_DATE VARCHAR2(50 BYTE),

CHARGE\_AMOUNT VARCHAR2(50 BYTE),

MASTER\_BAN VARCHAR2(50 BYTE),

BAN VARCHAR2(50 BYTE),

BILL\_DATE\_MONTH VARCHAR2(20 BYTE),

BILL\_DATE\_YEAR VARCHAR2(20 BYTE),

BILL\_DATE VARCHAR2(50 BYTE),

MANUAL\_ENTRY VARCHAR2(10 BYTE),

IN\_SERVICE\_DATE VARCHAR2(50 BYTE),

BTP\_EXTRACT\_DATE VARCHAR2(50 BYTE))';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_data\_wname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_DATA\_WRK)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_WRK%' and table\_name <> 'BTP\_DATA\_WRK');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_WRK%' and table\_name <> 'BTP\_DATA\_WRK');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

v\_tcnt := 0;

v\_deltcnt := 0;

--check btp\_dat if file already exists

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_data\_tname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_data\_tname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_data\_tname||

'( STRIPPED\_EC\_CIRCUIT\_ID VARCHAR2(60 BYTE),

BAN\_CUSTOM\_FIELD\_1 VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_3 VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_4 VARCHAR2(50 BYTE),

INVENTORY\_VENDOR\_CODE VARCHAR2(50 BYTE),

BAN\_CATEGORY\_1 VARCHAR2(50 BYTE),

BAN\_CATEGORY\_2 VARCHAR2(50 BYTE),

SERV\_ESTAB\_DATE VARCHAR2(50 BYTE),

CHARGE\_AMOUNT VARCHAR2(50 BYTE),

MASTER\_BAN VARCHAR2(50 BYTE),

BAN VARCHAR2(50 BYTE),

BILL\_DATE\_MONTH VARCHAR2(20 BYTE),

BILL\_DATE\_YEAR VARCHAR2(20 BYTE),

BILL\_DATE VARCHAR2(50 BYTE),

MANUAL\_ENTRY VARCHAR2(10 BYTE),

IN\_SERVICE\_DATE VARCHAR2(50 BYTE),

BTP\_EXTRACT\_DATE VARCHAR2(50 BYTE))';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_data\_tname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_DATA)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_%'

and table\_name not like 'BTP\_DATA\_FOR\_UPD%'

and table\_name not like 'BTP\_DATA\_PROCESS%'

and table\_name not like 'BTP\_DATA\_PP%'

and table\_name not like 'BTP\_DATA\_FOR\_CIRC\_ID%'

and table\_name not like 'BTP\_DATA\_WRK%'

and table\_name <> 'BTP\_DATA');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_%'

and table\_name not like 'BTP\_DATA\_FOR\_UPD%'

and table\_name not like 'BTP\_DATA\_PROCESS%'

and table\_name not like 'BTP\_DATA\_PP%'

and table\_name not like 'BTP\_DATA\_FOR\_CIRC\_ID%'

and table\_name not like 'BTP\_DATA\_WRK%'

and table\_name <> 'BTP\_DATA');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

end if;

if vtype = 'reported' then

--create and populate new version tables

v\_tcnt := 0;

v\_deltcnt := 0;

--check if btp\_xng\_audit\_hist file already exists

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_hist\_tname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_hist\_tname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_hist\_tname||

'( BAN\_CATEGORY\_1 VARCHAR2(50 BYTE),

BAN\_CATEGORY\_2 VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_1 VARCHAR2(50 BYTE),

MASTER\_BAN VARCHAR2(50 BYTE),

BAN VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_4 VARCHAR2(50 BYTE),

STRIPPED\_EC\_CIRCUIT\_ID VARCHAR2(60 BYTE),

INVENTORY\_VENDOR\_CODE VARCHAR2(50 BYTE),

CHARGE\_AMOUNT NUMBER,

BILL\_DATE\_MONTH VARCHAR2(20 BYTE),

BILL\_DATE\_YEAR VARCHAR2(20 BYTE),

MANUAL\_ENTRY VARCHAR2(10 BYTE),

BTP\_EXTRACT\_DATE DATE,

BTP\_XNG\_MATCH VARCHAR2(1 BYTE),

XNG\_AREA VARCHAR2(50 BYTE),

XNG\_REGION VARCHAR2(50 BYTE),

XNG\_CIRC\_INST\_ID NUMBER,

XNG\_CIRC\_HUM\_ID VARCHAR2(60 BYTE),

XNG\_TYPE VARCHAR2(30 BYTE),

XNG\_VENDOR VARCHAR2(30 BYTE),

XNG\_BANDWIDTH VARCHAR2(30 BYTE),

XNG\_STATUS VARCHAR2(20 BYTE),

XNG\_CIRC\_PATH\_INST\_ID NUMBER,

XNG\_NEXT\_PATH\_INST\_ID NUMBER,

XNG\_EXTRACT\_DATE DATE,

UNIQUELY\_IDENTIFIED VARCHAR2(1 BYTE),

BILL\_DATE DATE,

BAN\_CUSTOM\_FIELD\_3 VARCHAR2(50 BYTE),

IN\_SERVICE\_DATE DATE,

MATCH\_CODE VARCHAR2(20 BYTE),

MATCH\_STATUS VARCHAR2(50 BYTE),

XNG\_BAN VARCHAR2(50 BYTE))';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_hist\_tname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_XNG\_AUDIT\_HIST)';

execute immediate sql\_stmt;

commit work;

v\_tcnt := 0;

v\_deltcnt := 0;

--check if btp\_xng\_audit\_sum\_hist file already exists

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_hist\_sname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_hist\_sname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_hist\_sname||

'(AREA VARCHAR2(50 BYTE),

REGION VARCHAR2(50 BYTE),

CKT\_COUNT NUMBER,

CKTS\_MATCHED NUMBER,

CKTS\_EXCP NUMBER,

CKTS\_MATCHED\_PCT\_STR VARCHAR2(15 BYTE),

CKTS\_MATCHED\_PCT\_NUM NUMBER,

EXTRACT\_DATE DATE)';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_hist\_sname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUM\_HST)';

execute immediate sql\_stmt;

commit work;

--populate current tables with what is in btp\_audit from reported run

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.BTP\_XNG\_AUDIT\_HIST';

execute immediate sql\_stmt;

commit work;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.BTP\_XNG\_AUDIT\_HIST

(SELECT \* FROM XNG\_REPORTS.BTP\_XNG\_AUDIT)';

execute immediate sql\_stmt;

commit work;

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUM\_HST';

execute immediate sql\_stmt;

commit work;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUM\_HST

(SELECT \* FROM XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only three past version of hist table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_AUDIT\_HIST\_%'

and table\_name <> 'BTP\_XNG\_AUDIT\_HIST'

and table\_name <> 'BTP\_XNG\_AUDIT\_HIST\_PRE3W'

and table\_name <> 'BTP\_XNG\_AUDIT\_HIST\_PP\_V'

and table\_name not like 'BTP\_XNG\_AUDIT\_HIST\_BKUP%');

if v\_deltcnt > 3 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_AUDIT\_HIST\_%'

and table\_name <> 'BTP\_XNG\_AUDIT\_HIST'

and table\_name <> 'BTP\_XNG\_AUDIT\_HIST\_PRE3W'

and table\_name <> 'BTP\_XNG\_AUDIT\_HIST\_PP\_V'

and table\_name not like 'BTP\_XNG\_AUDIT\_HIST\_BKUP%');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

v\_deltcnt := 0;

--keep only three past version of sum hist table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_AUDIT\_SUM\_HST\_%'

and table\_name <> 'BTP\_XNG\_AUDIT\_SUM\_HST'

and table\_name <> 'BTP\_XNG\_AUDIT\_SUM\_HST\_HLD');

if v\_deltcnt > 3 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_AUDIT\_SUM\_HST\_%'

and table\_name <> 'BTP\_XNG\_AUDIT\_SUM\_HST'

and table\_name <> 'BTP\_XNG\_AUDIT\_SUM\_HST\_HLD');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

--putting update to watchdog here since btp\_xng\_audit\_update does not update reported

processName := 'BTP\_VS\_XNG\_AUDIT\_RPTD';

select start\_time

into v\_rptd\_start

from xng\_reports.all\_processes

where process\_name = 'BTP\_VS\_XNG\_AUDIT';

update all\_processes

set start\_time = v\_rptd\_start

where process\_name = processName;

commit;

-- tell watchDog process completed successfully

watchdog.updateprocessend (processName, 'STATUS\_SUCCESS', 'Y');

end if;

--EXCEPTION

--WHEN OTHERS THEN

-- dbms\_output.put(SQLCODE);

-- dbms\_output.put(SQLERRM);

-- raise;

END;

/

--------------------------------------------------------

-- DDL for Procedure BTP\_XNG\_AUDIT\_TRUNCATES

--------------------------------------------------------

set define off;

CREATE OR REPLACE PROCEDURE "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_TRUNCATES" (p\_tablename in varchar2)

AS

--

-- Purpose: Allows XNG\_APPS to truncate tables in XNG\_REPORTS

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- Cathy Sayre 12/10/10

-- --------- ------ -------------------------------------------

sql\_str VARCHAR2(32767);

BEGIN

sql\_str :=

'truncate table '||p\_tablename||' reuse storage';

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

raise;

END;

/

--------------------------------------------------------

-- DDL for Procedure BTP\_XNG\_UPDATE\_CREATE\_VERSIONS

--------------------------------------------------------

set define off;

CREATE OR REPLACE PROCEDURE "XNG\_REPORTS"."BTP\_XNG\_UPDATE\_CREATE\_VERSIONS" (vtype IN VARCHAR2 default 'update') as

--

-- Purpose: Create historical copies of XNG\_REPORTS.BTP\_XNG\_UPD\_DETAILS and

-- XNG\_REPORTS.BTP\_DATA\_FOR\_UPD tables

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- Cathy Sayre 12/10/10

-- --------- ------ -------------------------------------------

v\_date varchar2(8);

v\_tname varchar2(50);

v\_sname varchar2(50);

v\_pname varchar2(50);

v\_data\_tname varchar2(50);

v\_tcnt number;

v\_deltname varchar2(50);

v\_deltcnt number;

sql\_stmt varchar2(32767);

type results is ref cursor;

dar results;

BEGIN

v\_date := to\_char(sysdate,'yyyymmdd');

v\_tcnt := 0;

v\_deltcnt := 0;

--get new file names

select 'BTP\_XNG\_UPD\_DETAILS\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_XNG\_UPD\_REG\_SUM\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_DATA\_FOR\_UPD\_'||to\_char(sysdate,'yyyymmdd'),

'BTP\_DATA\_FOR\_UPD\_PROC\_'||to\_char(sysdate,'yyyymmdd')

into v\_tname, v\_sname, v\_data\_tname, v\_pname

from (select d1, btp\_extract\_date, to\_char(btp\_extract\_date,'mm') as btp\_extract\_mon

from (select 1 as d1, nvl(min(btp\_extract\_date),sysdate -1) as btp\_extract\_date

from xng\_reports.btp\_xng\_upd\_details)) a,

(select 1 as d1, extract\_date as btp\_extract\_date, to\_char(extract\_date,'mm') as btp\_extract\_mon,

to\_char(extract\_date,'dd') as btp\_extract\_day

from xng\_reports.btp\_data\_for\_upd where rownum = 1) b

where a.d1 = b.d1;

--check if btp\_xng\_upd\_details file already exists - if it does truncate it otherwise create a new one

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_tname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_tname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_tname||

'(STRIPPED\_EC\_CIRCUIT\_ID VARCHAR2(100 BYTE),

INVENTORY\_VENDOR\_CODE VARCHAR2(50 BYTE),

MASTER\_BAN VARCHAR2(50 BYTE),

BAN VARCHAR2(50 BYTE),

BILL\_DATE DATE,

CHARGE\_AMOUNT NUMBER,

LOGGED\_DATE DATE,

BAN\_CATEGORY\_1 VARCHAR2(50 BYTE),

BAN\_CATEGORY\_2 VARCHAR2(50 BYTE),

AREA VARCHAR2(50 BYTE),

REGION VARCHAR2(50 BYTE),

CIRC\_INST\_ID NUMBER,

CIRC\_HUM\_ID VARCHAR2(60 BYTE),

XNG\_VENDOR VARCHAR2(30 BYTE),

XNG\_STATUS VARCHAR2(20 BYTE),

EXCP\_IND VARCHAR2(1 BYTE),

BTP\_EXTRACT\_DATE DATE,

XNG\_EXTRACT\_DATE DATE,

MATCH\_CODE VARCHAR2(50 BYTE),

MATCH\_STATUS VARCHAR2(100 BYTE),

XNG\_BAN VARCHAR2(50 BYTE))';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_tname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_XNG\_UPD\_DETAILS)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_UPD\_DETAILS\_%'

and table\_name not like '%WK%'

and table\_name <> 'BTP\_XNG\_UPD\_DETAILS');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_UPD\_DETAILS\_%'

and table\_name not like '%WK%'

and table\_name <> 'BTP\_XNG\_UPD\_DETAILS');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

v\_tcnt := 0;

v\_deltcnt := 0;

--check if btp\_data\_for\_upd file already exists - if it does truncate it otherwise create a new one

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_data\_tname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_data\_tname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_data\_tname||

'(STRIPPED\_EC\_CIRCUIT\_ID VARCHAR2(100 BYTE),

BAN\_CUSTOM\_FIELD\_1 VARCHAR2(50 BYTE),

BAN\_CUSTOM\_FIELD\_3 VARCHAR2(50 BYTE),

INVENTORY\_VENDOR\_CODE VARCHAR2(50 BYTE),

BAN\_CATEGORY\_1 VARCHAR2(50 BYTE),

BAN\_CATEGORY\_2 VARCHAR2(50 BYTE),

MASTER\_BAN VARCHAR2(50 BYTE),

BAN VARCHAR2(50 BYTE),

BILL\_DATE\_MONTH VARCHAR2(50 BYTE),

BILL\_DATE\_YEAR VARCHAR2(50 BYTE),

BILL\_DATE VARCHAR2(50 BYTE),

CHARGE\_AMOUNT VARCHAR2(50 BYTE),

MANUAL\_ENTRY\_IND VARCHAR2(50 BYTE),

LOGGED\_DATE VARCHAR2(50 BYTE),

INVOICE\_SEQUENCE\_ID VARCHAR2(50 BYTE),

EXTRACT\_DATE VARCHAR2(20 BYTE))';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_data\_tname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_DATA\_FOR\_UPD)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_FOR\_UPD\_%'

and table\_name not like '%NSP%'

and table\_name not like '%TST%'

and table\_name not like '%OLD%'

and table\_name not like '%NEW%'

and table\_name not like '%PROCESS%'

and table\_name <> 'BTP\_DATA\_FOR\_UPD'

and table\_name not like 'BTP\_DATA\_FOR\_UPD\_PROC\_%');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_FOR\_UPD\_%'

and table\_name not like '%NSP%'

and table\_name not like '%TST%'

and table\_name not like '%OLD%'

and table\_name not like '%NEW%'

and table\_name not like '%PROCESS%'

and table\_name <> 'BTP\_DATA\_FOR\_UPD'

and table\_name not like 'BTP\_DATA\_FOR\_UPD\_PROC\_%');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

v\_tcnt := 0;

v\_deltcnt := 0;

--check if btp\_data\_for\_upd\_process file already exists - if it does truncate it otherwise create a new one

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_pname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_pname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_pname||

'(STRIPPED\_EC\_CIRCUIT\_ID VARCHAR2(100 BYTE),

BTP\_CIRC\_STRIP VARCHAR2(100 BYTE),

BAN\_CUSTOM\_FIELD\_1 VARCHAR2(50 BYTE),

INVENTORY\_VENDOR\_CODE VARCHAR2(50 BYTE),

BAN\_CATEGORY\_1 VARCHAR2(50 BYTE),

BAN\_CATEGORY\_2 VARCHAR2(50 BYTE),

MASTER\_BAN VARCHAR2(50 BYTE),

BAN VARCHAR2(50 BYTE),

BTP\_BAN\_STRIP VARCHAR2(50 BYTE),

BTP\_BD DATE,

BTP\_BD\_CHR VARCHAR2(50 BYTE),

BTP\_EXTRACT\_DATE DATE,

LOGGED\_DATE DATE,

BTP\_AMT NUMBER,

SEQNO NUMBER,

PROCESSED CHAR(1 BYTE))';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_pname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_DATA\_FOR\_UPD\_PROCESS)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_FOR\_UPD\_PROC\_%'

and table\_name <> 'BTP\_DATA\_FOR\_UPD\_PROCESS');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_DATA\_FOR\_UPD\_PROC\_%'

and table\_name <> 'BTP\_DATA\_FOR\_UPD\_PROCESS');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

v\_tcnt := 0;

v\_deltcnt := 0;

--check if btp\_xng\_upd\_reg\_sum file already exists - if it does truncate it otherwise create a new one

sql\_stmt := 'select count(\*) as tcnt

from user\_tab\_columns

where table\_name = '''||v\_sname||''' ';

open dar for sql\_stmt;

loop

fetch dar into v\_tcnt;

exit when dar%notfound;

end loop;

close dar;

if v\_tcnt <> 0 then

sql\_stmt := 'TRUNCATE TABLE XNG\_REPORTS.'||v\_sname;

execute immediate sql\_stmt;

commit work;

else

sql\_stmt := 'CREATE TABLE XNG\_REPORTS.'||v\_sname||

'(AREA VARCHAR2(50 BYTE),

REGION VARCHAR2(50 BYTE),

TOTAL\_DISCOVERED NUMBER,

UPDATED\_CNT NUMBER,

EXCEPTION\_CNT NUMBER,

NO\_UPDATE\_NEEDED NUMBER,

PERCENT\_UPDATED NUMBER)';

execute immediate sql\_stmt;

commit work;

end if;

sql\_stmt := 'INSERT INTO XNG\_REPORTS.'||v\_sname||

'(SELECT \* FROM XNG\_REPORTS.BTP\_XNG\_UPD\_REG\_SUMMARY)';

execute immediate sql\_stmt;

commit work;

v\_deltcnt := 0;

--keep only six past version of table

select count(\*) as table\_cnt

into v\_deltcnt

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_UPD\_REG\_SUM\_%'

and table\_name not like 'BTP\_XNG\_UPD\_REG\_SUMMARY%');

if v\_deltcnt > 6 then

select min(table\_name) as table\_name

into v\_deltname

from (select distinct table\_name

from all\_tab\_columns

where owner = 'XNG\_REPORTS'

and table\_name like 'BTP\_XNG\_UPD\_REG\_SUM\_%'

and table\_name not like 'BTP\_XNG\_UPD\_REG\_SUMMARY%');

sql\_stmt := 'DROP TABLE XNG\_REPORTS.'||v\_deltname||' CASCADE CONSTRAINTS';

execute immediate sql\_stmt;

commit work;

end if;

-- EXCEPTION

-- WHEN OTHERS THEN

-- dbms\_output.put(SQLCODE);

-- dbms\_output.put(SQLERRM);

-- raise;

END;

/

--------------------------------------------------------

-- DDL for Procedure CLEANUP\_DISCONNECT\_DATE\_UDA

--------------------------------------------------------

set define off;

CREATE OR REPLACE PROCEDURE "XNG\_REPORTS"."CLEANUP\_DISCONNECT\_DATE\_UDA"

IS

cursor get\_bad\_data is

select C.CIRC\_INST\_ID, C.ATTR\_VALUE from vzwnet.circ\_attr\_settings c , vzwnet.val\_attr\_name v

where

( C.ATTR\_VALUE like '\_\_-\_\_\_-\_\_\_\_\_\_' or C.ATTR\_VALUE like '\_\_-\_\_\_-\_\_\_\_\_')

and C.VAL\_ATTR\_INST\_ID = v.val\_attr\_inst\_id

and v.attr\_name = 'Disconnect Date';

sqlStmt varchar2(32000) := '';

BEGIN

for rec in get\_bad\_data

LOOP

sqlStmt := 'update vzwnet.circ\_attr\_settings d set D.ATTR\_VALUE = substr(''' || rec.attr\_value ||''',1,11)

where D.CIRC\_INST\_ID = ' || rec.circ\_inst\_id ;

dbms\_output.put\_line(sqlStmt);

execute immediate sqlStmt;

commit;

end loop;

EXCEPTION

WHEN others THEN

dbms\_output.put\_line( substr(SQLERRM, 1, 150) );

END;

/

--------------------------------------------------------

-- DDL for Procedure N2N\_OVERALL\_TEMP\_DEL\_ME

--------------------------------------------------------

set define off;

CREATE OR REPLACE PROCEDURE "XNG\_REPORTS"."N2N\_OVERALL\_TEMP\_DEL\_ME" IS

sqlStmt varchar2(200);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: n2n\_overall\_temp\_del\_me

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 1/31/2012 1. Created this procedure.

NOTES:

Automatically available Auto Replace Keywords:

Object Name: n2n\_overall\_temp\_del\_me

Sysdate: 1/31/2012

Date and Time: 1/31/2012, 10:06:20 AM, and 1/31/2012 10:06:20 AM

Username: (set in TOAD Options, Procedure Editor)

Table Name: (set in the "New PL/SQL Object" dialog)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BEGIN

sqlStmt := 'drop table N2N\_OVERALL\_SUMM\_PREV\_MONTH';

execute immediate sqlStmt;

sqlStmt := 'CREATE TABLE N2N\_OVERALL\_SUMM\_PREV\_MONTH as select \* from N2N\_OVERALL\_COMPLIANCE\_SUMM';

execute immediate sqlStmt;

--sqlStmt := 'alter table N2N\_OVERALL\_SUMM\_PREV\_MONTH add REGIONAL\_MONTHLY\_DELTA NUMBER';

--execute immediate sqlStmt;

--

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

WHEN OTHERS THEN

-- Consider logging the error and then re-raise

RAISE;

END n2n\_overall\_temp\_del\_me;

/

--------------------------------------------------------

-- DDL for Procedure TMP\_POPULATEBTPAUDITDATA

--------------------------------------------------------

set define off;

CREATE OR REPLACE PROCEDURE "XNG\_REPORTS"."TMP\_POPULATEBTPAUDITDATA" (area in varchar2) is

cursor getBTPData(ARG\_IN\_AREA varchar2) is

select bxa.xng\_area area, bxa.xng\_region region,

to\_char(sum((case

when bxa.btp\_xng\_match ='Y' --include to show multiple xng matches as errors "and bxa.uniquely\_identified = 'Y'"

then 1

else 0 end))

/count(\*)\*100,'999.99') as ckts\_matched\_pct

from XNG\_REPORTS.TEMP\_BTP\_XNG\_AUDIT bxa

where bxa.xng\_area is not null

and bxa.xng\_area=''||ARG\_IN\_AREA||''

group by rollup(bxa.xng\_area,bxa.xng\_region);

sql\_stmt varchar2(32000);

begin

for data in getBTPData(area)

loop

if(data.area is not null and data.region is not null)

then

sql\_stmt:= 'update xng\_reports.tmp\_executive\_summary es set es.ckt\_match\_pct='||data.ckts\_matched\_pct||' where es.area='''||data.area

||''' and es.region='''||data.region||''' and to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

--dbms\_output.put\_line(sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt;

commit;

elsif (data.area is not null and data.region is null)

then

sql\_stmt:= 'update xng\_reports.tmp\_executive\_summary es set es.ckt\_match\_pct='||data.ckts\_matched\_pct||' where es.area='''||data.area

||''' and es.region=''Totals'''||' and to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

EXECUTE IMMEDIATE sql\_stmt;

commit;

end if;

end loop;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in tmp\_populateBTPAuditData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end tmp\_populateBTPAuditData;

/

--------------------------------------------------------

-- DDL for Procedure TRUNCATE\_XNG\_REPORTS\_TABLE

--------------------------------------------------------

set define off;

CREATE OR REPLACE PROCEDURE "XNG\_REPORTS"."TRUNCATE\_XNG\_REPORTS\_TABLE" (p\_tablename in varchar2)

AS

--

-- Purpose: Allows XNG\_APPS to truncate tables in XNG\_REPORTS

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- Cathy Sayre 12/10/10

-- --------- ------ -------------------------------------------

sql\_str VARCHAR2(32767);

BEGIN

sql\_str :=

'truncate table '||p\_tablename||' reuse storage';

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

raise;

END;

/

--------------------------------------------------------

-- DDL for Package AUDIT\_REPORTS\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."AUDIT\_REPORTS\_PKG"

IS

--

-- Purpose: All the 'audit reports' will feed of off this central bussiness logic code.

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- ----------------- ----------------- ------------------------------------

-- Krishna Gadde Feb/March 2006 Initial version.

/\*

-- Audit reports ddl...Feb 2006

DROP TABLE xng\_reports.site\_summary

DROP TABLE xng\_reports.site\_audit

/

DROP TABLE xng\_reports.segment\_summary

/

DROP TABLE xng\_reports.segment\_audit

/

DROP TABLE xng\_reports.equipment\_summary

/

DROP TABLE xng\_reports.equipment\_audit

/

CREATE TABLE xng\_reports.site\_summary(

area VARCHAR2(50) NOT NULL,

region VARCHAR2(50) NOT NULL,

site\_count NUMBER NOT NULL,

extract\_date DATE NOT NULL)

/

CREATE UNIQUE INDEX xng\_reports.site\_summary\_pk ON xng\_reports.site\_summary (

area ASC,

region ASC )

/

CREATE TABLE xng\_reports.site\_audit(

area VARCHAR2(50) NOT NULL,

region VARCHAR2(50) NOT NULL,

attr\_name VARCHAR2(100) NOT NULL,

populated\_count NUMBER NOT NULL,

extract\_date DATE NOT NULL)

/

CREATE UNIQUE INDEX xng\_reports.site\_audit\_pk ON xng\_reports.site\_audit (

area ASC,

region ASC,

attr\_name ASC )

/

CREATE TABLE xng\_reports.segment\_summary(

area VARCHAR2(50) NOT NULL,

region VARCHAR2(50) NOT NULL,

paths NUMBER,

segments NUMBER,

segments\_w\_path NUMBER,

segments\_wo\_path NUMBER,

extract\_date DATE NOT NULL)

/

CREATE UNIQUE INDEX xng\_reports.segment\_summary\_pk ON xng\_reports.segment\_summary (

area ASC,

region ASC )

/

CREATE TABLE xng\_reports.segment\_audit(

area VARCHAR2(50) NOT NULL,

region VARCHAR2(50) NOT NULL,

btp\_segments NUMBER,

xng\_segment NUMBER,

btp\_mtc NUMBER,

xng\_mrc NUMBER,

paths NUMBER,

xng\_segs\_w\_path NUMBER,

xng\_segs\_wo\_path NUMBER,

extract\_date DATE NOT NULL)

/

CREATE UNIQUE INDEX xng\_reports.segment\_audit\_pk ON xng\_reports.segment\_audit (

area ASC,

region ASC )

/

CREATE TABLE xng\_reports.equipment\_summary(

area VARCHAR2(50) NOT NULL,

region VARCHAR2(50) NOT NULL,

vsm\_dacs\_count NUMBER NOT NULL,

vsm\_switch\_count NUMBER NOT NULL,

xng\_dacs\_match\_vs\_vsm NUMBER,

xng\_switch\_match\_vs\_vsm NUMBER,

extract\_date DATE NOT NULL)

/

CREATE UNIQUE INDEX xng\_reports.equipment\_summary\_pk ON xng\_reports.equipment\_summary (

area ASC,

region ASC )

/

CREATE TABLE xng\_reports.equipment\_audit(

area VARCHAR2(100) NOT NULL,

region VARCHAR2(500) NOT NULL,

device\_type VARCHAR2(30),

vsm\_class VARCHAR2(100),

vendor VARCHAR2(30),

xng\_device\_name VARCHAR2(100),

vsm\_device\_name VARCHAR2(100),

description VARCHAR2(40),

status VARCHAR2(20),

city VARCHAR2(100),

state VARCHAR2(40),

vsm\_city VARCHAR2(100),

vsm\_state VARCHAR2(2),

extract\_date DATE NOT NULL)

/

CREATE UNIQUE INDEX xng\_reports.equipment\_audit\_pk ON xng\_reports.equipment\_audit(

area ASC,

region ASC )

/

COMMIT;

-- Set the job....to run every hour

BEGIN

DECLARE

v\_JobNum NUMBER;

BEGIN

DBMS\_JOB.SUBMIT( v\_JobNum,

'AUDIT\_REPORTS\_PKG.loadAuditReportsData();',

sysdate,

'case when to\_char(sysdate, ''hh24'') between ''06'' and ''18'' then trunc(sysdate,''hh'')+1/24 when to\_char(SYSDATE, ''DAY'') = ''FRIDAY'' then trunc(sysdate+3)+6/24 else trunc(sysdate+1)+6/24 end',

TRUE );

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

END;

/

\*/

-- Starting point to load Audit report data fr Site, Ezuipment and Segment....

PROCEDURE loadAuditReportsData;

-- Get the region's audit data for the given area....

PROCEDURE deligate\_AuditReportsLoad( p\_area IN xng\_reports.vzw\_network\_org.area%TYPE );

-- Get Domain\_Inst\_Id for a given region

FUNCTION getDomainInstId( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;

-- Get site count for a given region

FUNCTION getSiteCountByRegion( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;

-- Save Region

PROCEDURE SaveRegion (p\_area IN xng\_reports.site\_summary.area%TYPE,

p\_region IN xng\_reports.site\_summary.region%TYPE);

-- Save Audit Summary table data ....

PROCEDURE SaveSiteSummary( p\_area IN xng\_reports.site\_summary.area%TYPE,

p\_region IN xng\_reports.site\_summary.region%TYPE,

p\_site\_count IN xng\_reports.site\_summary.site\_count%TYPE,

p\_mtso\_count IN xng\_reports.site\_summary.mtso\_count%TYPE );

-- Get Attribute count for a given region...

PROCEDURE getSiteAttrCount( p\_area IN xng\_reports.vzw\_network\_org.area%TYPE,

p\_region IN xng\_reports.vzw\_network\_org.region%TYPE );

-- Get Site Pots Number count for a given region...

PROCEDURE getSitePotsCount( p\_area IN xng\_reports.vzw\_network\_org.area%TYPE,

p\_region IN xng\_reports.vzw\_network\_org.region%TYPE );

-- Get required field populated count for an attribute in a region...

PROCEDURE getSiteReqFieldCount( p\_area IN xng\_reports.site\_summary.area%TYPE,

p\_region IN xng\_reports.site\_summary.region%TYPE,

p\_domain\_inst\_id IN vzwnet.domain\_inst.domain\_inst\_id%TYPE);

-- Save Site Audit table data ....

PROCEDURE SaveSiteAudit( p\_area IN xng\_reports.site\_audit.area%TYPE,

p\_region IN xng\_reports.site\_audit.region%TYPE,

p\_fieldName IN xng\_reports.site\_audit.attr\_name%TYPE,

p\_usage\_count IN xng\_reports.site\_audit.populated\_count%TYPE );

-- get BTP-Xng Scorecard data and call saveBTP\_XNG\_AUDIT\_SUMMARY

-- PROCEDURE getBTP\_XNG\_AUDIT\_SUMMARY(p\_AREA IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.AREA%TYPE );

-- Save data to BTP\_XNG\_AUDIT\_SUMMARY table...

/\* this was replaced by an update now included in xng\_apps.btp\_xng\_audit\_upd

PROCEDURE saveBTP\_XNG\_AUDIT\_SUMMARY

(p\_AREA IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.AREA%TYPE,

p\_REGION IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.REGION%TYPE,

p\_CKTS\_MATCHED IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CKTS\_MATCHED%TYPE,

p\_CKT\_COUNT IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CKT\_COUNT%TYPE,

p\_CHARGE\_AMT\_TOTAL IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CHARGE\_AMT\_TOTAL%TYPE,

p\_CHARGES\_MATCHED IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CHARGES\_MATCHED%TYPE,

p\_CKTS\_MATCHED\_PCT\_STR IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CKTS\_MATCHED\_PCT\_STR%TYPE,

p\_CKTS\_MATCHED\_PCT\_NUM IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CKTS\_MATCHED\_PCT\_NUM%TYPE,

p\_CHARGES\_MATCHED\_PCT\_STR IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CHARGES\_MATCHED\_PCT\_STR%TYPE,

p\_CHARGES\_MATCHED\_PCT\_NUM IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CHARGES\_MATCHED\_PCT\_NUM%TYPE);\*/

-- Get Equipment count for a given region...

/\* FUNCTION getVSM\_DACSEquipCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;

-- Get Equipment count for a given region...

FUNCTION getVSM\_SwitchEquipCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;

-- Get DACS Equipment Match count for a given region...

FUNCTION getXNG\_DACSEquipMatchCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;

-- Get Switch Equipment Match count for a given region...

FUNCTION getXNG\_SwitchEquipMatchCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;\*/

FUNCTION get\_audited\_mtso\_count(p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;

-- Get counts for compliant fields

FUNCTION getCompliantSiteCount(p\_name IN user\_tab\_columns.column\_name%TYPE,

p\_owner IN all\_tables.owner%TYPE,

p\_domain IN number)

RETURN NUMBER;

-- Save Equipment Audit data into table........

/\* PROCEDURE SaveEquipmentSummary( p\_area IN xng\_reports.equipment\_summary.area%TYPE,

p\_region IN xng\_reports.equipment\_summary.region%TYPE,

p\_vsm\_dacs\_count IN xng\_reports.equipment\_summary.vsm\_dacs\_count%TYPE,

p\_vsm\_switch\_count IN xng\_reports.equipment\_summary.vsm\_switch\_count%TYPE,

p\_xng\_dacs\_match\_vs\_vsm IN xng\_reports.equipment\_summary.xng\_dacs\_match\_vs\_vsm%TYPE,

p\_xng\_switch\_match\_vs\_vsm IN xng\_reports.equipment\_summary.xng\_switch\_match\_vs\_vsm%TYPE);

\*/

-- Segment ----------------

-- Get Path count...

FUNCTION getPathCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;

-- Segments count...

FUNCTION getSegmentCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE,

alltel\_mode IN varchar )

RETURN NUMBER;

-- Segments with paths...

FUNCTION getSegmentWithPathCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE,

alltel\_mode IN varchar )

RETURN NUMBER;

-- Segments with OUT paths...

FUNCTION getSegmentWithOutPathCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER;

-- Save Segment Summary table data ....

PROCEDURE SaveSegmentSummary( p\_area IN xng\_reports.site\_audit.area%TYPE,

p\_region IN xng\_reports.site\_audit.region%TYPE);

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package AUDIT\_STATUS\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."AUDIT\_STATUS\_PKG"

AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: AUDIT\_STATUS

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 04/15/2012 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE process\_audit\_status;

PROCEDURE delete\_audit\_status;

procedure update\_audit\_status;

END audit\_status\_pkg;

/

--------------------------------------------------------

-- DDL for Package CIRC\_SEG\_NEW\_DECOM\_RPT\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_RPT\_PKG"

AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: CIRC\_SEG\_NEW\_DECOM\_RPT\_PKG

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 7/15/2013 c0hedav 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE RUN\_DAILY\_DETAIL\_RPT;

PROCEDURE RUN\_ONE\_MONTHLY\_RPT (p\_run\_date DATE);

PROCEDURE RUN\_ONE\_WEEKLY\_RPT (p\_run\_date DATE);

PROCEDURE RUN\_ONE\_YEARLY\_RPT (p\_run\_date DATE);

PROCEDURE RUN\_ALL\_MONTHLY\_WEEKLY\_RPT;

END CIRC\_SEG\_NEW\_DECOM\_RPT\_PKG;

/

--------------------------------------------------------

-- DDL for Package COMPLETE\_EBH\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."COMPLETE\_EBH\_AUDIT"

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: COMPLETE\_EBH\_AUDIT

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/28/2010 1.Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE load\_ne\_vs\_Xng\_csr\_audit;

PROCEDURE update\_csr\_mis\_match;

PROCEDURE load\_csr\_missing\_in\_hpov;

PROCEDURE load\_gige\_paths\_for\_all\_csrs;

PROCEDURE load\_gige\_paths\_for\_a\_csr (csr\_equip\_inst\_id in number);

PROCEDURE load\_gige\_on\_native\_sonet (truncate\_tab in varchar2 default 'y');

PROCEDURE load\_gige\_on\_higher\_bw (truncate\_tab in varchar2 default 'y');

PROCEDURE load\_parents\_w\_seg\_per\_gige (gige\_path\_inst\_id in number, has\_vlan in boolean);

PROCEDURE load\_parents\_w\_microwave (gige\_path\_inst\_id in number, has\_vlan in boolean);

PROCEDURE load\_bts\_paths\_for\_all\_csrs;

PROCEDURE load\_bts\_paths\_for\_a\_csr (csr\_equip\_inst\_id in number);

PROCEDURE load\_gige\_vlans;

PROCEDURE get\_gige\_parents;

PROCEDURE get\_gige\_parents\_in\_1st\_leg (start\_path in number, current\_path in number, p\_sys\_conn\_by\_path in varchar2, deep in number);

PROCEDURE get\_vlan\_parents;

PROCEDURE get\_vlan\_parents\_in\_1st\_leg (start\_path in number, current\_path in number, p\_sys\_conn\_by\_path in varchar2, deep in number);

PROCEDURE load\_ebh\_summary\_by\_region;

PROCEDURE load\_csr\_summary\_by\_region;

PROCEDURE move\_csr\_data\_from\_wk;

PROCEDURE move\_csr\_data\_from\_prod\_to\_wk;

PROCEDURE qt\_audit;

PROCEDURE load\_qt\_summary\_by\_region;

PROCEDURE load\_vlan\_ngMLS\_map;

PROCEDURE load\_ngMLS\_devices;

PROCEDURE load\_mspp\_devices;

PROCEDURE load\_vlan\_mspp\_map;

PROCEDURE load\_vlan\_mspp\_tam;

PROCEDURE update\_ngMLS\_qt\_issues;

PROCEDURE update\_mspp\_qt\_issues;

PROCEDURE truncate\_csr\_wk;

RECURSIVE\_DEPTH number := 10;

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package DF\_STATUS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."DF\_STATUS"

IS

--

-- To modify this template, edit file PKGSPEC.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter package declarations as shown below

PROCEDURE do\_all;

PROCEDURE load\_df\_status\_details;

PROCEDURE load\_df\_status\_summary;

PROCESS\_NAME varchar2(30) := 'DF\_STATUS';

END;

/

--------------------------------------------------------

-- DDL for Package EBH\_DISCONNECT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."EBH\_DISCONNECT"

IS

--

-- To modify this template, edit file PKGSPEC.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter package declarations as shown below

PROCEDURE do\_all;

PROCEDURE load\_df\_seg\_w\_billing;

PROCEDURE load\_df\_seg\_wo\_billing;

PROCEDURE ebh\_disconnect\_summary;

PROCESS\_NAME varchar2(30) := 'EBH\_DISCONNECT';

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package EHEALTH\_TRAFFIC\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."EHEALTH\_TRAFFIC\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: ehealth\_traffic\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/9/2013 ramamla 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE runEhealthAudit;

END ehealth\_traffic\_audit;

/

--------------------------------------------------------

-- DDL for Package ENB\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."ENB\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: eNBAudit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/25/2010 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*FUNCTION MyFunction(Param1 IN NUMBER) RETURN NUMBER;\*/

AUDIT\_TABLE\_NAME varchar2(30) := 'NE\_VS\_XNG\_ENB\_AUDIT\_WRK';

SAM\_ENB varchar2(30) := 'SAM\_ENB';

OSS\_RC\_ENB varchar2(30) := 'oss\_rc\_enb';

XNG\_ENB\_PARSED varchar2(30) := 'xng\_parsed\_enb';

PROCEDURE eNbNeVsXngAudit;

/\* generate reports and save them for rendering adhoc reports\*/

procedure generateRegionalSummary;

procedure generateMarketIdSummary;

procedure copy\_enb\_wrk;

procedure truncate\_enb\_wrk;

procedure restore\_enb\_wrk;

END eNB\_Audit;

/

--------------------------------------------------------

-- DDL for Package ENB\_AUDIT\_WD

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."ENB\_AUDIT\_WD" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: eNBAudit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/25/2010 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*FUNCTION MyFunction(Param1 IN NUMBER) RETURN NUMBER;\*/

AUDIT\_TABLE\_NAME varchar2(30) := 'NE\_VS\_XNG\_ENB\_AUDIT\_WRK';

XMS\_ENB varchar2(30) := 'xms\_enb';

OSS\_RC\_ENB varchar2(30) := 'oss\_rc\_enb';

XNG\_ENB\_PARSED varchar2(30) := 'xng\_parsed\_enb';

PROCEDURE eNbNeVsXngAudit;

/\* generate reports and save them for rendering adhoc reports\*/

procedure generateRegionalSummary;

procedure generateMarketIdSummary;

procedure copy\_enb\_wrk;

procedure truncate\_enb\_wrk;

procedure restore\_enb\_wrk;

END eNB\_Audit\_wd;

/

--------------------------------------------------------

-- DDL for Package EXECUTIVE\_SUMMARY\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_PKG" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: TEST

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/10/2009 Latha Ramamurthy 1.Created this package.

1.1 02/10/2010 Jaya Chilakamarri 2. Modified computation for GrandTotals

1.2 03/04/2010 Jaya Chilakamarri 3. BTP-XNG Audit for All Including Alltel

1.3 05/27/2010 Jaya Chilakamarri 4. Made changes to fix rounding dicrepancy

1.4 07/09/2010 Jaya Chilakamarri 5. Modified date to to\_char to fox error

1.5 05/24/2012 Neeharika Peter 6. Removed Alltel Specific stuff,

added OBS,

introduced bind variables in several places

1.6 10/15/2012 Catherine Sayre 7. Added copyFromReportedtoPrevious

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

processName varchar2(100);

Procedure executiveSummaryReported;

Procedure executiveSummaryInProgress;

Procedure genExecutiveSummary;

Procedure createSnapshot;

Procedure createReportedSnapshot;

Procedure copyFromInProgressToReported;

Procedure copyFromWorkToInProgress;

Procedure createPreviousBackup;

Procedure copyFromReportedToPrevious;

Procedure deleteOldPrevious;

Procedure populateBTPAuditData;

Procedure populateXNGAuditData;

Procedure populateCrossConAuditData;

Procedure populateVSMData(area in varchar2);

Procedure populateSegmentData(area in varchar2);

Function getSegmentWithPathCount(p\_region IN varchar2) return number;

Function getSegmentCount(p\_region IN varchar2) return number;

--NSP added May 24, 2012, in preparation of adding OBS to ExecSumm

Procedure populateOBSData;

procedure populateOverallPct;

procedure populateMetricTotals;

procedure populatePrevAndDeltasCol;

procedure populateDeltasRow;

Procedure populatePrevGrandTotalsRow;

END Executive\_Summary\_Pkg;

/

--------------------------------------------------------

-- DDL for Package EXECUTIVE\_SUMMARY\_PKG\_NEW

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_PKG\_NEW" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: TEST

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/10/2009 Latha Ramamurthy 1.Created this package.

1.1 02/10/2010 Jaya Chilakamarri 2. Modified computation for GrandTotals

1.2 03/04/2010 Jaya Chilakamarri 3. BTP-XNG Audit for All Including Alltel

1.3 05/27/2010 Jaya Chilakamarri 4. Made changes to fix rounding dicrepancy

1.4 07/09/2010 Jaya Chilakamarri 5. Modified date to to\_char to fox error

1.5 05/24/2012 Neeharika Peter 6. Removed Alltel Specific stuff,

added OBS,

introduced bind variables in several places

1.6 10/15/2012 Catherine Sayre 7. Added copyFromReportedtoPrevious

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

processName varchar2(100);

Procedure executiveSummaryReported;

Procedure executiveSummaryInProgress;

Procedure genExecutiveSummary;

Procedure createSnapshot;

Procedure createReportedSnapshot;

Procedure copyFromInProgressToReported;

Procedure copyFromWorkToInProgress;

Procedure createPreviousBackup;

Procedure copyFromReportedToPrevious;

Procedure deleteOldPrevious;

Procedure populateBTPAuditData;

Procedure populateXNGAuditData;

Procedure populateCrossConAuditData;

Procedure populateVSMData(territory in varchar2);

Procedure populateSegmentData(territory in varchar2);

Function getSegmentWithPathCount(p\_sub\_mkt IN varchar2) return number;

Function getSegmentCount(p\_sub\_mkt IN varchar2) return number;

--NSP added May 24, 2012, in preparation of adding OBS to ExecSumm

Procedure populateOBSData;

procedure populateOverallPct;

procedure populateMetricTotals;

procedure populatePrevAndDeltasCol;

procedure populateDeltasRow;

Procedure populatePrevGrandTotalsRow;

END Executive\_Summary\_Pkg\_New;

/

--------------------------------------------------------

-- DDL for Package GEOPLAN\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."GEOPLAN\_PKG" as

PROCEDURE backup\_geoplan\_discovery;

PROCEDURE backup\_geoplan\_sc\_discovery;

end GEOPLAN\_PKG;

/

--------------------------------------------------------

-- DDL for Package HPOV\_NCM\_EH\_GI\_CSR\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."HPOV\_NCM\_EH\_GI\_CSR\_AUDIT"

IS

processName varchar2(100) := 'HPOV\_NCM\_GI\_CSR\_AUDIT';

PROCEDURE load\_ne\_vs\_Xng\_csr\_audit;

PROCEDURE insert\_ci\_csr\_equip\_issues (processname VARCHAR2);

PROCEDURE insert\_csr\_issues (processname VARCHAR2, methodname VARCHAR2, insertSql VARCHAR2, errorMsg VARCHAR2);

PROCEDURE load\_csr\_summary\_by\_clli;

PROCEDURE load\_csr\_summary\_by\_region;

procedure move\_csr\_data\_from\_prod\_to\_wk;

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package JUNIPER\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."JUNIPER\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: juniper\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/10/2014 SME Cloned from NetSmart Audit.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE GETGRANITEEQUIP;

PROCEDURE AUDITEQUIP;

PROCEDURE GENSUMMARY;

PROCEDURE RUNAUDIT;

PROCEDURE TRUNCATE\_WK;

PROCEDURE MOVE\_DATA\_FROM\_WK;

PROCEDURE MOVE\_DATA\_TO\_WK;

FUNCTION SUMMARY\_TBL RETURN JUNIPER\_SUMMARY\_TBL PIPELINED;

FUNCTION DETAIL\_TBL(REGION\_IN IN VARCHAR2, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN JUNIPER\_DETAIL\_TBL PIPELINED;

-- This is used to build the realized Summary Report table

FUNCTION DETAIL\_TBL\_WRK(START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN JUNIPER\_DETAIL\_TBL PIPELINED;

END JUNIPER\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package LDAP\_XNG\_USER\_FIX

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."LDAP\_XNG\_USER\_FIX" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: LDAP\_XNG\_USER\_FIX

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 6/1/2011 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE find\_uswin\_ids;

PROCEDURE update\_xng\_tables;

END LDAP\_XNG\_USER\_FIX;

/

--------------------------------------------------------

-- DDL for Package LEASED\_SEG\_DF

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."LEASED\_SEG\_DF"

IS

--

-- To modify this template, edit file PKGSPEC.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter package declarations as shown below

PROCEDURE do\_all;

PROCEDURE load\_LEASED\_SEG\_DF\_details;

PROCEDURE load\_LEASED\_SEG\_DF\_summary;

PROCESS\_NAME varchar2(30) := 'LEASED\_SEG\_DF';

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package LIVE\_OBJECTS\_NONLIVE\_PATHS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."LIVE\_OBJECTS\_NONLIVE\_PATHS" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: LIVE\_OBJECTS\_NONLIVE\_PATHS

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 6/03/2013 Jaya Chilakamarri 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE insert\_live\_object\_data;

PROCEDURE insert\_live\_object\_data\_rollup;

PROCEDURE do\_all;

END LIVE\_OBJECTS\_NONLIVE\_PATHS;

/

--------------------------------------------------------

-- DDL for Package LIVE\_OBJS\_NL\_SITES

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."LIVE\_OBJS\_NL\_SITES" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: LIVE\_OBJ\_NL\_SITES

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/17/2014 Gautam 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

table\_name VARCHAR2 (100) := 'XNG\_REPORTS.LIVE\_OBJ\_NL\_SITES\_SUMM\_WK';

processName VARCHAR2 (100) := 'LIVE\_OBJ\_NL\_SITES\_AUDIT';

PROCEDURE insert\_objs;

END LIVE\_OBJS\_NL\_SITES;

/

--------------------------------------------------------

-- DDL for Package LIVE\_OBJ\_NL\_SITES\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: LIVE\_OBJ\_NL\_SITES\_AUDIT

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 12/15/2014 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE insert\_live\_obj\_nl\_sites;

END LIVE\_OBJ\_NL\_SITES\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package LUCENT\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."LUCENT\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: lucent\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 2/14/2011 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

AUDIT\_TBL varchar2(30) := 'LUCENT\_NE\_VS\_XNG\_AUDIT\_WRK';

-- this is wrapper that calls the bottom 2 procs. done this way for backward compatibility

Procedure auditXngVsProspect;

PROCEDURE auditLucentBTSTerminations;

PROCEDURE generateLucentRegSummary;

PROCEDURE generateLucentSwitchSummary;

PROCEDURE copy\_lucent\_wrk;

PROCEDURE truncate\_lucent\_wrk;

PROCEDURE restore\_lucent\_wrk;

END lucent\_audit;

/

--------------------------------------------------------

-- DDL for Package MOTOROLA\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."MOTOROLA\_AUDIT" IS

-- this is wrapper that calls the bottom 2 procs. done this way for backward compatibility

PROCEDURE moto\_cdma\_audit;

PROCEDURE moto\_cdma\_regional\_summary;

PROCEDURE MOTO\_CDMA\_DEVICE\_SUMMARY;

PROCEDURE CREATE\_UBS\_AUDIT;

PROCEDURE moto\_evdo\_audit;

PROCEDURE moto\_evdo\_regional\_summary;

PROCEDURE MOTO\_EVDO\_DEVICE\_SUMMARY;

FUNCTION SET\_MESSAGE (fullMsg IN VARCHAR2, message IN VARCHAR2) RETURN VARCHAR2;

FUNCTION GET\_UBS\_MATCH\_STATUS (DS1\_NUMBER IN NUMBER, BANDWIDTH IN VARCHAR, BTS\_TYPE IN VARCHAR, IST1 IN INTEGER,

ISUBS IN INTEGER, PARSESTATS IN VARCHAR) RETURN VARCHAR2;

T1\_TYPE INTEGER := 1;

EBH\_TYPE INTEGER := 0;

UBS\_TYPE INTEGER := 1;

NON\_UBS\_TYPE INTEGER := 0;

CDMA\_DEVICE\_SUMM VARCHAR2(30) := 'MOTO\_CDMA\_AUDIT\_DEV\_SUMM\_WRK';

CDMA\_REGION\_SUMM VARCHAR2(30) := 'MOTO\_CDMA\_AUDIT\_REG\_SUMM\_WRK';

PROCEDURE copy\_moto\_cdma\_wrk;

PROCEDURE copy\_moto\_evdo\_wrk;

PROCEDURE truncate\_moto\_cdma\_wrk;

PROCEDURE truncate\_moto\_evdo\_wrk;

PROCEDURE restore\_moto\_cdma\_wrk;

PROCEDURE restore\_moto\_evdo\_wrk;

END MOTOROLA\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package MPLS\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."MPLS\_AUDIT"

IS

--

-- To modify this template, edit file PKGSPEC.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter package declarations as shown below

MPLS\_SUMMARY\_BY\_DEVICE varchar2(50) := 'MPLS\_SUMMARY\_BY\_DEVICE';

MPLS\_SUMMARY\_BY\_REGION varchar2(50) := 'MPLS\_SUMMARY\_BY\_REGION';

MPLS\_SUMMARY\_BY\_DEVICE\_WK varchar2(50) := 'MPLS\_SUMMARY\_BY\_DEVICE\_WK';

MPLS\_SUMMARY\_BY\_REGION\_WK varchar2(50) := 'MPLS\_SUMMARY\_BY\_REGION\_WK';

MPLS\_NNO\_SUMMARY\_BY\_DEVICE varchar2(50) := 'MPLS\_NNO\_SUMMARY\_BY\_DEVICE';

MPLS\_NNO\_SUMMARY\_BY\_REGION varchar2(50) := 'MPLS\_NNO\_SUMMARY\_BY\_REGION';

MPLS\_NNO\_SUMMARY\_BY\_DEVICE\_WK varchar2(50) := 'MPLS\_NNO\_SUMMARY\_BY\_DEVICE\_WK';

MPLS\_NNO\_SUMMARY\_BY\_REGION\_WK varchar2(50) := 'MPLS\_NNO\_SUMMARY\_BY\_REGION\_WK';

PROCEDURE copy\_mpls\_data\_from\_wk;

PROCEDURE copy\_mpls\_data\_from\_prod\_to\_wk;

END;

/

--------------------------------------------------------

-- DDL for Package MSPP\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."MSPP\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: MSPP\_AUDIT

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 7/7/2011 Satya Modugula 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

INC\_MSPP\_XCONN\_AUDIT\_DETAILS varchar2(50) := 'INC\_MSPP\_XCONN\_AUDIT\_DETAILS';

INC\_MSPP\_XCONN\_AUDIT\_SUMMARY varchar2(50) := 'INC\_MSPP\_XCONN\_AUDIT\_SUMMARY';

MSPP\_CIRC\_PATH\_ELEMENTS varchar2(50) := 'MSPP\_CIRC\_PATH\_ELEMENTS';

MSPP\_SUMMARY\_REGION\_MAP varchar2(50) := 'MSPP\_SUMMARY\_REGION\_MAP';

MSPP\_VSM\_MATCH varchar2(50) := 'MSPP\_VSM\_MATCH';

INC\_MSPP\_ADM\_COMP\_SUMMARY varchar2(50) := 'INC\_MSPP\_ADM\_COMP\_SUMMARY';

INC\_MSPP\_COMPLIANCE\_SUMMARY varchar2(50) := 'INC\_MSPP\_COMPLIANCE\_SUMMARY';

INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK varchar2(50) := 'INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK';

INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK varchar2(50) := 'INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK';

MSPP\_CIRC\_PATH\_ELEMENTS\_WK varchar2(50) := 'MSPP\_CIRC\_PATH\_ELEMENTS\_WK';

MSPP\_SUMMARY\_REGION\_MAP\_WK varchar2(50) := 'MSPP\_SUMMARY\_REGION\_MAP\_WK';

MSPP\_VSM\_MATCH\_WK varchar2(50) := 'MSPP\_VSM\_MATCH\_WK';

INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK varchar2(50) := 'INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK';

INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK varchar2(50) := 'INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK';

PROCEDURE copy\_mspp\_data\_from\_wk\_to\_prod;

PROCEDURE copy\_mspp\_data\_from\_prod\_to\_wk;

PROCEDURE truncate\_mspp\_wk;

END MSPP\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package MWR\_DISCOVERY

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."MWR\_DISCOVERY" as

PROCEDURE load\_xng\_mwr\_paths;

end mwr\_discovery;

/

--------------------------------------------------------

-- DDL for Package N2N\_OVERALL\_COMPLIANCE

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."N2N\_OVERALL\_COMPLIANCE" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: n2n\_overall\_compliance

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 1/27/2012 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE takeMonthlySnapshot;

END n2n\_overall\_compliance;

/

--------------------------------------------------------

-- DDL for Package NCM\_CSR\_VLAN\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDIT"

AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: SAM\_EBH\_NGMLS\_AUDIT

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/19/2012 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure audit\_CI\_csr\_vlans ;

procedure insert\_CI\_CSR\_vlan\_issues;

PROCEDURE assign\_area\_region\_to\_csr\_vlan;

PROCEDURE load\_vlan\_csr\_clli\_summary;

PROCEDURE load\_csr\_vlan\_regional\_summ;

END NCM\_CSR\_VLAN\_audit;

/

--------------------------------------------------------

-- DDL for Package NCM\_GI\_NGMLS\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."NCM\_GI\_NGMLS\_AUDIT"

IS

--

-- To modify this template, edit file PKGSPEC.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

PROCEDURE audit\_ngMLS\_equip;

PROCEDURE audit\_CI\_ngMLS\_equip;

PROCEDURE insert\_CI\_ngMLS\_equip\_issues;

PROCEDURE audit\_ngMLS\_vlans;

PROCEDURE audit\_CI\_ngMLS\_vlans;

PROCEDURE insert\_CI\_ngMLS\_vlan\_issues;

PROCEDURE assign\_area\_region\_to\_devices;

PROCEDURE load\_ngMLS\_regional\_summary;

PROCEDURE load\_vlan\_regional\_summary;

PROCEDURE load\_vlan\_device\_summary;

PROCEDURE load\_CI\_vlan\_device\_summary;

PROCEDURE move\_data\_from\_prod\_to\_wk;

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package NETSMART\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."NETSMART\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: netsmart\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 5/2/2012 NSP Created this package.

1.1 5/11/2012 SME Realized detail query and

added report table functions

1.2 3/9/2015 SME Split audit into MSPP and ADM

methods.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--

-- MSPP Methods

--

PROCEDURE GETGRANITEEQUIP;

PROCEDURE AUDITFWEQUIP;

PROCEDURE GENSUMMARY;

PROCEDURE AUDIT\_XCONNECTS;

FUNCTION NS\_SUMMARY\_TBL RETURN NETSMART\_SUMMARY\_TBL PIPELINED;

FUNCTION NS\_DETAIL\_TBL(REGION\_IN IN VARCHAR2, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_DETAIL\_TBL PIPELINED;

FUNCTION NS\_XCONNECT\_TBL (TID\_IN IN VARCHAR2, ONLY\_DISCREPANCIES\_IN IN NUMBER, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_XCONNECT\_TBL PIPELINED;

FUNCTION NS\_XCONNECT\_SUMMARY (REGION\_IN IN VARCHAR2, ONLY\_DISCREPANCIES\_IN IN NUMBER, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_XCONNECT\_TBL PIPELINED;

/\* This is used to build the realized Summary Report table \*/

FUNCTION NS\_DETAIL\_TBL\_WRK(START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_DETAIL\_TBL PIPELINED;

--

-- ADM Methods

--

PROCEDURE GETADMGRANITEEQUIP;

PROCEDURE AUDITADMFWEQUIP;

PROCEDURE GENADMSUMMARY;

PROCEDURE AUDIT\_ADM\_XCONNECTS;

FUNCTION NS\_ADM\_SUMMARY\_TBL RETURN NETSMART\_SUMMARY\_TBL PIPELINED;

FUNCTION NS\_ADM\_DETAIL\_TBL(REGION\_IN IN VARCHAR2, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_DETAIL\_TBL PIPELINED;

FUNCTION NS\_ADM\_XCONNECT\_TBL (TID\_IN IN VARCHAR2, ONLY\_DISCREPANCIES\_IN IN NUMBER, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_XCONNECT\_TBL PIPELINED;

FUNCTION NS\_ADM\_XCONNECT\_SUMMARY (REGION\_IN IN VARCHAR2, ONLY\_DISCREPANCIES\_IN IN NUMBER, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_XCONNECT\_TBL PIPELINED;

/\* This is used to build the realized Summary Report table \*/

FUNCTION NS\_ADM\_DETAIL\_TBL\_WRK(START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_DETAIL\_TBL PIPELINED;

--

-- Both MSPP and ADM

--

PROCEDURE RUNAUDIT;

PROCEDURE TRUNCATE\_WK;

PROCEDURE MOVE\_DATA\_FROM\_WK;

PROCEDURE MOVE\_DATA\_TO\_WK;

END NETSMART\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package NL\_OBJS\_IN\_LIVE\_PATHS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: NL\_OBJS\_IN\_LIVE\_PATHS

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/13/2014 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

table\_name VARCHAR2 (100) := 'XNG\_REPORTS.NL\_OBJ\_IN\_LIVE\_PATHS\_WK';

processName VARCHAR2 (100) := 'NON\_LIVE\_OBJECTS\_LIVE\_PATHS\_AUDIT';

summary\_table\_name VARCHAR2 (100) := 'XNG\_REPORTS.NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK';

PROCEDURE insert\_nl\_objs;

PROCEDURE insert\_nl\_objs\_summary;

PROCEDURE do\_all;

END NL\_OBJS\_IN\_LIVE\_PATHS;

/

--------------------------------------------------------

-- DDL for Package NONPRINTABLE\_CLEANUP\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."NONPRINTABLE\_CLEANUP\_PKG" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: TEST

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 12/15/2015 Catherine Sayre Created

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

processName varchar2(100);

Procedure runCleanup;

Procedure nonprintableCleanup;

END Nonprintable\_Cleanup\_Pkg;

/

--------------------------------------------------------

-- DDL for Package NORTEL\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."NORTEL\_AUDIT"

IS

-- Purpose: To create nortel audit tables for voice spans

NT\_DOM\_INV varchar2(30) := 'NORTEL\_DOM\_INVENTORY';

NT\_EVDO\_PORT\_INV varchar2(30) := 'NORTEL\_PORT\_INVENTORY';

XNG\_NT\_PARSED\_CELL\_PATHS varchar2(30) := 'VZW\_NT\_EVDO\_CELL\_PATHS';

NORTEL\_NE\_VS\_XNG\_AUDIT varchar2(30) := 'NORTEL\_NE\_VS\_XNG\_AUDIT';

PROCEDURE nt\_cdma\_audit;

PROCEDURE nt\_evdo\_audit;

NORTEL\_CDMA\_AUDIT\_REG\_SUMM varchar2(30) := 'NORTEL\_CDMA\_AUDIT\_REG\_SUMM';

NORTEL\_CDMA\_AUDIT\_DEV\_SUMM varchar2(30) := 'NORTEL\_CDMA\_AUDIT\_DEV\_SUMM';

NT\_EVDO\_BTS\_TERM\_REG\_SUMM varchar2(30) := 'NT\_EVDO\_BTS\_TERM\_REG\_SUMM';

NT\_EVDO\_BTS\_TERM\_EMS\_SUMM varchar2(30) := 'NT\_EVDO\_BTS\_TERM\_EMS\_SUMM';

PROCEDURE populate\_cdma\_regional\_summary;

PROCEDURE populate\_cdma\_device\_summary;

PROCEDURE populate\_evdo\_regional\_summary;

PROCEDURE populate\_evdo\_device\_summary;

PROCEDURE copy\_nortel\_cdma\_wrk;

PROCEDURE copy\_nortel\_evdo\_wrk;

PROCEDURE run\_cdma\_audit\_wrk;

PROCEDURE run\_evdo\_audit\_wrk;

PROCEDURE truncate\_nortel\_cdma\_wrk;

PROCEDURE truncate\_nortel\_evdo\_wrk;

PROCEDURE restore\_nortel\_cdma\_wrk;

PROCEDURE restore\_nortel\_evdo\_wrk;

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package RPT\_IAN\_NOCC\_SITE\_TEST

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."RPT\_IAN\_NOCC\_SITE\_TEST"

IS

--

-- Purpose: NOCC SITE ALARAMS AGGREAGATE SUMMARY and DETAIL reports.

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- ---------------------- ------------ -----------------------------------------

-- Krishna Gadde Oct 2005 Initial release.

-- Krihsna Gadde March 2006 Tunned...

--/\* ddl

--CREATE TABLE xng\_reports.nocc\_last\_test\_summary

-- (area VARCHAR2(40) NOT NULL,

-- region VARCHAR2(80) NOT NULL,

-- domain\_id VARCHAR2(80) NOT NULL,

-- domain\_name VARCHAR2(100) NOT NULL,

-- sites NUMBER(9,0),

-- antennas NUMBER(9,0),

-- vzw\_owned NUMBER(9,0),

-- lights\_reqd NUMBER(9,0),

-- lights\_tested NUMBER(9,0),

-- lights\_test\_overdue NUMBER(9,0),

-- lights\_test\_ok NUMBER(9,0),

-- nocc\_monitored\_sites NUMBER(9,0),

-- env\_tested NUMBER(9,0),

-- env\_test\_overdue NUMBER(9,0),

-- env\_test\_ok NUMBER(9,0),

-- run\_date DATE NOT NULL)

--/

--

--CREATE INDEX xng\_reports.indx\_last\_test\_summary ON xng\_reports.nocc\_last\_test\_summary

-- (

-- "RUN\_DATE" DESC,

-- domain\_id ASC

-- )

--/

--

--

--CREATE TABLE xng\_reports.nocc\_last\_test\_details

-- (domain\_id NUMBER(9,0) NOT NULL,

-- domain\_name VARCHAR2(50) NOT NULL,

-- site\_inst\_id NUMBER(9,0) NOT NULL,

-- site\_name VARCHAR2(100) NOT NULL,

-- site\_id VARCHAR2(30),

-- site\_tech\_name VARCHAR2(100),

-- inventoried\_cells VARCHAR2(1000),

-- site\_type VARCHAR2(30),

-- nocc\_monitored VARCHAR2(30),

-- light\_last\_tested VARCHAR2(30),

-- light\_next\_test VARCHAR2(30),

-- light\_test\_overdue VARCHAR2(30),

-- env\_last\_tested VARCHAR2(30),

-- env\_next\_test VARCHAR2(30),

-- env\_test\_overdue VARCHAR2(30),

-- run\_date DATE)

--/

--

--CREATE INDEX xng\_reports.indx\_last\_test\_details ON xng\_reports.nocc\_last\_test\_details

-- (

-- "RUN\_DATE" DESC,

-- domain\_id ASC,

-- site\_inst\_id ASC

-- )

--/

--

---- Schedule an Oracle job to run everyday... every hour from 6am to 6pm in weekdays.

--

--BEGIN

-- DECLARE

-- v\_JobNum NUMBER;

-- BEGIN

-- DBMS\_JOB.SUBMIT (v\_JobNum,

-- 'RPT\_NOCC\_SITE\_TEST.getAllDomains();',

-- sysdate,

-- 'case when to\_char(sysdate, ''hh24'') between ''06'' and ''18'' then trunc(sysdate,''hh'')+1/24 when to\_char(SYSDATE, ''DAY'') = ''FRIDAY'' then trunc(sysdate+3)+6/24 else trunc(sysdate+1)+6/24 end',

-- TRUE );

-- commit;

--

-- EXCEPTION

-- WHEN OTHERS THEN

-- dbms\_output.put\_line(SubStr('Error '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

-- RAISE;

-- END;

-- END;

--

--

--

--\*/

PROCEDURE getAllDomains; -- This will load all the test data. Starting point.

PROCEDURE getAllSites (

p\_DomainId IN vzwnet.Domain\_Inst.Domain\_Inst\_Id%TYPE,

p\_DomainName IN vzwnet.Domain\_Inst.Domain\_Name%TYPE,

p\_Area IN xng\_reports.VZW\_NETWORK\_ORG.Area%TYPE,

p\_Region IN xng\_reports.VZW\_NETWORK\_ORG.Region%TYPE);

FUNCTION getSiteTechName (

p\_SiteInstId IN vzwnet.site\_attr\_settings.site\_inst\_id%TYPE )

RETURN vzwnet.site\_attr\_settings.attr\_value%TYPE;

FUNCTION getCellEquipDesc(

p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE )

RETURN NOCC\_LAST\_TEST\_DETAILS.INVENTORIED\_CELLS%TYPE;

FUNCTION getIsNoccMonitored (

p\_SiteInstId IN vzwnet.site\_inst.site\_inst\_id%TYPE )

RETURN VARCHAR2;

FUNCTION getFailedFirstEnvAttempt (

p\_SiteInstId IN vzwnet.site\_inst.site\_inst\_id%TYPE )

RETURN VARCHAR2;

FUNCTION getENVLastTestedDate (

p\_SiteInstId IN vzwnet.site\_attr\_settings.site\_inst\_id%TYPE,

p\_SiteType IN vzwnet.val\_attr\_group.class\_category%TYPE )

RETURN vzwnet.site\_attr\_settings.attr\_value%TYPE;

FUNCTION getHasAntennaTower(

p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE )

RETURN VARCHAR2;

FUNCTION getIsAntennaVZWOwned (

p\_SiteInstId IN vzwnet.equip\_inst.site\_inst\_id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE )

RETURN vzwnet.equip\_attr\_settings.attr\_value%TYPE;

FUNCTION getisLightingRequired (

p\_SiteInstId IN vzwnet.equip\_inst.site\_inst\_id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE )

RETURN vzwnet.equip\_attr\_settings.attr\_value%TYPE;

FUNCTION getTowerLightLastTestedDate (

p\_SiteInstId IN vzwnet.site\_attr\_settings.site\_inst\_id%TYPE,

p\_SiteType IN vzwnet.val\_attr\_group.class\_category%TYPE )

RETURN vzwnet.site\_attr\_settings.attr\_value%TYPE;

PROCEDURE insert\_NOCC\_LAST\_TEST\_SUMMARY(

p\_Area IN NOCC\_LAST\_TEST\_SUMMARY.Area%TYPE,

p\_Region IN NOCC\_LAST\_TEST\_SUMMARY.Region%TYPE,

p\_DomainId IN NOCC\_LAST\_TEST\_SUMMARY.domain\_id%TYPE,

p\_DomainName IN NOCC\_LAST\_TEST\_SUMMARY.domain\_name%TYPE,

p\_Sites IN NOCC\_LAST\_TEST\_SUMMARY.sites%TYPE,

p\_Antennas IN NOCC\_LAST\_TEST\_SUMMARY.antennas%TYPE,

p\_VZWOwned IN NOCC\_LAST\_TEST\_SUMMARY.vzw\_owned%TYPE,

p\_LightsReqd IN NOCC\_LAST\_TEST\_SUMMARY.lights\_reqd%TYPE,

p\_LightsTested IN NOCC\_LAST\_TEST\_SUMMARY.lights\_tested%TYPE,

p\_LightsOverDue IN NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_overdue%TYPE,

p\_LightsOk IN NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_ok%TYPE,

p\_Monitored IN NOCC\_LAST\_TEST\_SUMMARY.NOCC\_monitored\_sites%TYPE,

p\_EnvTested IN NOCC\_LAST\_TEST\_SUMMARY.env\_tested%TYPE,

p\_EnvOverDue IN NOCC\_LAST\_TEST\_SUMMARY.env\_test\_overdue%TYPE,

p\_EnvOK IN NOCC\_LAST\_TEST\_SUMMARY.env\_test\_ok%TYPE);

PROCEDURE insert\_NOCC\_Last\_Test\_Details(

p\_DomainId IN NOCC\_LAST\_TEST\_DETAILS.domain\_Id%TYPE,

p\_DomainName IN NOCC\_LAST\_TEST\_DETAILS.domain\_name%TYPE,

p\_SiteInstId IN NOCC\_LAST\_TEST\_DETAILS.site\_Inst\_Id%TYPE,

p\_SiteName IN NOCC\_LAST\_TEST\_DETAILS.site\_name%TYPE,

p\_SiteId IN NOCC\_LAST\_TEST\_DETAILS.site\_id%TYPE,

p\_Sitetype IN NOCC\_LAST\_TEST\_DETAILS.site\_type%TYPE,

p\_SiteTechName IN NOCC\_LAST\_TEST\_DETAILS.site\_tech\_name%TYPE,

p\_inventoried\_cells IN NOCC\_LAST\_TEST\_DETAILS.inventoried\_cells%TYPE,

p\_NOCC\_Monitored IN NOCC\_LAST\_TEST\_DETAILS.nocc\_Monitored%TYPE,

p\_vzw\_owned IN NOCC\_LAST\_TEST\_DETAILS.vzw\_owned%TYPE,

p\_lights\_reqd IN NOCC\_LAST\_TEST\_DETAILS.lights\_reqd%TYPE,

p\_LightLastTested IN NOCC\_LAST\_TEST\_DETAILS.Light\_last\_tested%TYPE,

p\_LightNextTest IN NOCC\_LAST\_TEST\_DETAILS.Light\_next\_test%TYPE,

p\_LightTestOverDue IN NOCC\_LAST\_TEST\_DETAILS.light\_test\_overdue%TYPE,

p\_ENVLastTested IN NOCC\_LAST\_TEST\_DETAILS.env\_last\_tested%TYPE,

p\_ENVNextTest IN NOCC\_LAST\_TEST\_DETAILS.env\_next\_test%TYPE,

p\_ENVTestOverDue IN NOCC\_LAST\_TEST\_DETAILS.env\_test\_overdue%TYPE,

p\_EquipInstid IN NOCC\_LAST\_TEST\_DETAILS.equip\_inst\_id%TYPE );

FUNCTION varchar2date( p\_StrDate IN VARCHAR2 ) RETURN DATE;

PROCEDURE removeDetails( p\_DomainId IN NOCC\_LAST\_TEST\_SUMMARY.domain\_id%TYPE );

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package RPT\_STD\_NOCC\_SITE\_TEST

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."RPT\_STD\_NOCC\_SITE\_TEST"

IS

--

-- Purpose: NOCC SITE ALARAMS AGGREAGATE SUMMARY and DETAIL reports.

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- ---------------------- ------------ -----------------------------------------

-- Krishna Gadde Oct 2005 Initial release.

-- Krihsna Gadde March 2006 Tunned...

--/\* ddl

--CREATE TABLE xng\_reports.nocc\_last\_test\_summary

-- (area VARCHAR2(40) NOT NULL,

-- region VARCHAR2(80) NOT NULL,

-- domain\_id VARCHAR2(80) NOT NULL,

-- domain\_name VARCHAR2(100) NOT NULL,

-- sites NUMBER(9,0),

-- antennas NUMBER(9,0),

-- vzw\_owned NUMBER(9,0),

-- lights\_reqd NUMBER(9,0),

-- lights\_tested NUMBER(9,0),

-- lights\_test\_overdue NUMBER(9,0),

-- lights\_test\_ok NUMBER(9,0),

-- nocc\_monitored\_sites NUMBER(9,0),

-- env\_tested NUMBER(9,0),

-- env\_test\_overdue NUMBER(9,0),

-- env\_test\_ok NUMBER(9,0),

-- run\_date DATE NOT NULL)

--/

--

--CREATE INDEX xng\_reports.indx\_last\_test\_summary ON xng\_reports.nocc\_last\_test\_summary

-- (

-- "RUN\_DATE" DESC,

-- domain\_id ASC

-- )

--/

--

--

--CREATE TABLE xng\_reports.nocc\_last\_test\_details

-- (domain\_id NUMBER(9,0) NOT NULL,

-- domain\_name VARCHAR2(50) NOT NULL,

-- site\_inst\_id NUMBER(9,0) NOT NULL,

-- site\_name VARCHAR2(100) NOT NULL,

-- site\_id VARCHAR2(30),

-- site\_tech\_name VARCHAR2(100),

-- inventoried\_cells VARCHAR2(1000),

-- site\_type VARCHAR2(30),

-- nocc\_monitored VARCHAR2(30),

-- light\_last\_tested VARCHAR2(30),

-- light\_next\_test VARCHAR2(30),

-- light\_test\_overdue VARCHAR2(30),

-- env\_last\_tested VARCHAR2(30),

-- env\_next\_test VARCHAR2(30),

-- env\_test\_overdue VARCHAR2(30),

-- run\_date DATE)

--/

--

--CREATE INDEX xng\_reports.indx\_last\_test\_details ON xng\_reports.nocc\_last\_test\_details

-- (

-- "RUN\_DATE" DESC,

-- domain\_id ASC,

-- site\_inst\_id ASC

-- )

--/

--

---- Schedule an Oracle job to run everyday... every hour from 6am to 6pm in weekdays.

--

--BEGIN

-- DECLARE

-- v\_JobNum NUMBER;

-- BEGIN

-- DBMS\_JOB.SUBMIT (v\_JobNum,

-- 'RPT\_NOCC\_SITE\_TEST.getAllDomains();',

-- sysdate,

-- 'case when to\_char(sysdate, ''hh24'') between ''06'' and ''18'' then trunc(sysdate,''hh'')+1/24 when to\_char(SYSDATE, ''DAY'') = ''FRIDAY'' then trunc(sysdate+3)+6/24 else trunc(sysdate+1)+6/24 end',

-- TRUE );

-- commit;

--

-- EXCEPTION

-- WHEN OTHERS THEN

-- dbms\_output.put\_line(SubStr('Error '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

-- RAISE;

-- END;

-- END;

--

--

--

--\*/

PROCEDURE getAllDomains; -- This will load all the test data. Starting point.

PROCEDURE getAllSites (

p\_DomainId IN vzwnet.Domain\_Inst.Domain\_Inst\_Id%TYPE,

p\_DomainName IN vzwnet.Domain\_Inst.Domain\_Name%TYPE,

p\_Area IN xng\_reports.VZW\_NETWORK\_ORG.Area%TYPE,

p\_Region IN xng\_reports.VZW\_NETWORK\_ORG.Region%TYPE);

FUNCTION getSiteTechName (

p\_SiteInstId IN vzwnet.site\_attr\_settings.site\_inst\_id%TYPE )

RETURN vzwnet.site\_attr\_settings.attr\_value%TYPE;

FUNCTION getCellEquipDesc(

p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE )

RETURN NOCC\_LAST\_TEST\_DETAILS.INVENTORIED\_CELLS%TYPE;

FUNCTION getIsNoccMonitored (

p\_SiteInstId IN vzwnet.site\_inst.site\_inst\_id%TYPE )

RETURN VARCHAR2;

FUNCTION getFailedFirstEnvAttempt (

p\_SiteInstId IN vzwnet.site\_inst.site\_inst\_id%TYPE )

RETURN VARCHAR2;

FUNCTION getENVLastTestedDate (

p\_SiteInstId IN vzwnet.site\_attr\_settings.site\_inst\_id%TYPE,

p\_SiteType IN vzwnet.val\_attr\_group.class\_category%TYPE )

RETURN vzwnet.site\_attr\_settings.attr\_value%TYPE;

FUNCTION getHasAntennaTower(

p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE )

RETURN VARCHAR2;

FUNCTION getIsAntennaVZWOwned (

p\_SiteInstId IN vzwnet.equip\_inst.site\_inst\_id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE )

RETURN vzwnet.equip\_attr\_settings.attr\_value%TYPE;

FUNCTION getisLightingRequired (

p\_SiteInstId IN vzwnet.equip\_inst.site\_inst\_id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE )

RETURN vzwnet.equip\_attr\_settings.attr\_value%TYPE;

FUNCTION getTowerLightLastTestedDate (

p\_SiteInstId IN vzwnet.site\_attr\_settings.site\_inst\_id%TYPE,

p\_SiteType IN vzwnet.val\_attr\_group.class\_category%TYPE )

RETURN vzwnet.site\_attr\_settings.attr\_value%TYPE;

PROCEDURE insert\_NOCC\_LAST\_TEST\_SUMMARY(

p\_Area IN NOCC\_LAST\_TEST\_SUMMARY.Area%TYPE,

p\_Region IN NOCC\_LAST\_TEST\_SUMMARY.Region%TYPE,

p\_DomainId IN NOCC\_LAST\_TEST\_SUMMARY.domain\_id%TYPE,

p\_DomainName IN NOCC\_LAST\_TEST\_SUMMARY.domain\_name%TYPE,

p\_Sites IN NOCC\_LAST\_TEST\_SUMMARY.sites%TYPE,

p\_Antennas IN NOCC\_LAST\_TEST\_SUMMARY.antennas%TYPE,

p\_VZWOwned IN NOCC\_LAST\_TEST\_SUMMARY.vzw\_owned%TYPE,

p\_LightsReqd IN NOCC\_LAST\_TEST\_SUMMARY.lights\_reqd%TYPE,

p\_LightsTested IN NOCC\_LAST\_TEST\_SUMMARY.lights\_tested%TYPE,

p\_LightsOverDue IN NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_overdue%TYPE,

p\_LightsOk IN NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_ok%TYPE,

p\_Monitored IN NOCC\_LAST\_TEST\_SUMMARY.NOCC\_monitored\_sites%TYPE,

p\_EnvTested IN NOCC\_LAST\_TEST\_SUMMARY.env\_tested%TYPE,

p\_EnvOverDue IN NOCC\_LAST\_TEST\_SUMMARY.env\_test\_overdue%TYPE,

p\_EnvOK IN NOCC\_LAST\_TEST\_SUMMARY.env\_test\_ok%TYPE);

PROCEDURE insert\_NOCC\_Last\_Test\_Details(

p\_DomainId IN NOCC\_LAST\_TEST\_DETAILS.domain\_Id%TYPE,

p\_DomainName IN NOCC\_LAST\_TEST\_DETAILS.domain\_name%TYPE,

p\_SiteInstId IN NOCC\_LAST\_TEST\_DETAILS.site\_Inst\_Id%TYPE,

p\_SiteName IN NOCC\_LAST\_TEST\_DETAILS.site\_name%TYPE,

p\_SiteId IN NOCC\_LAST\_TEST\_DETAILS.site\_id%TYPE,

p\_Sitetype IN NOCC\_LAST\_TEST\_DETAILS.site\_type%TYPE,

p\_SiteTechName IN NOCC\_LAST\_TEST\_DETAILS.site\_tech\_name%TYPE,

p\_inventoried\_cells IN NOCC\_LAST\_TEST\_DETAILS.inventoried\_cells%TYPE,

p\_NOCC\_Monitored IN NOCC\_LAST\_TEST\_DETAILS.nocc\_Monitored%TYPE,

p\_vzw\_owned IN NOCC\_LAST\_TEST\_DETAILS.vzw\_owned%TYPE,

p\_lights\_reqd IN NOCC\_LAST\_TEST\_DETAILS.lights\_reqd%TYPE,

p\_LightLastTested IN NOCC\_LAST\_TEST\_DETAILS.Light\_last\_tested%TYPE,

p\_LightNextTest IN NOCC\_LAST\_TEST\_DETAILS.Light\_next\_test%TYPE,

p\_LightTestOverDue IN NOCC\_LAST\_TEST\_DETAILS.light\_test\_overdue%TYPE,

p\_ENVLastTested IN NOCC\_LAST\_TEST\_DETAILS.env\_last\_tested%TYPE,

p\_ENVNextTest IN NOCC\_LAST\_TEST\_DETAILS.env\_next\_test%TYPE,

p\_ENVTestOverDue IN NOCC\_LAST\_TEST\_DETAILS.env\_test\_overdue%TYPE,

p\_EquipInstid IN NOCC\_LAST\_TEST\_DETAILS.equip\_inst\_id%TYPE );

FUNCTION varchar2date( p\_StrDate IN VARCHAR2 ) RETURN DATE;

PROCEDURE removeDetails( p\_DomainId IN NOCC\_LAST\_TEST\_SUMMARY.domain\_id%TYPE );

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package SAM\_EBH\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."SAM\_EBH\_AUDIT"

AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: SAM\_EBH\_NGMLS\_AUDIT

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/19/2012 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--PROCEDURE audit\_ngMLS\_equip;

PROCEDURE audit\_AL\_ngMLS\_equip;

procedure insert\_AL\_ngMLS\_equip\_issues (processName varchar2);

procedure load\_ngMLS\_regional\_summary;

PROCEDURE assign\_area\_region\_to\_ngMLS;

PROCEDURE audit\_al\_ngmls\_vlans;

PROCEDURE insert\_AL\_ngMLS\_vlan\_issues;

PROCEDURE load\_AL\_vlan\_device\_summary;

PROCEDURE load\_vlan\_regional\_summary;

--PROCEDURE AUDIT\_CSR\_EQUIP;

PROCEDURE audit\_AL\_csr\_equip;

procedure insert\_AL\_csr\_equip\_issues (processName varchar2);

PROCEDURE insert\_csr\_issues (processname VARCHAR2, methodname VARCHAR2, insertSql VARCHAR2, errorMsg VARCHAR2);

PROCEDURE assign\_area\_region\_to\_csr;

procedure load\_csr\_clli\_summary ;

procedure load\_csr\_regional\_summary ;

procedure audit\_al\_csr\_vlans ;

procedure load\_csr\_vlan\_regional\_summ;

procedure insert\_AL\_CSR\_vlan\_issues;

PROCEDURE load\_vlan\_csr\_clli\_summary;

PROCEDURE assign\_area\_region\_to\_csr\_vlan;

PROCEDURE assign\_area\_region\_ngmls\_vlan;

END sam\_ebh\_audit;

/

--------------------------------------------------------

-- DDL for Package SEVONE\_PERFORMANCE

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."SEVONE\_PERFORMANCE" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: ehealth\_traffic\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/9/2013 ramamla 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE graniteSevOneAudit;

PROCEDURE graniteSevOneCsrAudit;

PROCEDURE graniteSevOneMplsAudit;

PROCEDURE graniteSevOneNgmlsAudit;

PROCEDURE graniteSevOne6500Audit;

PROCESSNAME varchar2(256) :='SEVONE\_PERFORMANCE\_AUDIT';

END sevone\_performance;

/

--------------------------------------------------------

-- DDL for Package SITE\_PORTAL\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."SITE\_PORTAL\_AUDIT"

IS

--

-- To modify this template, edit file PKGSPEC.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter package declarations as shown below

PROCEDURE UPDATESITEPORTALAUDIT;

PROCEDURE UPDATESITEPORTALAUDITSUMMARY;

PROCEDURE RESTORE\_SITEPORTALAUDIT;

AUDIT\_TABLE\_NAME CONSTANT varchar2(30) := 'NE\_VS\_XNG\_SP\_AUDIT\_WRK';

PROD\_AUDIT\_TABLE\_NAME CONSTANT varchar2(30) := 'NE\_VS\_XNG\_SP\_AUDIT';

AUDIT\_SUMMARY\_TABLE\_NAME CONSTANT varchar2(30) := 'NE\_VS\_XNG\_SP\_SUMMARY\_WRK';

PROD\_AUDIT\_SUMMARY\_TABLE\_NAME CONSTANT varchar2(30) := 'NE\_VS\_XNG\_SP\_SUMMARY';

END SITE\_PORTAL\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package T1\_SUMMARY

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."T1\_SUMMARY" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: T1\_Summary

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 4/13/2011 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SWITCH\_PATH\_T1\_SUMMARY varchar2(30) := 'switch\_path\_t1\_summary';

PROCEDURE load\_t1\_summary;

END T1\_Summary;

/

--------------------------------------------------------

-- DDL for Package TDM\_EBH\_30\_REPORT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."TDM\_EBH\_30\_REPORT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: TDM\_EBH\_30\_REPORT

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 3/8/2011 1. Created this package.

2,0 6/22/2011 2. Production Release

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE store\_ebh\_data;

PROCEDURE store\_tdm\_data;

PROCEDURE gen\_Reg\_Summary;

PROCEDURE gen\_ebh\_Reg\_Summ;

PROCEDURE gen\_tdm\_w\_ins\_date\_Reg\_Summ;

PROCEDURE gen\_tdm\_wo\_ins\_date\_Reg\_Summ;

PROCEDURE do\_all;

END TDM\_EBH\_30\_REPORT;

/

--------------------------------------------------------

-- DDL for Package VPI\_GI\_AUDIT\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."VPI\_GI\_AUDIT\_PKG" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: TEST

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 12/15/2015 Catherine Sayre Created

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

processName varchar2(100);

Procedure runAudit;

Procedure vpiGiMatches;

Procedure giOnly;

Procedure vpiGiSitesUnique;

END Vpi\_Gi\_Audit\_Pkg;

/

--------------------------------------------------------

-- DDL for Package VZ\_CONSTANTS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."VZ\_CONSTANTS" IS

/\*\*\*\*\*\*\*\*\*\*\*\*

$Log: VZ\_CONSTANTS.PKS,v $

Revision 1.2 2004/03/01 20:55:24 c\_anderj

update after initial testing on EA @ scorpion.ea

\*\*\*\*\*\*\*\*\*\*\*\*/

-- Default single schema database name

VS\_SINGLE\_SCHEMA\_DBNAME CONSTANT VARCHAR2(100) := 'vzwnet';

-- Determines if xml output allowed in reports that are so capable

VS\_DISPLAY\_DATA\_IN\_XML\_ALLOWED CONSTANT BOOLEAN:= TRUE;

--

VS\_DAD\_NAME VARCHAR2(100) := '/pls/xng\_reports/'; --NSP changed from ea to xng\_reports

-- VS\_DAD\_NAME VARCHAR2(100) := '/pls/ea/';

VS\_MACHINE\_NAME VARCHAR2(100) := 'txslxngpd.nss.vzwnet.com'; --NSP added it

-- VS\_HOMEPAGE CONSTANT VARCHAR2(200) := 'http://plowking.ddc.vzwcorp.com/wnea/owa/vz\_Reports.marketsMenu?ps\_rptName=''testdates''';

VS\_HOMEPAGE CONSTANT VARCHAR2(200) := 'http://'|| VS\_MACHINE\_NAME ||'/xngonlinereports.html';

-- NSP Where is this being used. This plowking link has been dead for ages.

-- path of resources like images, applets

VS\_SRC CONSTANT VARCHAR2(200) := 'http://'||VS\_MACHINE\_NAME||'/xng\_reports/';

-- NSP. Will have to make sure we create this points to the proper place on the server

-- variable to hold today's date.

VD\_TODAY CONSTANT DATE := SYSDATE;

VS\_BROWSER VARCHAR2(1000) := OWA\_UTIL.GET\_CGI\_ENV('HTTP\_USER\_AGENT');

------------------------------------------------------------------------------------------

-- place holder for next test time in last date tested report.

-- vn\_lightTestTime number(3) := 90 -- the next test date for lights is 90 days.

-- vn\_envTestTime number(3) := 365 -- the next test date for env is 365 days.

-------------------------------------------------------------------------------------------

--------markets in menu have hyperlinks to the entry point procs.

VS\_TESTDATES CONSTANT VARCHAR2(20) := 'MAIN'; -- show test date options.

----------------------------------------------------------

-- base URL 10.96.31.243 could not be used

--VS\_MIDWESTURL CONSTANT VARCHAR2(50) := 'http://10.96.31.243/'; -- for xperweb

-----------------------------------------------------------------------------

-- Change by Mike Jackson. Was getting ORA-06502 buffer too small error

--type site\_tbl\_ty is table of varchar2(100) index by binary\_integer;

type site\_tbl\_ty is table of varchar2(1000) index by binary\_integer;

type varchar100\_tbl\_ty is table of varchar2(100) index by binary\_integer;

type number9\_tbl\_ty is table of number(9) index by binary\_integer;

v\_highlight\_color varchar2(10) := 'GOLD';

v\_original\_color varchar2(10) := '#FFFFFF';

v\_highlight\_row varchar2(100) :=

'<TR bgcolor="#FFFFFF" onMouseOver="this.bgColor= '''|| v\_highlight\_color ||''';" onMouseOut="this.bgColor='''|| v\_original\_color ||''';">';

noFilterSelected EXCEPTION;

PRAGMA EXCEPTION\_INIT(noFilterSelected, -20001);

v\_all varchar2(20); -- used by reports to see if 'ALL' is selected from criteria for all

END VZ\_CONSTANTS;

/

--------------------------------------------------------

-- DDL for Package VZ\_REPORTS\_UTILS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."VZ\_REPORTS\_UTILS" IS

/\*\*\*\*\*\*\*\*\*\*\*\*

$Log: VZ\_REPORTS\_UTILS.PKS,v $

Revision 1.2 2004/03/01 20:55:24 c\_anderj

update after initial testing on EA @ scorpion.ea

\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION iff(tf IN BOOLEAN, yes IN VARCHAR2, no IN VARCHAR2 := NULL) RETURN VARCHAR2;

FUNCTION GETNAME(p\_userID varchar2) RETURN varchar2;

FUNCTION fs\_pickMonth return varchar2 ; -- -- generate a pick list of months, defaulting to current month.

procedure embedscript(p\_link varchar2, p\_imageLocation varchar2 );

procedure setlinkstyle;

PROCEDURE VZ\_RPT\_FOOTER;

PROCEDURE VZ\_RPT\_FOOTER(man\_url IN VARCHAR2);

PROCEDURE selectDateRange;

procedure setXLTab(p\_tabName varchar2 default 'sheet1');

procedure setTitleStyle;

procedure print\_xml\_tag(tag\_name in varchar2, tag\_value in varchar2, tag\_attr in varchar2 default null );

FUNCTION xdate( vs\_dateStr IN VARCHAR2 ) RETURN DATE; -- added to handle date strings from Xpercom

FUNCTION chopNPC( v VARCHAR2 ) RETURN varchar2; -- chop non printable chars - CR, LF, BS used while dumping into XL.

procedure setreportStyle; -- temporary; need to normalize styles

procedure html\_header(p\_displayType varchar2 default 'excel', p\_title varchar2 default '');

PROCEDURE write\_formatted\_stream( p\_column\_data\_inout IN OUT NOCOPY VARCHAR2, p\_data\_type varchar2 default null);

procedure print\_query( p\_sql in varchar2, p\_write\_type varchar2 default 'excel',

p\_title varchar2 default '',

p\_isHeader boolean default TRUE );

/\*

procedure print\_query( p\_sql in varchar2, p\_write\_type varchar2 default 'excel',

p\_title varchar2 default '',

p\_isHeader boolean default TRUE,

p\_xml varchar2 default 'n' );

\*/

--used to prepare where clause for use in reports.

PROCEDURE SET\_IN\_LIST(p\_varcharTbl IN VZ\_CONSTANTS.site\_tbl\_ty,

p\_in\_col varchar2,

p\_sql\_inout IN OUT long,

p\_colName varchar2,

p\_selectionTbl IN OUT VZ\_CONSTANTS.site\_tbl\_ty

) ;

--FUNCTION authorize return boolean;

function fn\_this\_caller return varchar2;

gs\_browser varchar2(100);

FUNCTION xml\_formatter(original\_sql IN VARCHAR2) RETURN VARCHAR2;

END VZ\_REPORTS\_UTILS;

/

--------------------------------------------------------

-- DDL for Package WATCHDOG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."WATCHDOG" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: WATCHDOG

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 6/1/2011 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION compare\_table\_columns(p\_process\_id number, p\_schema varchar2, p\_wk\_table varchar2, p\_app\_table varchar2) RETURN number;

PROCEDURE updateProcessStart(p\_processName IN VARCHAR2);

PROCEDURE updateProcessEnd(p\_processName IN VARCHAR2, p\_ExecutionStatus in VARCHAR2, p\_isReady in VARCHAR2);

PROCEDURE updateExtractDate(p\_processName IN VARCHAR2, p\_fileExtractDate in VARCHAR2);

PROCEDURE logError(p\_processName IN VARCHAR2, p\_errorCode in VARCHAR2, p\_errorMessage in VARCHAR2, p\_emailNotify in VARCHAR2);

PROCEDURE populate\_last\_successful\_date;

PROCEDURE populate\_last\_success\_dt\_proc(p\_processname IN VARCHAR2);

PROCEDURE retore\_is\_ready\_flag;

PROCEDURE update\_copy\_status(p\_processName IN VARCHAR2, p\_ExecutionStatus in VARCHAR2, p\_isReady in VARCHAR2);

PROCEDURE copy\_table\_prd\_to\_wk (p\_process\_name in varchar2);

PROCEDURE copy\_table\_wk\_to\_prd (p\_process\_name in varchar2);

PROCEDURE copy\_wk\_tables\_process;

PROCEDURE all\_processes\_wrapper;

END WATCHDOG;

/

--------------------------------------------------------

-- DDL for Package XNGSTD\_AUDIT\_BATCH\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."XNGSTD\_AUDIT\_BATCH\_PKG"

AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: XNGSTD\_AUDIT\_BATCH\_PKG

PURPOSE: To be called to batch process audit data for Granite Standard Audit reports (xngstd.war)

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 2/03/2015 David He 1. Created this package.

1.1 2/11/2015 David He 2. packed all related functions to this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE XNGSTD\_AUDIT\_BATCH\_PROC;

PROCEDURE XNGSTD\_AUDIT\_SUMMARY\_PROC;

PROCEDURE XNGSTD\_AUDIT\_SITE\_PROC;

PROCEDURE XNGSTD\_AUDIT\_EQUIP\_PROC;

PROCEDURE XNGSTD\_AUDIT\_PATH\_PROC;

PROCEDURE XNGSTD\_AUDIT\_SEGMENT\_PROC;

PROCEDURE XNGSTD\_AUDIT\_POST\_PROC;

FUNCTION GET\_SUBCATEGORY (p\_CIRC\_PATH\_HUM\_ID VARCHAR2,

p\_DOM\_IP VARCHAR2,

p\_type\_name VARCHAR2)

RETURN VARCHAR2;

FUNCTION GET\_PATH\_PATTERN (p\_Category VARCHAR2)

RETURN VARCHAR2;

FUNCTION CHECK\_ID (p\_id VARCHAR2, p\_pattern VARCHAR2)

RETURN NUMBER;

END XNGSTD\_AUDIT\_BATCH\_PKG;

/

--------------------------------------------------------

-- DDL for Package XNG\_CSR\_DISCOVERY

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."XNG\_CSR\_DISCOVERY" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: XNG\_CSR\_DISCOVERY

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 12/26/2012 1. Created this package.

This file is checked in as /network-discovery-audit/sql/sam/csr/xng\_csr\_discovery\_ddl.sql in svn

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE load\_granite\_CSR\_vlan\_paths;

END XNG\_CSR\_DISCOVERY;

/

--------------------------------------------------------

-- DDL for Package XNG\_NGMLS\_DISCOVERY

--------------------------------------------------------

CREATE OR REPLACE PACKAGE "XNG\_REPORTS"."XNG\_NGMLS\_DISCOVERY"

IS

--

-- To modify this template, edit file PKGSPEC.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter package declarations as shown below

PROCEDURE load\_granite\_ngMLS\_equip;

PROCEDURE load\_granite\_ngMLS\_vlan\_paths;

END; -- Package spec

/

--------------------------------------------------------

-- DDL for Package Body AUDIT\_REPORTS\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."AUDIT\_REPORTS\_PKG"

IS

--

-- Purpose: All the 'audit reports' will feed of off this central bussiness logic code.

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- ----------------- ---------------- --------------------------------------

-- Krishna Gadde Feb/March 2006 Initial version.

-- Starting point to load Audit report's data for Site, Equipment and Segment....

-- Modify for new Executive Summary requirements Jim Anderson June 2006, May 2007, Sept 2007

PROCEDURE loadAuditReportsData

IS

l\_area VARCHAR2(50);

CURSOR crsr\_area IS

SELECT DISTINCT area

FROM xng\_reports.vzw\_network\_org;

BEGIN

-- getBTP\_XNG\_AUDIT\_SUMMARY('DUMMY');

OPEN crsr\_area;

LOOP

FETCH crsr\_area INTO l\_area;

EXIT WHEN crsr\_area%NOTFOUND;

-- Send an area to the deligator...to process and load data...

deligate\_AuditReportsLoad( l\_area );

END LOOP;

CLOSE crsr\_area;

EXCEPTION

WHEN OTHERS THEN

IF ( crsr\_area%ISOPEN ) THEN

CLOSE crsr\_area;

END IF;

dbms\_output.put\_line(SubStr('Error in loadAuditReportData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Get the region's audit data for the given area....

PROCEDURE deligate\_AuditReportsLoad( p\_area IN xng\_reports.vzw\_network\_org.area%TYPE )

IS

CURSOR crsr\_region IS

SELECT DISTINCT region

FROM xng\_reports.vzw\_network\_org WHERE area = p\_area

ORDER BY 1;

l\_region VARCHAR2(50);

l\_value NUMBER := 0;

l\_domain\_inst\_id number;

BEGIN

OPEN crsr\_region;

LOOP

FETCH crsr\_region INTO l\_region;

EXIT WHEN crsr\_region%NOTFOUND;

-- Load Site\_summary table...

l\_domain\_inst\_id:=getDomainInstId(l\_region);

SaveRegion( p\_area, l\_region);

SaveSiteSummary( p\_area, l\_region, getSiteCountByRegion( l\_region ),get\_audited\_mtso\_count( l\_region ) );

-- Load Site\_audit table...

getSiteAttrCount( p\_area, l\_region );

getSitePotsCount( p\_area, l\_region );

getSiteReqFieldCount( p\_area, l\_region,l\_domain\_inst\_id );

-- Load equipmet\_audit(details).....

-- use vzw\_vsm\_xng\_xref table for data...

-- Load segment\_audit table...

SaveSegmentSummary( p\_area, l\_region );

-- dbms\_output.put\_line('Region: '||l\_region||'value: '||l\_value);

COMMIT;

END LOOP;

CLOSE crsr\_region;

EXCEPTION

WHEN OTHERS THEN

IF ( crsr\_region%ISOPEN ) THEN

CLOSE crsr\_region;

END IF;

dbms\_output.put\_line(SubStr('Error in deligateAuditReportsLoad(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Site --------

-- Get site count for a given region

FUNCTION getDomainInstId( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER

IS

l\_number number;

l\_char\_str varchar2(2000);

BEGIN

l\_char\_str:='select distinct(di.domain\_inst\_id)

from vzwnet.domain\_inst di,

xng\_reports.vzw\_network\_org vno

where di.domain\_name=vno.region\_domain\_name

and vno.region='''||p\_region||'''';

execute immediate l\_char\_str

into l\_number;

return l\_number;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in getDomainInstId(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

FUNCTION getSiteCountByRegion( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER

IS

retval NUMBER := 0;

BEGIN

SELECT count( distinct si.site\_inst\_id)

INTO retval

FROM vzwnet.domain\_inst di,

vzwnet.site\_domain\_map sdm,

vzwnet.site\_inst si,

vzwnet.val\_deploy\_status vds,

xng\_reports.vzw\_network\_org vno

WHERE di.domain\_name = vno.region\_domain\_name

AND vno.region = p\_region

AND sdm.domain\_inst\_id = di.domain\_inst\_id

AND vds.root\_status = 'A'

and vds.status\_name = si.status

AND si.site\_inst\_id = sdm.site\_inst\_id

AND (vno.inst\_id = 1127 and si.num IN ('HUB','MICROWAVE','TANDEM')

or vno.inst\_id != 1127

and si.num IN ('CELL','ENHANCER','HUB','MICROCELL', 'MICROWAVE','MTSO','TANDEM'))

AND si.site\_inst\_id not in (select site\_inst\_id

from vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

where van.group\_name='Site Alarm Verification Test'

and sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id

and van.attr\_name='Enviromentals NOCC Monitored'

and upper(sas.attr\_value) = 'NO');

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in getSiteCountByRegion(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

------------------------------------------

FUNCTION get\_audited\_mtso\_count(p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER

IS retval number := 0;

BEGIN

--------Dynamically Generate query

IF p\_region = 'NNO'

THEN

retval:=0;

ELSE

EXECUTE IMMEDIATE

'with AUDITED\_MTSOs as (

select distinct si.site\_hum\_id

from vzwnet.domain\_inst di,

vzwnet.site\_inst si,

vzwnet.site\_domain\_map sdm,

vzwnet.val\_deploy\_status vds,

xng\_reports.vzw\_network\_org vno

where si.num = ''MTSO''

and di.domain\_name = vno.region\_domain\_name

and vno.region = '''|| p\_region||'''

and sdm.domain\_inst\_id = di.domain\_inst\_id

and vds.root\_status =''A''

and vds.status\_name = si.status

and si.site\_inst\_id = sdm.site\_inst\_id

and si.site\_inst\_id not in (select site\_inst\_id

from vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

where van.group\_name=''Site Alarm Verification Test''

and sas.val\_attr\_inst\_id= van.val\_attr\_inst\_id

and van.attr\_name=''Enviromentals NOCC Monitored''

and upper(attr\_value) =''NO'')

)

select count(distinct site\_hum\_id)

from AUDITED\_MTSOs'

into retval;

END IF;

RETURN retval;

EXCEPTION

WHEN OTHERS

THEN

dbms\_output.put\_line(SubStr('Error in get\_audited\_mtso\_count(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

---------------------------------------------------------

FUNCTION getCompliantSiteCount(p\_name IN user\_tab\_columns.column\_name%TYPE,

p\_owner IN all\_tables.owner%TYPE,

p\_domain IN number)

RETURN NUMBER

IS retval number :=0;

--------Dynamically Generate Queries for IAN vs rest of regions

sqlstr varchar2(32767);

BEGIN

IF p\_domain = 1012

THEN

if p\_name != 'CLLI'

THEN

EXECUTE IMMEDIATE 'SELECT count ('||p\_name||')

FROM '||p\_owner||'.site\_inst si,

'||p\_owner||'.site\_domain\_map sdm,

'||p\_owner||'.val\_deploy\_status vds

WHERE si.site\_inst\_id = sdm. site\_inst\_id

AND sdm.domain\_inst\_id = '||p\_domain||'

AND si.num in (''HUB'',''MICROWAVE'',''TANDEM'')

AND vds.status\_name = si.status

AND vds.root\_status = ''A''

AND upper(rtrim(si.'||p\_name||')) != ''N/A''

AND upper(rtrim(si.'||p\_name||')) != ''NA''

AND upper(rtrim(si.'||p\_name||')) is not null

AND upper(rtrim(si.'||p\_name||')) != ''UNKNOWN''

AND si.site\_inst\_id not in (SELECT site\_inst\_id

FROM '||p\_owner||'.site\_attr\_settings sas,

'||p\_owner||'.val\_attr\_name van

WHERE van.group\_name=''Site Alarm Verification Test''

AND sas.val\_attr\_inst\_id= van.val\_attr\_inst\_id

AND van.attr\_name=''Enviromentals NOCC Monitored''

AND upper(attr\_value) =''NO'')'

into retval;

ELSE

execute immediate 'select count(distinct si1.site\_inst\_id)

from vzwnet.site\_inst si1,

clones.ntwk nw,

'||p\_owner||'.site\_domain\_map sdm,

'||p\_owner||'.val\_deploy\_status vds

where si1.site\_inst\_id not in (SELECT site\_inst\_id

FROM '||p\_owner||'.site\_attr\_settings sas,

'||p\_owner||'.val\_attr\_name van

WHERE van.group\_name=''Site Alarm Verification Test''

AND sas.val\_attr\_inst\_id= van.val\_attr\_inst\_id

AND van.attr\_name=''Enviromentals NOCC Monitored''

AND upper(attr\_value) =''NO'')

and si1.num in (''HUB'',''MICROWAVE'',''TANDEM'')

and si1.site\_inst\_id = sdm.site\_inst\_id

AND vds.status\_name = si1.status

AND vds.root\_status = ''A''

and sdm.domain\_inst\_id = '||p\_domain||'

and si1.clli is not null

and substr(si1.clli,1,4) = nw.geo\_code

and substr(si1.clli,5,2) = nw.GEOPOLITICAL\_CODE

and substr(si1.clli,7,2) = nw.NETWORK\_SITE\_CODE'

into retval;

END IF;

ELSE

if p\_name != 'CLLI'

THEN

EXECUTE IMMEDIATE 'SELECT count ('||p\_name||')

FROM '||p\_owner||'.site\_inst si,

'||p\_owner||'.site\_domain\_map sdm,

'||p\_owner||'.val\_deploy\_status vds

WHERE si.site\_inst\_id = sdm. site\_inst\_id

AND sdm.domain\_inst\_id = '||p\_domain||'

AND si.num in (''CELL'',''ENHANCER'',''HUB'',''MICROCELL'',

''MICROWAVE'',''MTSO'',''TANDEM'')

AND vds.status\_name = si.status

AND vds.root\_status = ''A''

AND upper(rtrim(si.'||p\_name||')) != ''N/A''

AND upper(rtrim(si.'||p\_name||')) != ''NA''

AND upper(rtrim(si.'||p\_name||')) is not null

AND upper(rtrim(si.'||p\_name||')) != ''UNKNOWN''

AND si.site\_inst\_id not in (SELECT site\_inst\_id

FROM '||p\_owner||'.site\_attr\_settings sas,

'||p\_owner||'.val\_attr\_name van

WHERE van.group\_name=''Site Alarm Verification Test''

AND sas.val\_attr\_inst\_id= van.val\_attr\_inst\_id

AND van.attr\_name=''Enviromentals NOCC Monitored''

AND upper(attr\_value) =''NO'')'

into retval;

ELSE

execute immediate 'select count(distinct si1.site\_inst\_id)

from vzwnet.site\_inst si1,

clones.ntwk nw,

vzwnet.site\_domain\_map sdm,

vzwnet.val\_deploy\_status vds

where si1.site\_inst\_id not in (SELECT site\_inst\_id

FROM vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

WHERE van.group\_name=''Site Alarm Verification Test''

AND sas.val\_attr\_inst\_id= van.val\_attr\_inst\_id

AND van.attr\_name=''Enviromentals NOCC Monitored''

AND upper(attr\_value) =''NO'')

and si1.num in (''CELL'',''ENHANCER'',''HUB'',''MICROCELL'',

''MICROWAVE'',''MTSO'',''TANDEM'')

and si1.site\_inst\_id = sdm.site\_inst\_id

AND vds.status\_name = si1.status

AND vds.root\_status = ''A''

and sdm.domain\_inst\_id = '||p\_domain||'

and si1.clli is not null

and substr(si1.clli,1,4) = nw.geo\_code

and substr(si1.clli,5,2) = nw.GEOPOLITICAL\_CODE

and substr(si1.clli,7,2) = nw.NETWORK\_SITE\_CODE

and si1.num in (''CELL'',''ENHANCER'',''HUB'',''MICROCELL'',

''MOCROWAVE'', ''MTSO'',''TANDEM'')'

into retval;

END IF;

END IF;

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in getCompliantSiteCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- SaveRegion Deletes data from region and inserts a blank row for area / region combination

PROCEDURE SaveRegion(p\_area IN xng\_reports.site\_summary.area%TYPE,

p\_region IN xng\_reports.site\_summary.region%TYPE)

IS

BEGIN -- Remove the old data for the give Area and region...

DELETE from xng\_reports.site\_summary WHERE area = p\_area AND region = p\_region;

if p\_region != 'OSS'

THEN

INSERT INTO xng\_reports.site\_summary( area, region,site\_count,extract\_date )

VALUES ( p\_area, p\_region ,1,sysdate);

END IF;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in SaveRegion(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Save Audit Summary table data ....

PROCEDURE SaveSiteSummary( p\_area IN xng\_reports.site\_summary.area%TYPE,

p\_region IN xng\_reports.site\_summary.region%TYPE,

p\_site\_count IN xng\_reports.site\_summary.site\_count%TYPE,

p\_mtso\_count IN xng\_reports.site\_summary.mtso\_count%TYPE)

IS

BEGIN

-- Remove the old data for the give Area and region...

-- DELETE from xng\_reports.site\_summary WHERE area = p\_area AND region = p\_region;

-- Save Area and region's site summary/count info...

update xng\_reports.site\_summary

set site\_count = p\_site\_count,

mtso\_count = p\_mtso\_count,

extract\_date = sysdate

where area = p\_area

and region = p\_region;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in SaveSiteSummary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-------------------------Site\_Audit-----------------------------------------

-- Get a count for a given region and attribute...

PROCEDURE getSiteAttrCount( p\_area IN xng\_reports.vzw\_network\_org.area%TYPE,

p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

IS

l\_attr\_name VARCHAR2(50);

l\_attr\_count VARCHAR2(50);

CURSOR crsr\_AttrCount( lp\_area IN xng\_reports.vzw\_network\_org.area%TYPE,

lp\_region IN xng\_reports.vzw\_network\_org.region%TYPE ) IS

select upper(van.attr\_name), count(\*) as attr\_count

from

( select distinct si.site\_inst\_id

from VZWNET.site\_inst si,

VZWNET.val\_deploy\_status vds,

VZWNET.domain\_inst di,

VZWNET.site\_domain\_map sdm,

XNG\_REPORTS.vzw\_network\_org vno

where vno.region = lp\_region

and vno.region\_domain\_name = di.domain\_name

and di.domain\_inst\_id = sdm.domain\_inst\_id

and ( vno.inst\_id = 1127

and si.num in ('HUB','MICROWAVE','TANDEM')

or vno.inst\_id != 1127

and si.num in ('CELL','HUB','MICROCELL',

'MICROWAVE','MTSO','TANDEM'))

and vds.status\_name = si.status

and vds.root\_status = 'A'

and si.site\_inst\_id = sdm.site\_inst\_id

and si.site\_inst\_id not in (select site\_inst\_id

from VZWNET.site\_attr\_settings sas,

VZWNET.val\_attr\_name van

where van.group\_name='Site Alarm Verification Test'

and sas.val\_attr\_inst\_id= van.val\_attr\_inst\_id

and van.attr\_name='Enviromentals NOCC Monitored' and upper(attr\_value) ='NO')) AUDITED\_SITES,

VZWNET.site\_attr\_settings sas,

VZWNET.val\_attr\_name van

where AUDITED\_SITES.site\_inst\_id = sas.site\_inst\_id

and (upper(van.attr\_name) in ('LOCAL FIRE NUMBER',

'LOCAL POLICE NUMBER',

'POWER ACCOUNT NUMBER',

'POWER COMPANY',

'POWER METER NUMBER',

'POWER PHONE NUMBER')

and upper(van.group\_name) = 'EMERGENCY CONTACTS'

or (upper(van.group\_name) = 'FIBER/COPPER/MW'

and upper(van.attr\_name) = 'DELIVERED TO DEMARC'))

and sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id

and rtrim(sas.attr\_value) is not null

and rtrim(upper(sas.attr\_value)) != 'N/A'

and rtrim(upper(sas.attr\_value)) != 'NA'

and rtrim(upper(sas.attr\_value)) != 'UNKNOWN'

group by attr\_name;

BEGIN

OPEN crsr\_AttrCount( p\_area, p\_region );

LOOP

FETCH crsr\_AttrCount INTO l\_attr\_name, l\_attr\_count;

EXIT WHEN crsr\_AttrCount%NOTFOUND;

SaveSiteAudit( p\_area, p\_region, l\_attr\_name, l\_attr\_count );

END LOOP;

CLOSE crsr\_AttrCount;

EXCEPTION

WHEN OTHERS THEN

IF ( crsr\_AttrCount%ISOPEN ) THEN

CLOSE crsr\_AttrCount;

END IF;

dbms\_output.put\_line(SubStr('Error in getSiteAttrCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Get count for Compliant Site Pots Number --------------------------

-- Only requirements are that it exist and is not null -----------------

PROCEDURE getSitePotsCount( p\_area IN xng\_reports.vzw\_network\_org.area%TYPE,

p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

IS

l\_attr\_name VARCHAR2(50);

l\_attr\_count VARCHAR2(50);

CURSOR crsr\_AttrCount( lp\_area IN xng\_reports.vzw\_network\_org.area%TYPE,

lp\_region IN xng\_reports.vzw\_network\_org.region%TYPE ) IS

select upper(van.attr\_name), count(\*) as attr\_count

from

( select distinct si.site\_inst\_id

from VZWNET.site\_inst si,

VZWNET.val\_deploy\_status vds,

VZWNET.domain\_inst di,

VZWNET.site\_domain\_map sdm,

XNG\_REPORTS.vzw\_network\_org vno

where vno.region = lp\_region

and vno.region\_domain\_name = di.domain\_name

and di.domain\_inst\_id = sdm.domain\_inst\_id

and ( vno.inst\_id = 1127

and si.num in ('HUB','MICROWAVE','TANDEM')

or vno.inst\_id != 1127

and si.num in ('CELL','HUB','MICROCELL',

'MICROWAVE','MTSO','TANDEM'))

and vds.status\_name = si.status

and vds.root\_status = 'A'

and si.site\_inst\_id = sdm.site\_inst\_id

and si.site\_inst\_id not in (select site\_inst\_id

from VZWNET.site\_attr\_settings sas,

VZWNET.val\_attr\_name van

where van.group\_name='Site Alarm Verification Test'

and sas.val\_attr\_inst\_id= van.val\_attr\_inst\_id

and van.attr\_name='Enviromentals NOCC Monitored' and upper(attr\_value) ='NO')) AUDITED\_SITES,

VZWNET.site\_attr\_settings sas,

VZWNET.val\_attr\_name van

where AUDITED\_SITES.site\_inst\_id = sas.site\_inst\_id

and upper(van.attr\_name) = 'SITE POTS NUMBER'

and upper(van.group\_name) = 'EMERGENCY CONTACTS'

and sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id

and rtrim(sas.attr\_value) is not null

group by attr\_name;

BEGIN

OPEN crsr\_AttrCount( p\_area, p\_region );

LOOP

FETCH crsr\_AttrCount INTO l\_attr\_name, l\_attr\_count;

EXIT WHEN crsr\_AttrCount%NOTFOUND;

SaveSiteAudit( p\_area, p\_region, l\_attr\_name, l\_attr\_count );

END LOOP;

CLOSE crsr\_AttrCount;

EXCEPTION

WHEN OTHERS THEN

IF ( crsr\_AttrCount%ISOPEN ) THEN

CLOSE crsr\_AttrCount;

END IF;

dbms\_output.put\_line(SubStr('Error in getSitePotsCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Get required field usage/populated count for an attribute in a region...

PROCEDURE getSiteReqFieldCount( p\_area IN xng\_reports.site\_summary.area%TYPE,

p\_region IN xng\_reports.site\_summary.region%TYPE,

p\_domain\_inst\_id IN vzwnet.domain\_inst.domain\_inst\_id%TYPE)

IS

l\_req\_field\_name varchar2(50);

l\_req\_field\_count NUMBER;

-- l\_domain\_inst\_id NUMBER;

-- l\_domain\_inst\_id := getDomainInstId('Central Texas');

CURSOR crsr\_field\_list

IS

select a.column\_name colName ,

getCompliantSiteCount(a.column\_name, 'VZWNET',p\_domain\_inst\_id) siteVals

from all\_tab\_columns a

where table\_name = 'SITE\_INST'

and upper(column\_name) in ('SITE\_HUM\_ID',

'CLLI',

'NUM',

'LATITUDE',

'LONGITUDE',

'CONTACTS',

'ADDRESS',

'POST\_CODE\_1',

'CITY',

'STATE\_PROV',

'COUNTY')

and owner = upper('VZWNET')

order by 1 ;

BEGIN

-- l\_req\_field\_name varchar2(50);

-- l\_req\_field\_count NUMBER;

OPEN crsr\_field\_list;

LOOP

FETCH crsr\_field\_list INTO l\_req\_field\_name, l\_req\_field\_count;

EXIT WHEN crsr\_field\_list%NOTFOUND;

-- Save Site required field count in Site\_sudit table...

SaveSiteAudit( p\_area, p\_region, l\_req\_field\_name, l\_req\_field\_count );

END LOOP;

CLOSE crsr\_field\_list;

EXCEPTION

WHEN OTHERS THEN

IF ( crsr\_field\_list%ISOPEN ) THEN

CLOSE crsr\_field\_list;

END IF;

dbms\_output.put\_line(SubStr('Error in getSiteReqFieldCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Save Audit Summary table data ....

PROCEDURE SaveSiteAudit( p\_area IN xng\_reports.site\_audit.area%TYPE,

p\_region IN xng\_reports.site\_audit.region%TYPE,

p\_fieldName IN xng\_reports.site\_audit.attr\_name%TYPE,

p\_usage\_count IN xng\_reports.site\_audit.populated\_count%TYPE)

IS

BEGIN

-- Remove the old data for the give Area and region...

DELETE from xng\_reports.site\_audit sa

WHERE sa.area = p\_area

AND sa.region = p\_region

AND sa.attr\_name = p\_fieldName;

-- Save Area and region's site summary/count info...

INSERT INTO xng\_reports.site\_audit( area, region, attr\_name, populated\_count, extract\_date)

VALUES( p\_area, p\_region, p\_fieldName, p\_usage\_count, sysdate );

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in SaveSiteAudit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

/\*PROCEDURE saveBTP\_XNG\_AUDIT\_SUMMARY

(p\_AREA IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.AREA%TYPE,

p\_REGION IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.REGION%TYPE,

p\_CKTS\_MATCHED IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CKTS\_MATCHED%TYPE,

p\_CKT\_COUNT IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CKT\_COUNT%TYPE,

p\_CHARGE\_AMT\_TOTAL IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CHARGE\_AMT\_TOTAL%TYPE,

p\_CHARGES\_MATCHED IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CHARGES\_MATCHED%TYPE,

p\_CKTS\_MATCHED\_PCT\_STR IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CKTS\_MATCHED\_PCT\_STR%TYPE,

p\_CKTS\_MATCHED\_PCT\_NUM IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CKTS\_MATCHED\_PCT\_NUM%TYPE,

p\_CHARGES\_MATCHED\_PCT\_STR IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CHARGES\_MATCHED\_PCT\_STR%TYPE,

p\_CHARGES\_MATCHED\_PCT\_NUM IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.CHARGES\_MATCHED\_PCT\_NUM%TYPE)

IS

BEGIN

DELETE from XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY

WHERE AREA=p\_AREA

and REGION=p\_REGION;

insert into XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY

(AREA,

REGION,

CKTS\_MATCHED,

CKT\_COUNT,

CHARGE\_AMT\_TOTAL,

CHARGES\_MATCHED,

CKTS\_MATCHED\_PCT\_STR,

CKTS\_MATCHED\_PCT\_NUM,

CHARGES\_MATCHED\_PCT\_STR,

CHARGES\_MATCHED\_PCT\_NUM,

EXTRACT\_DATE)

values

(p\_AREA,

p\_REGION,

p\_CKTS\_MATCHED,

p\_CKT\_COUNT,

p\_CHARGE\_AMT\_TOTAL,

p\_CHARGES\_MATCHED,

p\_CKTS\_MATCHED\_PCT\_STR,

p\_CKTS\_MATCHED\_PCT\_NUM,

p\_CHARGES\_MATCHED\_PCT\_STR,

p\_CHARGES\_MATCHED\_PCT\_NUM,

sysdate);

COMMIT;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line

(SubStr('Error in saveBTP\_XNG\_AUDIT\_SUMMARY(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;\*/

-- Segment -----------

-- Get Path count for a given region

FUNCTION getPathCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER

IS

l\_domain\_inst\_id number;

retval NUMBER := 0;

BEGIN

select count (cpi.circ\_path\_inst\_id)

into retval

from vzwnet.circ\_path\_inst cpi,

(select distinct pdm.circ\_path\_inst\_id

from vzwnet.path\_domain\_map pdm,

xng\_reports.vzw\_network\_org vno,

vzwnet.domain\_inst di

where vno.region = p\_region

and vno.region\_domain\_name = di.domain\_name

and pdm.domain\_inst\_id = di.domain\_inst\_id) pi\_ids

where cpi.circ\_path\_inst\_id=pi\_ids.circ\_path\_inst\_id;

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in getPathCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Segment count...

FUNCTION getSegmentCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE,

alltel\_mode IN varchar )

RETURN NUMBER

IS

retval NUMBER := 0;

-- Default is All Minus Alltel

alltel\_sql varchar(200) := ' and cdm.circ\_inst\_id not in (select distinct(circ\_inst\_id) from vzwnet.circ\_domain\_map where domain\_inst\_id=1191) ';

BEGIN

if alltel\_mode = 'all\_minus\_alltel' then null; end if; -- Default

if alltel\_mode = 'all\_including\_alltel' then alltel\_sql := ' '; end if;

if alltel\_mode = 'alltel\_only' then alltel\_sql := ' and cdm.circ\_inst\_id in (select distinct(circ\_inst\_id) from vzwnet.circ\_domain\_map where domain\_inst\_id=1191) '; end if;

execute immediate '

select count(distinct ci.circ\_inst\_id)

from vzwnet.circ\_inst ci,

(select distinct cdm.circ\_inst\_id

from vzwnet.circ\_domain\_map cdm,

xng\_reports.vzw\_network\_org vno,

vzwnet.domain\_inst di

where vno.region = '''||p\_region||'''

and vno.region\_domain\_name = di.domain\_name

and cdm.domain\_inst\_id = di.domain\_inst\_id '||alltel\_sql||') ci\_ids

where upper(ci.bandwidth) !=''DS0''

and ci.circ\_inst\_id=ci\_ids.circ\_inst\_id

and ci.status = ''Live''' into retval;

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in getSegmentCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Segments with paths....

FUNCTION getSegmentWithPathCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE,

alltel\_mode IN varchar )

RETURN NUMBER

IS

retval NUMBER := 0;

-- Default is All Minus Alltel

alltel\_sql varchar(200) := ' and cdm.circ\_inst\_id not in (select distinct(circ\_inst\_id) from vzwnet.circ\_domain\_map where domain\_inst\_id=1191) ';

BEGIN

if alltel\_mode = 'all\_minus\_alltel' then null; end if; -- Default

if alltel\_mode = 'all\_including\_alltel' then alltel\_sql := ' '; end if;

if alltel\_mode = 'alltel\_only' then alltel\_sql := ' and cdm.circ\_inst\_id in (select distinct(circ\_inst\_id) from vzwnet.circ\_domain\_map where domain\_inst\_id=1191) '; end if;

execute immediate '

select count(distinct ci.circ\_inst\_id)

from vzwnet.circ\_inst ci,

vzwnet.circ\_path\_inst cpi,

(select distinct cdm.circ\_inst\_id

from vzwnet.circ\_domain\_map cdm,

xng\_reports.vzw\_network\_org vno,

vzwnet.domain\_inst di

where vno.region = '''||p\_region||'''

and vno.region\_domain\_name = di.domain\_name

and cdm.domain\_inst\_id = di.domain\_inst\_id '||alltel\_sql||') ci\_ids

where ci.circ\_inst\_id=ci\_ids.circ\_inst\_id

and (cpi.circ\_path\_inst\_id=ci.CIRC\_PATH\_INST\_ID

or cpi.circ\_path\_inst\_id=ci.NEXT\_PATH\_INST\_ID)

and ci.status = ''Live''

and upper(ci.bandwidth) !=''DS0''' into retval;

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in getSegmentWithPathCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Segments with OUT paths....

FUNCTION getSegmentWithOutPathCount( p\_region IN xng\_reports.vzw\_network\_org.region%TYPE )

RETURN NUMBER

IS

retval NUMBER := 0;

BEGIN

select count(\*)

into retval

from vzwnet.circ\_inst ci,

vzwnet.circ\_path\_inst cpi,

vzwnet.val\_deploy\_status vds,

(select distinct cdm.circ\_inst\_id

from vzwnet.circ\_domain\_map cdm,

xng\_reports.vzw\_network\_org vno,

vzwnet.domain\_inst di

where vno.region = p\_region

and vno.region\_domain\_name = di.domain\_name

and cdm.domain\_inst\_id = di.domain\_inst\_id) ci\_ids

where ci.circ\_inst\_id=ci\_ids.circ\_inst\_id

and upper(ci.bandwidth) !='DS0'

and (cpi.circ\_path\_inst\_id=ci.CIRC\_PATH\_INST\_ID

or cpi.circ\_path\_inst\_id=ci.NEXT\_PATH\_INST\_ID)

and ci.status = vds.status\_name

and vds.root\_status in ('A','P');

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in getSegmentWithOutPathCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Save Segment Summary table data ....

PROCEDURE SaveSegmentSummary( p\_area IN xng\_reports.site\_audit.area%TYPE,

p\_region IN xng\_reports.site\_audit.region%TYPE)

IS

l\_path\_count number := 0;

l\_seg\_count number := 0;

l\_seg\_with\_path number := 0;

BEGIN

l\_path\_count := getPathCount( p\_region );

l\_seg\_count := getSegmentCount( p\_region, '' );

l\_seg\_with\_path := getSegmentWithPathCount( p\_region, '' );

-- Remove the old data for the give Area and region...

DELETE from xng\_reports.segment\_summary ss

WHERE ss.area = p\_area

AND ss.region = p\_region;

-- Save Area and region's Segment summary/count info...

INSERT INTO xng\_reports.segment\_summary

( area,

region,

paths,

segments,

segments\_w\_path,

segments\_wo\_path,

extract\_date)

VALUES( p\_area,

p\_region,

l\_path\_count,

l\_seg\_count,

l\_seg\_with\_path,

l\_seg\_count - l\_seg\_with\_path,

-- decode(l\_seg\_count,0,0,trunc((l\_seg\_with\_path/l\_seg\_count)\*100,0)),

sysdate );

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line

(SubStr('Error in SaveSegmentSummary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Segment Audit.....

-- Get BTP-Xng Audit results and calls SaveBTP\_XNG\_SUMMARY to store the results in a summary table...

/\* PROCEDURE getBTP\_XNG\_AUDIT\_SUMMARY(p\_AREA IN XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY.AREA%TYPE )

IS

l\_AREA varchar2(50);

l\_REGION varchar2(50);

l\_CKTS\_MATCHED number;

l\_CKT\_COUNT number;

l\_CHARGE\_AMT\_TOTAL number;

l\_CHARGES\_MATCHED number;

l\_CKTS\_MATCHED\_PCT\_STR varchar2(150);

l\_CKTS\_MATCHED\_PCT\_NUM number;

l\_CHARGES\_MATCHED\_PCT\_STR varchar2(7);

l\_CHARGES\_MATCHED\_PCT\_NUM number;

CURSOR crsr\_BTP\_XNG\_SUMMARY

IS select bxa.xng\_area,

bxa.xng\_region, --add "and bxa.uniquely\_identified ='Y'" to when statements to show multiple xng matches as errors

sum((case when bxa.btp\_xng\_match ='Y'then 1 else 0 end)) as ckts\_matched,

count(\*) as ckt\_count,

sum(bxa.charge\_amount) as charge\_amt\_tot,

sum((case when bxa.btp\_xng\_match ='Y'then bxa.charge\_amount else 0 end)) as charges\_matched,

to\_char(sum((case

when bxa.btp\_xng\_match ='Y'

then 1

else 0 end))

/count(\*)\*100,'999.99') as ckts\_matched\_pct\_str,

sum((case

when bxa.btp\_xng\_match ='Y'

then 1

else 0 end))

/count(\*)\*100 as ckts\_matched\_pct\_num,

to\_char(sum((case

when bxa.btp\_xng\_match ='Y'

then bxa.charge\_amount

else 0 end))

/sum(bxa.charge\_amount)\*100,'999.99') as chgs\_matched\_pct\_str,

sum((case

when btp\_xng\_match ='Y'

then bxa.charge\_amount

else 0 end))

/sum(bxa.charge\_amount)\*100 as chgs\_matched\_pct\_num

from XNG\_REPORTS.BTP\_XNG\_AUDIT\_HIST bxa

where bxa.xng\_area is not null

group by rollup(bxa.xng\_area,bxa.xng\_region);

BEGIN

delete from XNG\_REPORTS.BTP\_XNG\_AUDIT\_SUMMARY;

OPEN crsr\_BTP\_XNG\_SUMMARY;

LOOP

FETCH crsr\_BTP\_XNG\_SUMMARY INTO l\_AREA,

l\_REGION,

l\_CKTS\_MATCHED,

l\_CKT\_COUNT,

l\_CHARGE\_AMT\_TOTAL,

l\_CHARGES\_MATCHED,

l\_CKTS\_MATCHED\_PCT\_STR,

l\_CKTS\_MATCHED\_PCT\_NUM,

l\_CHARGES\_MATCHED\_PCT\_STR,

l\_CHARGES\_MATCHED\_PCT\_NUM ;

EXIT WHEN crsr\_BTP\_XNG\_SUMMARY%NOTFOUND;

-- Save BTP\_XNG\_AUDIT\_SUMMARY into BTP\_XNG\_AUDIT\_SUMMMARY table...

SaveBTP\_XNG\_AUDIT\_SUMMARY(l\_AREA,

l\_REGION,

l\_CKTS\_MATCHED,

l\_CKT\_COUNT,

l\_CHARGE\_AMT\_TOTAL,

l\_CHARGES\_MATCHED,

l\_CKTS\_MATCHED\_PCT\_STR,

l\_CKTS\_MATCHED\_PCT\_NUM,

l\_CHARGES\_MATCHED\_PCT\_STR,

l\_CHARGES\_MATCHED\_PCT\_NUM );

END LOOP;

CLOSE crsr\_BTP\_XNG\_SUMMARY;

EXCEPTION

WHEN OTHERS THEN

IF ( crsr\_BTP\_XNG\_SUMMARY%ISOPEN ) THEN

CLOSE crsr\_BTP\_XNG\_SUMMARY;

END IF;

dbms\_output.put\_line(SubStr('Error in get\_BTP\_XNG\_SUMMARY(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;\*/

END;

/

--------------------------------------------------------

-- DDL for Package Body AUDIT\_STATUS\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."AUDIT\_STATUS\_PKG"

IS

PROCEDURE process\_audit\_status

IS

PROCESS\_NAME varchar2(100) := 'AUDIT\_STATUS';

BEGIN

watchdog.updateProcessStart(PROCESS\_NAME);

update\_audit\_status();

delete\_audit\_status();

watchdog.updateProcessEnd(PROCESS\_NAME,'STATUS\_SUCCESS','Y');

END process\_audit\_status;

PROCEDURE delete\_audit\_status

IS

v\_inst\_id NUMBER;

message varchar2(250);

CURSOR modified\_staus

IS

SELECT uda, aus.inst\_id, aus.audit\_name, usm.UPDATE\_TYPE

FROM xng\_apps.uda\_sql\_map usm, xng\_reports.audit\_status aus

WHERE usm.operation\_type = 'UPDATE\_BLANK'

AND usm.update\_type in ('updateShelf','updatePath')

AND usm.audit\_type = aus.audit\_name

AND aus.match\_code = 'Xng Only'

AND update\_date < SYSDATE - 4;

BEGIN

FOR rec IN modified\_staus

LOOP

v\_inst\_id := null;

message := 'Error: ' ||'(): Can''t delete from table: XNG\_REPORTS.audit\_status';

if(rec.update\_type = 'updatePath') then

select rec.inst\_id into v\_inst\_id from dual

WHERE NOT EXISTS (

SELECT 1

FROM vzwnet.val\_attr\_name van,

vzwnet.circ\_path\_attr\_settings eas

WHERE van.val\_attr\_inst\_id = eas.val\_attr\_inst\_id

AND van.attr\_name IN ( SELECT REGEXP\_SUBSTR (uda, '[^,]+', 1, rn) uda

FROM (select rec.uda

FROM dual)

CROSS JOIN

(SELECT ROWNUM rn

FROM (SELECT MAX

(LENGTH

(REGEXP\_REPLACE

(uda,

'[^,]+'

)

)

)

+ 1 max\_value

FROM (select rec.uda

FROM dual))

CONNECT BY LEVEL <= max\_value)

WHERE REGEXP\_SUBSTR (uda, '[^,]+', 1, rn) IS NOT NULL)

AND eas.circ\_path\_inst\_id = rec.inst\_id

AND eas.attr\_value IS NOT NULL);

Else

select rec.inst\_id into v\_inst\_id from dual

WHERE NOT EXISTS (

SELECT 1

FROM vzwnet.val\_attr\_name van,

vzwnet.EQUIP\_ATTR\_SETTINGS eas

WHERE van.val\_attr\_inst\_id = eas.val\_attr\_inst\_id

AND van.attr\_name IN ( SELECT REGEXP\_SUBSTR (uda, '[^,]+', 1, rn) uda

FROM (select rec.uda

FROM dual)

CROSS JOIN

(SELECT ROWNUM rn

FROM (SELECT MAX

(LENGTH

(REGEXP\_REPLACE

(uda,

'[^,]+'

)

)

)

+ 1 max\_value

FROM (select rec.uda

FROM dual))

CONNECT BY LEVEL <= max\_value)

WHERE REGEXP\_SUBSTR (uda, '[^,]+', 1, rn) IS NOT NULL)

AND eas.EQUIP\_INST\_ID = rec.inst\_id

AND eas.attr\_value IS NOT NULL);

End if;

if(v\_inst\_id is not null)then

DBMS\_OUTPUT.put\_line (v\_inst\_id || rec.audit\_name);

DELETE xng\_reports.audit\_status

WHERE inst\_id = v\_inst\_id AND audit\_name = rec.audit\_name;

end if;

END LOOP;

delete from audit\_status a where a.inst\_id in (

SELECT aus.inst\_id

FROM xng\_apps.uda\_sql\_map usm, xng\_reports.audit\_status aus

WHERE usm.operation\_type = 'UPDATE\_BLANK'

AND usm.update\_type in ('updateShelf','updatePath')

AND usm.audit\_type = aus.audit\_name

AND aus.match\_code = 'Xng Only'

AND update\_date < SYSDATE - 4) ;

commit;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

v\_inst\_id := NULL;

END delete\_audit\_status;

Procedure update\_audit\_status

IS

CURSOR modified\_staus

IS

--vLANS

SELECT sam\_csr\_vlan.match\_code AS match\_code,

sam\_csr\_vlan.vlan\_inst\_id AS inst\_id, ast.audit\_name AS audit\_name

FROM xng\_reports.sam\_csr\_vlan\_audit sam\_csr\_vlan,

xng\_reports.audit\_status ast

WHERE sam\_csr\_vlan.vlan\_inst\_id = ast.inst\_id

AND sam\_csr\_vlan.match\_code != ast.match\_code

and port\_name like '%1/5/8%'

AND ast.audit\_name = 'SAM\_CSR\_VLAN\_AUDIT'

union

SELECT sam\_ngmls\_vlan.match\_code AS match\_code,

sam\_ngmls\_vlan.vlan\_inst\_id AS inst\_id, ast.audit\_name AS audit\_name

FROM xng\_reports.sam\_ngmls\_vlan\_audit sam\_ngmls\_vlan,

xng\_reports.audit\_status ast

WHERE sam\_ngmls\_vlan.vlan\_inst\_id = ast.inst\_id

AND sam\_ngmls\_vlan.match\_code != ast.match\_code

AND length(sam\_ngmls\_vlan.PORT\_NAME) > 5

AND ast.audit\_name = 'SAM\_NGMLS\_VLAN\_AUDIT'

union

SELECT ncm\_ngmls\_vlan.match\_code AS match\_code,

ncm\_ngmls\_vlan.vlan\_inst\_id AS inst\_id, ast.audit\_name AS audit\_name

FROM xng\_reports.ncm\_ngmls\_vlan\_audit ncm\_ngmls\_vlan,

xng\_reports.audit\_status ast

WHERE ncm\_ngmls\_vlan.vlan\_inst\_id = ast.inst\_id

AND ncm\_ngmls\_vlan.match\_code != ast.match\_code

AND NE\_PORT\_IP like '%.%.%.%'

AND ast.audit\_name = 'NCM\_NGMLS\_VLAN\_AUDIT'

union

SELECT ncm\_csr\_vlan.match\_code AS match\_code,

ncm\_csr\_vlan.vlan\_inst\_id AS inst\_id, ast.audit\_name AS audit\_name

FROM xng\_reports.ncm\_csr\_vlan\_audits ncm\_csr\_vlan,

xng\_reports.audit\_status ast

WHERE ncm\_csr\_vlan.vlan\_inst\_id = ast.inst\_id

AND ncm\_csr\_vlan.match\_code != ast.match\_code

and ncm\_csr\_vlan.displayed\_name = 'GigabitEthernet0/0/0'

and ncm\_csr\_vlan.DEVICE\_IP like '%.%.%.%'

AND ast.audit\_name = 'NCM\_CSR\_VLAN\_AUDIT'

union

--Devices

SELECT ncm\_ngmls.match\_code AS match\_code,

ncm\_ngmls.NE\_INST\_ID AS inst\_id, ast.audit\_name AS audit\_name

FROM xng\_reports.NCM\_GI\_CI\_NGMLS\_AUDIT ncm\_ngmls,

xng\_reports.audit\_status ast

WHERE ncm\_ngmls.NE\_INST\_ID = ast.inst\_id

AND ncm\_ngmls.match\_code != ast.match\_code

AND ast.audit\_name = 'NCM\_NGMLS\_AUDIT'

union

SELECT ncm\_csr.match\_code AS match\_code,

ncm\_csr.EQUIP\_INST\_ID AS inst\_id, ast.audit\_name AS audit\_name

FROM xng\_reports.NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT ncm\_csr,

xng\_reports.audit\_status ast

WHERE ncm\_csr.EQUIP\_INST\_ID = ast.inst\_id

AND ncm\_csr.match\_code != ast.match\_code

AND ast.audit\_name = 'NCM\_CSR\_AUDIT'

union

SELECT sam\_ngmls.match\_code AS match\_code,

sam\_ngmls.NE\_INST\_ID AS inst\_id, ast.audit\_name AS audit\_name

FROM xng\_reports.SAM\_GI\_AL\_NGMLS\_AUDIT sam\_ngmls,

xng\_reports.audit\_status ast

WHERE sam\_ngmls.NE\_INST\_ID = ast.inst\_id

AND sam\_ngmls.match\_code != ast.match\_code

AND ast.audit\_name = 'SAM\_NGMLS\_AUDIT'

union

SELECT sam\_csr.match\_code AS match\_code,

sam\_csr.EQUIP\_INST\_ID AS inst\_id, ast.audit\_name AS audit\_name

FROM xng\_reports.SAM\_GI\_AL\_CSR\_AUDIT sam\_csr,

xng\_reports.audit\_status ast

WHERE sam\_csr.EQUIP\_INST\_ID = ast.inst\_id

AND sam\_csr.match\_code != ast.match\_code

AND ast.audit\_name = 'SAM\_CSR\_AUDIT' ;

BEGIN

--Insert 'BOTH' if it is not available with update time stamp

INSERT INTO xng\_reports.audit\_status

SELECT vlan\_inst\_id AS inst\_id, 'SAM\_CSR\_VLAN\_AUDIT', match\_code,

SYSDATE

FROM xng\_reports.sam\_csr\_vlan\_audit sam\_csr\_vlan

WHERE match\_code = 'BOTH' and port\_name like '%1/5/8%'

AND NOT EXISTS (

SELECT 'x'

FROM xng\_reports.audit\_status

WHERE inst\_id = sam\_csr\_vlan.vlan\_inst\_id

AND audit\_name = 'SAM\_CSR\_VLAN\_AUDIT')

union

SELECT vlan\_inst\_id AS inst\_id, 'SAM\_NGMLS\_VLAN\_AUDIT', match\_code,

SYSDATE

FROM xng\_reports.sam\_ngmls\_vlan\_audit sam\_ngmls\_vlan

WHERE match\_code = 'BOTH' AND length(sam\_ngmls\_vlan.PORT\_NAME) > 5

AND NOT EXISTS (

SELECT 'x'

FROM xng\_reports.audit\_status

WHERE inst\_id = sam\_ngmls\_vlan.vlan\_inst\_id

AND audit\_name = 'SAM\_NGMLS\_VLAN\_AUDIT')

union

SELECT vlan\_inst\_id AS inst\_id, 'NCM\_NGMLS\_VLAN\_AUDIT', match\_code,

SYSDATE

FROM xng\_reports.ncm\_ngmls\_vlan\_audit ncm\_ngmls\_vlan

WHERE match\_code = 'BOTH' AND NE\_PORT\_IP like '%.%.%.%'

AND NOT EXISTS (

SELECT 'x'

FROM xng\_reports.audit\_status

WHERE inst\_id = ncm\_ngmls\_vlan.vlan\_inst\_id

AND audit\_name = 'NCM\_NGMLS\_VLAN\_AUDIT')

union

SELECT vlan\_inst\_id AS inst\_id, 'NCM\_CSR\_VLAN\_AUDIT', match\_code,

SYSDATE

FROM xng\_reports.ncm\_csr\_vlan\_audits ncm\_csr\_vlan

WHERE match\_code = 'BOTH' and ncm\_csr\_vlan.DEVICE\_IP like '%.%.%.%'

and ncm\_csr\_vlan.displayed\_name = 'GigabitEthernet0/0/0'

AND NOT EXISTS (

SELECT 'x'

FROM xng\_reports.audit\_status

WHERE inst\_id = ncm\_csr\_vlan.vlan\_inst\_id

AND audit\_name = 'NCM\_CSR\_VLAN\_AUDIT')

union

SELECT NE\_INST\_ID AS inst\_id, 'NCM\_NGMLS\_AUDIT', match\_code,

SYSDATE

FROM xng\_reports.NCM\_GI\_CI\_NGMLS\_AUDIT ncm\_ngmls

WHERE match\_code = 'BOTH'

AND NOT EXISTS (

SELECT 'x'

FROM xng\_reports.audit\_status

WHERE inst\_id = ncm\_ngmls.NE\_INST\_ID

AND audit\_name = 'NCM\_NGMLS\_AUDIT')

union

SELECT EQUIP\_INST\_ID AS inst\_id, 'NCM\_CSR\_AUDIT', match\_code,

SYSDATE

FROM xng\_reports.NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT ncm\_csr

WHERE match\_code = 'BOTH'

AND NOT EXISTS (

SELECT 'x'

FROM xng\_reports.audit\_status

WHERE inst\_id = ncm\_csr.EQUIP\_INST\_ID

AND audit\_name = 'NCM\_CSR\_AUDIT')

union

SELECT NE\_INST\_ID AS inst\_id, 'SAM\_NGMLS\_AUDIT', match\_code,

SYSDATE

FROM xng\_reports.SAM\_GI\_AL\_NGMLS\_AUDIT sam\_ngmls

WHERE match\_code = 'BOTH'

AND NOT EXISTS (

SELECT 'x'

FROM xng\_reports.audit\_status

WHERE inst\_id = sam\_ngmls.NE\_INST\_ID

AND audit\_name = 'SAM\_NGMLS\_AUDIT')

union

SELECT EQUIP\_INST\_ID AS inst\_id, 'SAM\_CSR\_AUDIT', match\_code,

SYSDATE

FROM xng\_reports.SAM\_GI\_AL\_CSR\_AUDIT sam\_csr

WHERE match\_code = 'BOTH'

AND NOT EXISTS (

SELECT 'x'

FROM xng\_reports.audit\_status

WHERE inst\_id = sam\_csr.EQUIP\_INST\_ID

AND audit\_name = 'SAM\_CSR\_AUDIT') ;

FOR rec IN modified\_staus

LOOP

-- update if it is different status with update time stamp

UPDATE xng\_reports.audit\_status ast

SET ast.update\_date = SYSDATE,

ast.match\_code = rec.match\_code

WHERE ast.inst\_id = rec.inst\_id AND ast.audit\_name = rec.audit\_name;

END LOOP;

commit;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line (SUBSTR (SQLERRM, 1, 150));

END update\_audit\_status;

END AUDIT\_STATUS\_PKG;

/

--------------------------------------------------------

-- DDL for Package Body CIRC\_SEG\_NEW\_DECOM\_RPT\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_RPT\_PKG"

AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: CIRC\_SEG\_NEW\_DECOM\_RPT\_PKG

PURPOSE:

REVISIONS:

Ver Date Author Description

------ ---------- ---------- ------------------------------------

1.0 7/15/2013 c0hedav 1. Created this package body.

1.1 7/19/2013 c0hedav 2. add 6 new fields per request into table CIRC\_SEG\_NEW\_DECOM\_DETAIL :

SEGMENT\_TYPE, CHG\_BY, DISCONNECT\_PON, DISCONNECT\_ORDER\_NUM, DISCONNECT\_ORDER\_TS, DISCONNECT\_TS

1.2 8/5/2013 c0hedav 3. add BILING\_CODE,

1.3 8/7/2013 c0hedav 4. exclude duplicated audit log for both NEW and DECOM

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE RUN\_DAILY\_DETAIL\_RPT

IS

v\_id NUMBER;

v\_cutoff\_time\_last DATE;

v\_cutoff\_time\_new DATE;

v\_group\_name VARCHAR2 (30);

BEGIN

SELECT XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN\_SEQ.NEXTVAL

INTO v\_id

FROM DUAL;

BEGIN

SELECT MAX (PROC\_START\_TS)

INTO v\_cutoff\_time\_last

FROM CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

SELECT TO\_DATE ('01-01-1970', 'MM-DD-YYYY')

INTO v\_cutoff\_time\_last

FROM DUAL;

END;

SELECT SYSDATE INTO v\_cutoff\_time\_new FROM DUAL;

INSERT INTO CIRC\_SEG\_NEW\_DECOM\_DETAIL (RPT\_ID,

DOMAIN\_INST\_ID,

DOMAIN,

AREA,

REGION,

MARKET,

CIRC\_INST\_ID,

SEGMENT\_NAME, --circ\_hum\_id

SEGMENT\_TYPE,

Z\_SITE\_ID,

SITE\_NAME, --site\_hum\_id

VENDOR,

BILLING\_CODE,

STATUS,

ACTION\_TYPE,

CHG\_TS,

CHG\_BY,

BANDWIDTH,

MRC -- recur\_costs

)

WITH INST\_AUDIT\_UPDATED

AS (SELECT a.inst\_id,

a.element\_type,

a.chg\_ts,

a.chg\_by,

a.edit\_operation,

a.version\_chgd

FROM VZWNET.INST\_AUDIT a

WHERE a.chg\_ts >= v\_cutoff\_time\_last

AND a.chg\_ts < v\_cutoff\_time\_new

AND a.element\_type = 'S'

AND a.edit\_operation = 'U'),

INST\_UPDATE\_LOG\_DECOM

AS (SELECT u.inst\_id, u.element\_type, u.version\_chgd

FROM VZWNET.INST\_UPDATE\_LOG u

WHERE u.field\_chgd = 'Status'

AND u.new\_value = 'Decommissioned'),

AUDIT\_DECOM\_LOG

AS (SELECT a.inst\_id, a.chg\_ts, chg\_by

FROM INST\_AUDIT\_UPDATED a, INST\_UPDATE\_LOG\_DECOM u

WHERE u.inst\_id = a.inst\_id

AND u.element\_type = a.element\_type

AND u.version\_chgd = a.version\_chgd),

AUDIT\_DECOM

AS (SELECT a.inst\_id, a.chg\_ts, chg\_by

FROM AUDIT\_DECOM\_LOG a

WHERE a.chg\_ts = (SELECT MAX (chg\_ts)

FROM AUDIT\_DECOM\_LOG b

WHERE a.inst\_id = b.inst\_id)),

AUDIT\_INSERT

AS (SELECT a.inst\_id, a.chg\_ts, chg\_by

FROM VZWNET.INST\_AUDIT a

WHERE a.chg\_ts >= v\_cutoff\_time\_last

AND a.chg\_ts < v\_cutoff\_time\_new

AND a.element\_type = 'S'

AND a.edit\_operation = 'I'),

AUDIT\_INSERT\_MAX

AS (SELECT a.inst\_id, a.chg\_ts, chg\_by

FROM AUDIT\_INSERT a

WHERE a.chg\_ts = (SELECT MAX (chg\_ts)

FROM AUDIT\_INSERT b

WHERE a.inst\_id = b.inst\_id)),

AUDIT\_INSERT\_DECOM

AS (SELECT inst\_id,

chg\_ts,

chg\_by,

'NEW' AS action

FROM AUDIT\_INSERT\_MAX

UNION

SELECT inst\_id,

chg\_ts,

chg\_by,

'DECOM' AS action

FROM AUDIT\_DECOM),

CIRC\_INST\_ALL

AS (SELECT circ\_inst\_id,

TYPE AS segment\_type,

circ\_hum\_id,

z\_site\_id,

vendor,

billing\_code,

status,

bandwidth,

recur\_costs

FROM VZWNET.DEL\_CIRC\_INST

UNION

SELECT circ\_inst\_id,

TYPE AS segment\_type,

circ\_hum\_id,

z\_site\_id,

vendor,

billing\_code,

status,

bandwidth,

recur\_costs

FROM VZWNET.CIRC\_INST)

SELECT v\_id,

rpt.domain\_inst\_id,

rpt.domain\_name,

rpt.area,

rpt.region,

rpt.market,

circ.circ\_inst\_id,

circ.circ\_hum\_id,

circ.segment\_type,

circ.z\_site\_id,

z.site\_hum\_id,

circ.vendor,

circ.billing\_code,

circ.status,

aud.action,

aud.chg\_ts,

aud.chg\_by,

circ.bandwidth,

circ.recur\_costs

FROM AUDIT\_INSERT\_DECOM aud,

CIRC\_INST\_ALL circ,

VZWNET.CIRC\_DOMAIN\_MAP dm,

XNG\_REPORTS.DOMAINS\_LEAF\_REPORTING rpt,

VZWNET.SITE\_INST z -- site is for detail only

WHERE aud.inst\_id = circ.circ\_inst\_id

AND circ.circ\_inst\_id = dm.circ\_inst\_id

AND dm.domain\_inst\_id = rpt.domain\_inst\_id

AND circ.z\_site\_id = z.site\_inst\_id;

-- Update 4 fields related to Disconnect

v\_group\_name := 'Disconnect Information';

DELETE FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DETAIL a

WHERE a.chg\_ts <>

(SELECT MAX (chg\_ts)

FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DETAIL b

WHERE a.domain\_inst\_id = b.domain\_inst\_id

AND a.circ\_inst\_id = b.circ\_inst\_id

AND a.action\_type = b.action\_type);

UPDATE XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DETAIL c

SET DISCONNECT\_PON =

(SELECT ATTR\_VALUE

FROM VZWNET.CIRC\_ATTR\_SETTINGS v, VZWNET.VAL\_ATTR\_NAME n

WHERE c.CIRC\_INST\_ID = v.CIRC\_INST\_ID

AND v.VAL\_ATTR\_INST\_ID = n.VAL\_ATTR\_INST\_ID

AND n.GROUP\_NAME = v\_group\_name

AND n.ATTR\_NAME = 'Disconnect PON')

WHERE CHG\_TS >= v\_cutoff\_time\_last AND CHG\_TS < v\_cutoff\_time\_new;

UPDATE XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DETAIL c

SET DISCONNECT\_ORDER\_TS =

(SELECT ATTR\_VALUE

FROM VZWNET.CIRC\_ATTR\_SETTINGS v, VZWNET.VAL\_ATTR\_NAME n

WHERE c.CIRC\_INST\_ID = v.CIRC\_INST\_ID

AND v.VAL\_ATTR\_INST\_ID = n.VAL\_ATTR\_INST\_ID

AND n.GROUP\_NAME = v\_group\_name

AND n.ATTR\_NAME = 'Disconnect Order Date')

WHERE CHG\_TS >= v\_cutoff\_time\_last AND CHG\_TS < v\_cutoff\_time\_new;

UPDATE XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DETAIL c

SET DISCONNECT\_ORDER\_NUM =

(SELECT ATTR\_VALUE

FROM VZWNET.CIRC\_ATTR\_SETTINGS v, VZWNET.VAL\_ATTR\_NAME n

WHERE c.CIRC\_INST\_ID = v.CIRC\_INST\_ID

AND v.VAL\_ATTR\_INST\_ID = n.VAL\_ATTR\_INST\_ID

AND n.GROUP\_NAME = v\_group\_name

AND n.ATTR\_NAME = 'Disconnect Order Number')

WHERE CHG\_TS >= v\_cutoff\_time\_last AND CHG\_TS < v\_cutoff\_time\_new;

UPDATE XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DETAIL c

SET DISCONNECT\_TS =

(SELECT ATTR\_VALUE

FROM VZWNET.CIRC\_ATTR\_SETTINGS v, VZWNET.VAL\_ATTR\_NAME n

WHERE c.CIRC\_INST\_ID = v.CIRC\_INST\_ID

AND v.VAL\_ATTR\_INST\_ID = n.VAL\_ATTR\_INST\_ID

AND n.GROUP\_NAME = v\_group\_name

AND n.ATTR\_NAME = 'Disconnect Date')

WHERE CHG\_TS >= v\_cutoff\_time\_last AND CHG\_TS < v\_cutoff\_time\_new;

UPDATE CIRC\_SEG\_NEW\_DECOM\_DETAIL D

SET D.SEGMENT\_NAME =

(SELECT CI.CIRC\_HUM\_ID

FROM VZWNET.CIRC\_INST CI

WHERE D.CIRC\_INST\_ID = CI.CIRC\_INST\_ID)

WHERE (D.SEGMENT\_NAME IS NULL)

OR D.SEGMENT\_NAME != (SELECT CI.CIRC\_HUM\_ID

FROM VZWNET.CIRC\_INST CI

WHERE D.CIRC\_INST\_ID = CI.CIRC\_INST\_ID);

UPDATE CIRC\_SEG\_NEW\_DECOM\_DETAIL D

SET D.MRC =

(SELECT CI.RECUR\_COSTS

FROM VZWNET.CIRC\_INST CI

WHERE D.CIRC\_INST\_ID = CI.CIRC\_INST\_ID)

WHERE (D.MRC IS NULL)

OR D.MRC != (SELECT CI.RECUR\_COSTS

FROM VZWNET.CIRC\_INST CI

WHERE D.CIRC\_INST\_ID = CI.CIRC\_INST\_ID);

RUN\_ALL\_MONTHLY\_WEEKLY\_RPT (); --as we deleted some duplicated, so need recalculate all count in summary

--RUN\_ONE\_MONTHLY\_RPT (ADD\_MONTHS (v\_cutoff\_time\_new, 1)); --current month=next mon's previous mon

--RUN\_ONE\_WEEKLY\_RPT (v\_cutoff\_time\_new + 7); --current week=next week's previous week

INSERT

INTO CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN (RPT\_ID, PROC\_START\_TS, PROC\_END\_TS)

VALUES (v\_id, v\_cutoff\_time\_new, SYSDATE);

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

ROLLBACK;

END;

PROCEDURE RUN\_ONE\_MONTHLY\_RPT (p\_run\_date DATE)

IS

v\_first\_day\_this\_month DATE;

v\_first\_day\_last\_month DATE;

v\_last\_month VARCHAR2 (20);

v\_run\_date DATE;

BEGIN

v\_run\_date := p\_run\_date;

IF (v\_run\_date IS NULL)

THEN

SELECT SYSDATE INTO v\_run\_date FROM DUAL;

END IF;

--first day of the month contains v\_run\_date

SELECT TO\_DATE (TO\_CHAR (v\_run\_date, 'yyyymm'), 'yyyymm')

INTO v\_first\_day\_this\_month --at 00hr:00mm:00sec

FROM DUAL;

--first day of the previous month

SELECT TO\_DATE (TO\_CHAR (v\_first\_day\_this\_month - 1, 'yyyymm'),

'yyyymm')

INTO v\_first\_day\_last\_month --at 00hr:00mm:00sec

FROM DUAL;

SELECT TO\_CHAR (v\_first\_day\_last\_month, 'Mon, yyyy')

INTO v\_last\_month

FROM DUAL;

DELETE FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_SUMMARY

WHERE PERIOD = v\_last\_month AND PERIOD\_TYPE = 'MONTHLY';

INSERT INTO XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_SUMMARY (PERIOD,

PERIOD\_TYPE,

AREA,

REGION,

COUNT\_DECOM,

COUNT\_NEW,

COUNT\_NEW\_TOTAL,

COUNT\_NEW\_BY\_TESS,

COUNT\_NEW\_NOT\_TESS\_CONTRACT,

COUNT\_NEW\_NOT\_TESS\_NO\_CONTRACT,

PERCENTAGE\_NEW\_BY\_TESS

)

WITH MONTHLY

AS (SELECT \*

FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DET\_TESS\_VW

WHERE CHG\_TS >= v\_first\_day\_last\_month

AND CHG\_TS < v\_first\_day\_this\_month),

MAX\_CHG\_TS

AS (SELECT \*

FROM MONTHLY a

WHERE a.CHG\_TS =

(SELECT MAX (CHG\_TS)

FROM MONTHLY b

WHERE a.CIRC\_INST\_ID = b.CIRC\_INST\_ID

AND a.ACTION\_TYPE = b.ACTION\_TYPE))

SELECT v\_last\_month,

'MONTHLY',

area,

region,

SUM (DECOM) DECOM\_COUNT,

SUM (NEW) COUNT\_NEW,

SUM (NEW\_TOTAL) COUNT\_NEW\_TOTAL,

SUM (NEW\_BY\_TESS) COUNT\_NEW\_BY\_TESS,

SUM (NEW\_NOT\_TESS\_CONTRACT) COUNT\_NEW\_NOT\_TESS\_CONTRACT,

SUM (NEW\_NOT\_TESS\_NO\_CONTRACT) COUNT\_NEW\_NOT\_TESS\_NO\_CONTRACT,

ROUND (

CASE

WHEN SUM (NEW\_TOTAL) > 0

THEN

SUM (NEW\_BY\_TESS) \* 100.0 / SUM (NEW\_TOTAL)

ELSE

0.00

END,

2)

PERCENTAGE\_NEW\_BY\_TESS

FROM MAX\_CHG\_TS

GROUP BY area, region

ORDER BY area, region;

END;

PROCEDURE RUN\_ONE\_WEEKLY\_RPT (p\_run\_date DATE)

IS

v\_first\_day\_this\_week DATE;

v\_last\_day\_last\_week DATE;

v\_first\_day\_last\_week DATE;

v\_run\_date DATE;

v\_period VARCHAR2 (50);

BEGIN

v\_run\_date := p\_run\_date;

IF (v\_run\_date IS NULL)

THEN

SELECT SYSDATE INTO v\_run\_date FROM DUAL;

END IF;

--Sunday after v\_run\_date

SELECT TRUNC (NEXT\_DAY (v\_run\_date - 7, 'SUN'))

INTO v\_first\_day\_this\_week

FROM DUAL;

--Saturday of the week contain v\_run\_date

v\_last\_day\_last\_week := v\_first\_day\_this\_week - 1;

--Sunday of the week contain v\_run\_date

SELECT TRUNC (NEXT\_DAY (v\_run\_date - 14, 'SUN'))

INTO v\_first\_day\_last\_week

FROM DUAL;

SELECT TO\_CHAR (v\_first\_day\_last\_week, 'mm/dd/yyyy')

|| ' - '

|| TO\_CHAR (v\_last\_day\_last\_week, 'mm/dd/yyyy')

INTO v\_period

FROM DUAL;

DELETE FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_SUMMARY

WHERE PERIOD = v\_period AND PERIOD\_TYPE = 'WEEKLY';

INSERT INTO XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_SUMMARY (PERIOD,

PERIOD\_TYPE,

AREA,

REGION,

COUNT\_DECOM,

COUNT\_NEW,

COUNT\_NEW\_TOTAL,

COUNT\_NEW\_BY\_TESS,

COUNT\_NEW\_NOT\_TESS\_CONTRACT,

COUNT\_NEW\_NOT\_TESS\_NO\_CONTRACT,

PERCENTAGE\_NEW\_BY\_TESS

)

WITH WEEKLY

AS (SELECT \*

FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DET\_TESS\_VW

WHERE CHG\_TS >= v\_first\_day\_last\_week

AND CHG\_TS < v\_first\_day\_this\_week),

MAX\_CHG\_TS

AS (SELECT \*

FROM WEEKLY a

WHERE a.chg\_ts =

(SELECT MAX (chg\_ts)

FROM WEEKLY b

WHERE a.CIRC\_INST\_ID = b.CIRC\_INST\_ID

AND a.ACTION\_TYPE = b.ACTION\_TYPE))

SELECT v\_period,

'WEEKLY',

area,

region,

SUM (DECOM) DECOM\_COUNT,

SUM (NEW) COUNT\_NEW,

SUM (NEW\_TOTAL) COUNT\_NEW\_TOTAL,

SUM (NEW\_BY\_TESS) COUNT\_NEW\_BY\_TESS,

SUM (NEW\_NOT\_TESS\_CONTRACT) COUNT\_NEW\_NOT\_TESS\_CONTRACT,

SUM (NEW\_NOT\_TESS\_NO\_CONTRACT) COUNT\_NEW\_NOT\_TESS\_NO\_CONTRACT,

ROUND (

CASE

WHEN SUM (NEW\_TOTAL) > 0

THEN

SUM (NEW\_BY\_TESS) \* 100.0 / SUM (NEW\_TOTAL)

ELSE

0.00

END,

2)

PERCENTAGE\_NEW\_BY\_TESS

FROM MAX\_CHG\_TS

GROUP BY area, region

ORDER BY area, region;

END;

PROCEDURE RUN\_ALL\_MONTHLY\_WEEKLY\_RPT

IS

v\_min\_time DATE;

v\_max\_time DATE;

v\_run\_time DATE;

v\_end\_time DATE;

BEGIN

SELECT MIN (CHG\_TS) INTO v\_min\_time FROM CIRC\_SEG\_NEW\_DECOM\_DETAIL;

SELECT MAX (CHG\_TS) INTO v\_max\_time FROM CIRC\_SEG\_NEW\_DECOM\_DETAIL;

--clean table

EXECUTE IMMEDIATE

'TRUNCATE TABLE XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_SUMMARY';

--run monthly for all

SELECT TO\_DATE (TO\_CHAR (v\_min\_time, 'yyyymm'), 'yyyymm')

INTO v\_run\_time -- 1st day of the month at 00hr:00mm:00sec for earliest record

FROM DUAL;

SELECT TO\_DATE (TO\_CHAR (v\_max\_time, 'yyyymm'), 'yyyymm')

INTO v\_end\_time -- 1st day of the month at 00hr:00mm:00sec for latest record

FROM DUAL;

SELECT ADD\_MONTHS (v\_end\_time, 1) INTO v\_end\_time FROM DUAL; --1st day of next month

/\*

IF (v\_end\_time > SYSDATE)

THEN

SELECT ADD\_MONTHS (v\_end\_time, -1) INTO v\_end\_time FROM DUAL; -- roll back to 1st day of this month, to avoid partial monthly report

END IF;

\*/

WHILE (v\_run\_time <= v\_end\_time)

LOOP

BEGIN

RUN\_ONE\_MONTHLY\_RPT (v\_run\_time);

SELECT ADD\_MONTHS (v\_run\_time, 1) INTO v\_run\_time FROM DUAL;

END;

END LOOP;

--run weekly for all

--Sunday after the earliest record

SELECT TRUNC (NEXT\_DAY (v\_min\_time, 'SUN')) INTO v\_run\_time FROM DUAL;

--Sunday after the latest record

SELECT TRUNC (NEXT\_DAY (v\_max\_time, 'SUN')) INTO v\_end\_time FROM DUAL;

/\*

IF (v\_end\_time > SYSDATE)

THEN -- last record is in this week which is in partial week

v\_end\_time := v\_end\_time - 7;

END IF;

\*/

WHILE (v\_run\_time <= v\_end\_time)

LOOP

BEGIN

RUN\_ONE\_WEEKLY\_RPT (v\_run\_time);

v\_run\_time := v\_run\_time + 7;

END;

END LOOP;

--run yearly for all

--first day(mid-night) of the year earlier than v\_min\_time

SELECT TRUNC(TO\_DATE ('01/01'||TO\_CHAR (v\_min\_time, 'yyyy'), 'mm/dd/yyyy'))

INTO v\_run\_time

FROM DUAL;

--first day(mid-night) of the next year after v\_max\_time

SELECT TRUNC(TO\_DATE ('12/31'||TO\_CHAR (v\_max\_time, 'yyyy'), 'mm/dd/yyyy')) + 1

INTO v\_end\_time

FROM DUAL;

WHILE (v\_run\_time <= v\_end\_time)

LOOP

BEGIN

RUN\_ONE\_YEARLY\_RPT (v\_run\_time);

-- add one year

SELECT TRUNC(TO\_DATE ('12/31'||TO\_CHAR (v\_run\_time, 'yyyy'), 'mm/dd/yyyy')) + 1

INTO v\_run\_time

FROM DUAL;

END;

END LOOP;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

ROLLBACK;

END;

PROCEDURE RUN\_ONE\_YEARLY\_RPT (p\_run\_date DATE)

IS

v\_first\_day\_this\_year DATE;

v\_first\_day\_next\_year DATE;

v\_this\_year VARCHAR2 (20);

v\_run\_date DATE;

BEGIN

v\_run\_date := p\_run\_date;

IF (v\_run\_date IS NULL)

THEN

SELECT SYSDATE INTO v\_run\_date FROM DUAL;

END IF;

--first day(mid-night) of the year contains v\_run\_date

SELECT TRUNC(TO\_DATE ('01/01'||TO\_CHAR (v\_run\_date, 'yyyy'), 'mm/dd/yyyy'))

INTO v\_first\_day\_this\_year

FROM DUAL;

-- this year

SELECT TO\_CHAR (v\_first\_day\_this\_year, 'yyyy')

INTO v\_this\_year

FROM DUAL;

--first day(mid-night) of the next year

SELECT TRUNC(TO\_DATE ('12/31'||v\_this\_year, 'mm/dd/yyyy')) + 1

INTO v\_first\_day\_next\_year

FROM DUAL;

DELETE FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_SUMMARY

WHERE PERIOD = v\_this\_year AND PERIOD\_TYPE = 'YEARLY';

INSERT INTO XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_SUMMARY (PERIOD,

PERIOD\_TYPE,

AREA,

REGION,

COUNT\_DECOM,

COUNT\_NEW,

COUNT\_NEW\_TOTAL,

COUNT\_NEW\_BY\_TESS,

COUNT\_NEW\_NOT\_TESS\_CONTRACT,

COUNT\_NEW\_NOT\_TESS\_NO\_CONTRACT,

PERCENTAGE\_NEW\_BY\_TESS

)

WITH YEARLY

AS (SELECT \*

FROM XNG\_REPORTS.CIRC\_SEG\_NEW\_DECOM\_DET\_TESS\_VW

WHERE CHG\_TS >= v\_first\_day\_this\_year

AND CHG\_TS < v\_first\_day\_next\_year),

MAX\_CHG\_TS

AS (SELECT \*

FROM YEARLY a

WHERE a.CHG\_TS =

(SELECT MAX (CHG\_TS)

FROM YEARLY b

WHERE a.CIRC\_INST\_ID = b.CIRC\_INST\_ID

AND a.ACTION\_TYPE = b.ACTION\_TYPE))

SELECT v\_this\_year,

'YEARLY',

area,

region,

SUM (DECOM) DECOM\_COUNT,

SUM (NEW) COUNT\_NEW,

SUM (NEW\_TOTAL) COUNT\_NEW\_TOTAL,

SUM (NEW\_BY\_TESS) COUNT\_NEW\_BY\_TESS,

SUM (NEW\_NOT\_TESS\_CONTRACT) COUNT\_NEW\_NOT\_TESS\_CONTRACT,

SUM (NEW\_NOT\_TESS\_NO\_CONTRACT) COUNT\_NEW\_NOT\_TESS\_NO\_CONTRACT,

ROUND (

CASE

WHEN SUM (NEW\_TOTAL) > 0

THEN

SUM (NEW\_BY\_TESS) \* 100.0 / SUM (NEW\_TOTAL)

ELSE

0.00

END,

2)

PERCENTAGE\_NEW\_BY\_TESS

FROM MAX\_CHG\_TS

GROUP BY area, region

ORDER BY area, region;

END;

END;

/

--------------------------------------------------------

-- DDL for Package Body COMPLETE\_EBH\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."COMPLETE\_EBH\_AUDIT"

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--- Foundation of CSR Audit

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure load\_ne\_vs\_Xng\_csr\_audit is

sql\_stmt varchar2(32000);

begin

sql\_stmt:='drop table HPOV\_VS\_XNG\_CSR\_AUDIT\_WK';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error truncating the table: HPOV\_VS\_XNG\_CSR\_AUDIT\_WK '

|| TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

sql\_stmt:='TRUNCATE table CSR\_AUDIT\_ISSUE\_WK';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error truncating the table: CSR\_AUDIT\_ISSUE\_WK '

|| TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

-- create a full audit of CSR HPOV vs Xng

sql\_stmt := 'create table HPOV\_VS\_XNG\_CSR\_AUDIT\_WK as

WITH

XNG\_CSR as

(

select csr\_device\_name, ei.descr, ei.status, ei.site\_inst\_id, ei.equip\_inst\_id

, case when status = ''Live'' then ''Y''

else ''N''

end Live\_in\_xng

, parse\_status

from vzwnet.equip\_inst ei

join XNG\_CSR\_PARSED\_WK csr

on csr.EQUIP\_INST\_ID = ei.equip\_inst\_id

)

select ne.csr\_id, ne.csr\_device\_name ne\_csr\_device\_name

, case when ne.csr\_device\_name is null then ''Xng Only''

when ei.descr is null then ''NE Only''

else ''BOTH''

end match\_code

, parse\_status match\_status

, ei.csr\_device\_name xng\_csr\_device\_name

, ei.descr

, live\_in\_xng, equip\_inst\_id, ne.csr\_ip

from CSR\_DEVICES\_WK ne

full outer join XNG\_CSR ei on upper(ne.csr\_device\_name) = upper(ei.csr\_device\_name)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

update\_csr\_mis\_match;

BEGIN

sql\_stmt:='CREATE INDEX NE\_VS\_XNG\_CSR\_WK\_1\_IDA ON HPOV\_VS\_XNG\_CSR\_AUDIT\_WK(XNG\_CSR\_DEVICE\_NAME)';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='CREATE INDEX NE\_VS\_XNG\_CSR\_WK\_2\_IDX ON HPOV\_VS\_XNG\_CSR\_AUDIT\_WK(NE\_CSR\_DEVICE\_NAME)';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='CREATE INDEX NE\_VS\_XNG\_CSR\_WK\_3\_IDX ON HPOV\_VS\_XNG\_CSR\_AUDIT\_WK(MATCH\_CODE, MATCH\_STATUS)';

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(substr(sql\_stmt, 1, 255));

dbms\_output.put\_line(SubStr('Error creating index on the table: '

||'HPOV\_VS\_XNG\_CSR\_AUDIT\_WK '|| TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

sql\_stmt:='insert into CSR\_AUDIT\_ISSUE\_WK columns (csr\_id, xng\_equip\_inst\_id, csr\_issue\_id)

WITH

DUPS as

(

select ei.CSR\_ID, 1 issue\_id -- Duplicate CSR matches in Xng

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK ei

where csr\_id is not null

group by ei.csr\_id

having count(1) > 1

)

select d.csr\_id, aud.equip\_inst\_id, d.issue\_id

from DUPS d

join HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

on aud.csr\_id = d.csr\_id

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into CSR\_AUDIT\_ISSUE\_WK columns (csr\_id, xng\_equip\_inst\_id, csr\_issue\_id)

WITH

DUPS as

(

SELECT CD.CSR\_DEVICE\_NAME, 10 issue\_id -- Duplicate CSR matches in NE

FROM CSR\_DEVICES\_WK cd

WHERE CD.CSR\_ID IS NOT NULL AND CD.CSR\_DEVICE\_NAME IS NOT NULL

GROUP BY CD.CSR\_DEVICE\_NAME

HAVING COUNT(1) > 1

)

select distinct aud.csr\_id, aud.equip\_inst\_id, d.issue\_id

from DUPS d

join HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

on aud.NE\_CSR\_DEVICE\_NAME = d.CSR\_DEVICE\_NAME

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- if the equipment does not matches NE and equip name does not match standard then insert it in

sql\_stmt:='insert into CSR\_AUDIT\_ISSUE\_WK columns (xng\_equip\_inst\_id, csr\_issue\_id)

select distinct ei.EQUIP\_INST\_ID, 9 -- Xng Only and equipment name in Xng does not match standard

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK ei

where match\_code = ''Xng Only''

and match\_status = ''Xng CSR Name does not match Standard''

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- if the equipment matches NE but equip name does not match standard then update that to be 'NE Only'

sql\_stmt:='insert into CSR\_AUDIT\_ISSUE\_WK columns (csr\_id, xng\_EQUIP\_INST\_ID, csr\_issue\_id)

select distinct ei.CSR\_ID, ei.EQUIP\_INST\_ID, 7 -- equipment name in Xng does not match standard

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK ei

where match\_code = ''BOTH''

and match\_status = ''Xng CSR Name does not match Standard''

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- We might have discovered some CLLI that is not in the list of MTSO CLLI

-- mark those as bad

-- We might have discovered some CLLI where 1st 6-char match MTSO CLLI but all 8 dont

-- mark those as bad

-- some times there are 2 legitimate cllis for the same site

-- in those cases the o/p comes as followd

-- clli\_1\_csr clli\_2\_vsm CLLI Mismatch

-- clli\_2\_csr clli\_1\_vsm CLLI Mismatch

-- so reverse the order and subtract it. That will give only the ones that are

-- truly bad

sql\_stmt:='insert into CSR\_AUDIT\_ISSUE\_WK columns (csr\_id, xng\_equip\_inst\_id, csr\_issue\_id)

WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map\_v cdm

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.csr\_id, ei.csr\_device\_name,

SUBSTR (ei.csr\_device\_name, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM CSR\_DEVICES\_WK ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.csr\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map\_v cdm

WHERE SUBSTR (ei.csr\_device\_name, 1, 8) = cdm.clli)),

BAD\_CLLI AS

(

SELECT ei.csr\_id, ei.csr\_device\_name

FROM CSR\_DEVICES\_WK ei

MINUS

SELECT ei.csr\_id, ei.csr\_device\_name

FROM CSR\_DEVICES\_WK ei JOIN clli\_domain\_map\_v cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.csr\_device\_name, 1, 8)

MINUS

SELECT ei.csr\_id, ei.csr\_device\_name

FROM CSR\_DEVICES\_WK ei JOIN clli\_domain\_map\_v cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.csr\_device\_name, 1, 6)

),

ISSUES AS

(

SELECT cm.csr\_id, 3 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.csr\_id, 2 issue\_id

FROM bad\_clli bc

)

SELECT aud.csr\_id, aud.equip\_inst\_id, i.issue\_id

FROM ISSUES i

join HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

on aud.csr\_id = i.csr\_id

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done inserting into CSR\_AUDIT\_ISSUE\_WK');

sql\_stmt := 'update HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud set match\_code=''NE Only''

where exists (

select 1 from(

select distinct ne.CSR\_ID from CSR\_AUDIT\_ISSUE\_WK csr,

HPOV\_VS\_XNG\_CSR\_AUDIT\_WK ne,

CSR\_ISSUE ci

where csr.CSR\_ISSUE\_ID in (1, 2, 3, 7)

and csr.csr\_issue\_id = ci.csr\_issue\_id

and ci.is\_critical = ''Y''

and ne.CSR\_ID=csr.CSR\_ID

)csr

where aud.csr\_id=csr.csr\_id

)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done updating match\_code to ''NE Only''');

sql\_stmt := 'update HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud set match\_status =

NVL2(MATCH\_STATUS,MATCH\_STATUS||'', Duplicate in NE'',''Duplicate in NE'')

where exists (

select 1 from(

select distinct ne.CSR\_ID from CSR\_AUDIT\_ISSUE\_WK csr,

HPOV\_VS\_XNG\_CSR\_AUDIT\_WK ne,

CSR\_ISSUE ci

where csr.CSR\_ISSUE\_ID=10 -- duplicate NE

and csr.csr\_issue\_id = ci.csr\_issue\_id

and ci.is\_critical = ''Y''

and ne.CSR\_ID=csr.CSR\_ID

)csr

where aud.csr\_id=csr.csr\_id

)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done updating match\_status to ''Duplicate in NE''');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_ne\_vs\_Xng\_csr\_audit(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_ne\_vs\_Xng\_csr\_audit;

PROCEDURE update\_csr\_mis\_match

IS

processname VARCHAR2 (30) := 'CSR\_AUDIT';

updstmt VARCHAR2 (32767);

delstmt VARCHAR2 (32767);

sql\_stmt VARCHAR2 (32767);

MESSAGE VARCHAR2 (500);

cursor csr\_mis\_match is

WITH ne\_only AS

(SELECT \*

FROM hpov\_vs\_xng\_csr\_audit\_wk

WHERE match\_code = 'NE Only' AND xng\_csr\_device\_name IS NULL),

xng\_only AS

(SELECT \*

FROM hpov\_vs\_xng\_csr\_audit\_wk

WHERE match\_code = 'Xng Only' AND ne\_csr\_device\_name IS NULL)

SELECT ne\_only.ne\_csr\_device\_name,ne\_only.match\_code, 'Name Mismatch' as match\_status,xng\_only.xng\_csr\_device\_name,xng\_only.descr,xng\_only.live\_in\_xng, xng\_only.equip\_inst\_id,ne\_only.csr\_ip

FROM ne\_only,xng\_only

WHERE substr (ne\_only.ne\_csr\_device\_name,

0,length (ne\_only.ne\_csr\_device\_name) - 2

)||substr (ne\_only.ne\_csr\_device\_name,

length (ne\_only.ne\_csr\_device\_name)

) =

substr (xng\_only.xng\_csr\_device\_name,

0,length (xng\_only.xng\_csr\_device\_name) - 2

) || substr (xng\_only.xng\_csr\_device\_name,

length (xng\_only.xng\_csr\_device\_name)

);

begin

updstmt :=

'update hpov\_vs\_xng\_csr\_audit\_wk

set xng\_csr\_device\_name = :xng\_csr\_device\_name ,

descr = :descr ,

live\_in\_xng = :live\_in\_xng ,

equip\_inst\_id = :equip\_inst\_id ,

match\_status = :match\_status

where match\_code = ''NE Only'' and ne\_csr\_device\_name = :ne\_csr\_device\_name'

;

delstmt :=

'delete hpov\_vs\_xng\_csr\_audit\_wk

where xng\_csr\_device\_name = :xng\_csr\_device\_name

and match\_code = ''Xng Only''';

sql\_stmt:='insert into CSR\_AUDIT\_ISSUE\_WK columns (csr\_id, xng\_EQUIP\_INST\_ID, csr\_issue\_id)

select distinct ei.CSR\_ID, ei.EQUIP\_INST\_ID, 32 -- Equipment exists but Name Change on NE/Xng.

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK ei

where match\_code = ''NE Only''

and match\_status = ''Name Mismatch''';

FOR rec IN csr\_mis\_match

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.xng\_csr\_device\_name, rec.descr, rec.live\_in\_xng, rec.equip\_inst\_id, rec.match\_status, rec.ne\_csr\_device\_name;

EXECUTE IMMEDIATE delstmt

USING rec.xng\_csr\_device\_name;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| 'update\_csr\_mis\_match'

|| '(): Can''t update '

|| rec.ne\_csr\_device\_name

|| ' for CSR-extract in table: hpov\_vs\_xng\_csr\_audit\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

EXECUTE IMMEDIATE sql\_stmt;

commit;

End update\_csr\_mis\_match;

PROCEDURE load\_csr\_missing\_in\_hpov IS

sql\_stmt varchar2(32767);

BEGIN

execute immediate ('truncate table csr\_missing\_in\_hpov\_wk');

sql\_stmt := 'insert into csr\_missing\_in\_hpov\_wk

WITH

REPORTING\_DOMAINS as

(

select area, region

from xng\_reports.domains\_regional\_reporting

where region not in (''NNO'', ''OSS'')

union

select distinct area, null

from xng\_reports.domains\_regional\_reporting

where region not in (''NNO'', ''OSS'')

union

select null, null

from dual

union

select ''unknown'' area, ''unknown'' region

from dual

)

,

ALL\_NE\_BY\_DOMAIN as

(

select distinct nvl(area, ''unknown'') area, nvl(region, ''unknown'') region, CSR\_DEVICE\_NAME

from csr\_devices\_wk cd

left outer join clli\_domain\_map\_v cdm

on substr(cd.CSR\_DEVICE\_NAME, 1, 6) = substr(cdm.CLLI, 1, 6)

and region not in (''NNO'', ''OSS'')

)

,

ALL\_NE\_BY\_DOMAIN\_CNT as

(

select area, region, count(CSR\_DEVICE\_NAME) tot\_ne\_csr\_per\_region

from ALL\_NE\_BY\_DOMAIN

group by rollup (area, region)

)

,

MATCHED\_IN\_XNG as

(

-- get anything in xng that matched NE

-- ignore status of csr in xng and even the multiple matches

select area, region, count(NE\_CSR\_DEVICE\_NAME) tot\_csr\_matched\_per\_region

from (

select distinct area, region, a.NE\_CSR\_DEVICE\_NAME

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK a

join ALL\_NE\_BY\_DOMAIN ne

on ne.csr\_device\_name = a.NE\_CSR\_DEVICE\_NAME

where a.EQUIP\_INST\_ID is not null

)

group by rollup (area, region)

)

,

LIVE\_XNG\_ONLY as

(

select area, region, count(DESCR) tot\_xo\_live\_csr\_per\_region

from (

select distinct area, region, a.DESCR

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK a

join vzwnet.equip\_domain\_map edm

on edm.EQUIP\_INST\_ID = a.EQUIP\_INST\_ID

join xng\_reports.domains\_leaf\_reporting dlr

on edm.domain\_inst\_id = dlr.domain\_inst\_id

and region not in (''NNO'', ''OSS'')

where match\_code = ''Xng Only''

and a.LIVE\_IN\_XNG = ''Y''

)

group by rollup (area, region)

)

select nvl(dlr.area, ''Enterprise Totals'') area, dlr.region

, tot\_ne\_csr\_per\_region

, tot\_csr\_matched\_per\_region

, nvl(tot\_xo\_live\_csr\_per\_region, 0) tot\_xo\_live\_csr\_per\_region

, round(nvl(tot\_xo\_live\_csr\_per\_region, 0)/decode(nvl(tot\_csr\_matched\_per\_region,0)+nvl(tot\_xo\_live\_csr\_per\_region, 0), 0, 1, nvl(tot\_csr\_matched\_per\_region,0)+nvl(tot\_xo\_live\_csr\_per\_region, 0)) \*100, 2)

--, round(tot\_xo\_live\_csr\_per\_region/(tot\_csr\_matched\_per\_region+tot\_xo\_live\_csr\_per\_region) \*100, 2) pct\_live\_csr\_missing\_in\_hpov

from REPORTING\_DOMAINS dlr

left outer join ALL\_NE\_BY\_DOMAIN\_CNT cd

on (cd.region = dlr.region

or (cd.area = dlr.area and cd.region is null and dlr.region is null )

or (cd.area is null and dlr.area is null )

)

left outer join MATCHED\_IN\_XNG m

on (m.region = dlr.region

or (m.area = dlr.area and m.region is null and dlr.region is null )

or (m.area is null and dlr.area is null )

)

left outer join LIVE\_XNG\_ONLY lx

on (lx.region = dlr.region

or (lx.area = dlr.area and lx.region is null and dlr.region is null )

or (lx.area is null and dlr.area is null )

)

order by dlr.area, dlr.region';

execute immediate sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_csr\_missing\_in\_hpov(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END load\_csr\_missing\_in\_hpov;

-- this query does not compare anything against discovered data

-- for ALL csr in Xng check if they have gige or not

-- and if it is there then is it live or not

-- \*\*\* By the end of 2011 year there will be 35K CSR \*\*\*\*

-- in that case this query will blow up

-- hence have to wrap this in an outer cursor

procedure load\_gige\_paths\_for\_all\_csrs is

sql\_stmt varchar2(32000);

cursor CSRs is

select ei.equip\_inst\_id

from vzwnet.equip\_inst ei

where ei.type = 'CSR';

cnt number := 0;

begin

-- drop the table

sql\_stmt:='truncate table XNG\_CSR\_GIGE\_PATHS\_WK';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error truncating table: XNG\_CSR\_GIGE\_PATHS\_WK '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

end;

-- insert all CSR-GigE into this table

for rec in CSRs

LOOP

load\_gige\_paths\_for\_a\_csr(rec.equip\_inst\_id);

cnt := cnt + 1;

if (cnt = 1000)

THEN

commit;

cnt := 0;

END IF;

END LOOP;

commit;

dbms\_output.put\_line('Update XNG\_CSR\_GIGE\_PATHS\_WK completed');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_gige\_paths\_for\_all\_csrs(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_gige\_paths\_for\_all\_csrs;

Procedure load\_gige\_paths\_for\_a\_csr (CSR\_equip\_inst\_id in number) is

sql\_stmt varchar2(32000);

begin

sql\_stmt:='insert into XNG\_CSR\_GIGE\_PATHS\_WK

WITH

GET\_ALL\_CSRS as (

select :y matched\_equip\_inst\_id

, ei.equip\_inst\_id, decode (ei.status, ''Live'', ''Y'', ''N'') csr\_live\_in\_xng

from XNG\_CSR\_PARSED\_WK csr, vzwnet.equip\_inst ei

where ei.EQUIP\_INST\_ID = csr.EQUIP\_INST\_ID

and csr.equip\_inst\_id = :x

)

,

GET\_CSR\_PORTS as

(

select p.site\_inst\_id, ei.\*, p.PORT\_INST\_ID, p.PORT\_HUM\_ID, p.BANDWIDTH

, p.CIRC\_PATH\_INST\_ID, cpi.BANDWIDTH path\_bw, cpi.CIRC\_PATH\_HUM\_ID path\_hum\_id

, p.NEXT\_PATH\_INST\_ID, cpi.BANDWIDTH next\_bw, cpi.CIRC\_PATH\_HUM\_ID next\_hum\_id

, decode(cpi.STATUS, ''Live'', ''Y'', ''N'') gige\_live\_in\_xng

from GET\_ALL\_CSRS ei

join vzwnet.epa p

on ei.equip\_inst\_id = p.equip\_inst\_id

left outer join vzwnet.circ\_path\_inst cpi

on cpi.CIRC\_PATH\_INST\_ID = p.CIRC\_PATH\_INST\_ID

left outer join vzwnet.circ\_path\_inst npi

on npi.CIRC\_PATH\_INST\_ID = p.Next\_PATH\_INST\_ID

where

(cpi.TYPE = ''EBH'' AND cpi.BANDWIDTH <> ''VLAN'')

or

(npi.TYPE = ''EBH'' AND npi.BANDWIDTH <> ''VLAN'')

)

,

-- get all GigE paths that have a eNB in them

GigE\_W\_eNB as

(

select csr.port\_inst\_id

from vzwnet.circ\_path\_element cpe

join GET\_CSR\_PORTS csr

on cpe.circ\_path\_inst\_id in (csr.circ\_path\_inst\_id, next\_path\_inst\_id)

join vzwnet.epa eNB

on eNB.port\_inst\_id = cpe.port\_inst\_id

join XNG\_ENB\_PARSED ei

on ei.EQUIP\_INST\_ID = eNB.equip\_inst\_id

)

-- ignore all CSR ports that are connected to eNB

select matched\_equip\_inst\_id, p.equip\_inst\_id, p.port\_inst\_id

, p.CIRC\_PATH\_INST\_ID

, p.NEXT\_PATH\_INST\_ID

, csr\_live\_in\_xng

, gige\_live\_in\_xng

from GET\_CSR\_PORTS p

where not exists (select 1

from GigE\_W\_eNB e

where e.port\_inst\_id = p.port\_inst\_id

)

'

;

EXECUTE IMMEDIATE sql\_stmt using CSR\_equip\_inst\_id, CSR\_equip\_inst\_id ;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_gige\_paths\_for\_a\_csr(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_gige\_paths\_for\_a\_csr;

-- gige (to cellsite) that does not have a seg to the cell site, then we check

-- if the parent has segment, it is stored as optical BW

--------------------- NSP --------

-- All parents (1st leg)of VLANs (1st leg, including gige and its parents) are stored in xng\_vlan\_parents

-- this method should be modified to NOT to do that part again. rest of the functionality can be kept as needed

--

procedure load\_gige\_on\_higher\_bw (truncate\_tab in varchar2 default 'y') is

sql\_stmt varchar2(32000);

-- giges no seg, if csr is in mtso, then exempt

cursor GIGE\_MISSING\_SEG is

select distinct

case when cpi.circ\_path\_inst\_id is null then cpi.next\_path\_inst\_id

else cpi.circ\_path\_inst\_id

end gige\_path\_inst\_id

from XNG\_CSR\_GIGE\_PATHS\_WK cpi

left outer join vzwnet.circ\_path\_element cpe

on cpe.CIRC\_PATH\_INST\_ID in (cpi.CIRC\_PATH\_INST\_ID, cpi.NEXT\_PATH\_INST\_ID)

and cpe.element\_type = 'S'

where cpe.SEGMENT\_INST\_ID is null

minus

select distinct

case when cpi.circ\_path\_inst\_id is null then cpi.next\_path\_inst\_id

else cpi.circ\_path\_inst\_id

end gige\_path\_inst\_id

from

HPOV\_VS\_XNG\_CSR\_AUDIT\_WK hpov,

XNG\_CSR\_GIGE\_PATHS\_WK cpi,

vzwnet.equip\_inst ei, vzwnet.site\_inst si

where EI.SITE\_INST\_ID=SI.SITE\_INST\_ID

and SI.NUM='MTSO'

and HPOV.EQUIP\_INST\_ID=EI.EQUIP\_INST\_ID

and CPI.EQUIP\_INST\_ID = HPOV.EQUIP\_INST\_ID

and HPOV.EQUIP\_INST\_ID > 0

;

cnt number := 0;

begin

if (truncate\_tab is null or lower(truncate\_tab) = 'y') then

sql\_stmt:='truncate table XNG\_GIGE\_DELIVERY\_TABLE\_WK';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error truncating table: '

||'XNG\_GIGE\_DELIVERY\_TABLE\_WK '|| TO\_CHAR(SQLCODE)||': '

||SQLERRM, 1, 255));

raise;

end;

sql\_stmt:='truncate table XNG\_GIGE\_PARENT\_MW\_WK';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error truncating table: '

||'XNG\_GIGE\_PARENT\_MW\_WK '|| TO\_CHAR(SQLCODE)||': '

||SQLERRM, 1, 255));

raise;

end;

END IF;

for rec in GIGE\_MISSING\_SEG

LOOP

dbms\_output.put\_line(rec.gige\_path\_inst\_id);

load\_parents\_w\_seg\_per\_gige(rec.gige\_path\_inst\_id, false);

load\_parents\_w\_microwave(rec.gige\_path\_inst\_id, false);

cnt := cnt + 1;

commit;

if (cnt = 1000)

THEN

dbms\_output.put\_line('Committing');

commit;

cnt := 0;

END IF;

END LOOP;

commit;

dbms\_output.put\_line('Added GigE delivered on optical to XNG\_GIGE\_DELIVERY\_TABLE\_WK');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_gige\_on\_higher\_bw(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_gige\_on\_higher\_bw;

-- for the given gige, check if the parent has a segment and in XNG\_GIGE\_DELIVERY\_TABLE stored as OPTICAL BW

procedure load\_parents\_w\_seg\_per\_gige (gige\_path\_inst\_id in number, has\_vlan in boolean) Is

sql\_stmt\_common varchar2(32000);

sql\_stmt\_has\_vlan varchar2(32000);

sql\_stmt\_no\_vlan varchar2(32000);

sql\_stmt varchar2(32000);

begin

sql\_stmt := 'insert into XNG\_GIGE\_DELIVERY\_TABLE\_WK

WITH ALL\_PARENTS as

(

select distinct gp.gige\_inst\_id, gp.parent\_path\_inst\_id path\_inst\_id

from xng\_gige\_parents\_wk gp

where gp.gige\_inst\_id = :x

),

PARENTS\_OF\_GIGE\_W\_SEG as

(

-- now get only those parents of GigE that have a segment in them

select pp.gige\_inst\_id, pp.PATH\_INST\_ID, cpe.segment\_inst\_id, cpe.sequence seg\_order

from ALL\_PARENTS pp

join vzwnet.circ\_path\_element cpe

on cpe.CIRC\_PATH\_INST\_ID = pp.path\_inst\_id

and cpe.segment\_inst\_id is not null

)

select distinct gigE.GIGE\_INST\_ID, gige.segment\_inst\_id, --gigE.path\_inst\_id,

case when upper(ci.BANDWIDTH) like ''OC%'' then ''OPTICAL BW''

--when upper(ci.BANDWIDTH) like ''%MBPS'' then ''MICROWAVE BW''

else ci.BANDWIDTH

end delivered\_on,

seg\_order

from PARENTS\_OF\_GIGE\_W\_SEG gige,

vzwnet.circ\_path\_inst gpi,

vzwnet.circ\_path\_inst cpi,

XNG\_CSR\_GIGE\_PATHS\_WK csr,

vzwnet.equip\_inst ei,

vzwnet.circ\_inst ci

where

gpi.circ\_PATH\_INST\_ID = gige.path\_inst\_id

and cpi.circ\_PATH\_INST\_ID = gige.gige\_inst\_id

and gige.gige\_inst\_id in (csr.circ\_PATH\_INST\_ID , csr.NEXT\_PATH\_INST\_ID)

and csr.EQUIP\_INST\_ID = ei.equip\_inst\_id

and ei.SITE\_INST\_ID in ( gpi.A\_SIDE\_SITE\_ID, gpi.Z\_SIDE\_SITE\_ID )

and cpi.Z\_SIDE\_SITE\_ID in (gpi.A\_SIDE\_SITE\_ID,gpi.Z\_SIDE\_SITE\_ID)

and ci.CIRC\_INST\_ID = gige.segment\_inst\_id

'

;

execute immediate sql\_stmt using gige\_path\_inst\_id;

--commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_parents\_w\_seg\_per\_gige(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_parents\_w\_seg\_per\_gige;

procedure load\_parents\_w\_microwave (gige\_path\_inst\_id in number, has\_vlan in boolean) Is

sql\_stmt\_common varchar2(32000);

sql\_stmt\_has\_vlan varchar2(32000);

sql\_stmt\_no\_vlan varchar2(32000);

sql\_stmt varchar2(32000);

begin

sql\_stmt := 'insert into XNG\_GIGE\_PARENT\_MW\_WK

WITH ALL\_PARENTS as

(

select distinct gp.gige\_inst\_id, gp.parent\_path\_inst\_id path\_inst\_id

from xng\_gige\_parents\_wk gp

where gp.gige\_inst\_id = :x

),

PARENTS\_OF\_GIGE\_W\_MICROWAVE as

(

-- now get only those parents of GigE that have a mw in them

select pp.gige\_inst\_id, pp.PATH\_INST\_ID, cpe.port\_inst\_id, cpe.sequence port\_order

from ALL\_PARENTS pp

join vzwnet.circ\_path\_element cpe

on cpe.CIRC\_PATH\_INST\_ID = pp.path\_inst\_id

and cpe.port\_inst\_id is not null

join vzwnet.epa

on epa.PORT\_INST\_ID = cpe.PORT\_INST\_ID

join vzwnet.equip\_inst ei

on ei.EQUIP\_INST\_ID = epa.equip\_inst\_id

and (

ei.type like ''MICROWAVE%''

or ei.type = ''DARK FIBER MUX''

)

)select distinct gige\_inst\_id from PARENTS\_OF\_GIGE\_W\_MICROWAVE'

;

execute immediate sql\_stmt using gige\_path\_inst\_id;

--commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_parents\_w\_microwave(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_parents\_w\_microwave;

-- if the gige has a segment of BW=GBPS or MBPS, then in XNG\_GIGE\_DELIVERY\_TABLE, it is stored as ETHERNET BW

Procedure load\_gige\_on\_native\_sonet (truncate\_tab in varchar2 default 'y') is

sql\_stmt varchar2(32000);

begin

if (truncate\_tab is null or lower(truncate\_tab) = 'y') then

sql\_stmt:='truncate table XNG\_GIGE\_DELIVERY\_TABLE\_WK';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error truncating table: '

||'XNG\_GIGE\_DELIVERY\_TABLE\_WK '|| TO\_CHAR(SQLCODE)||': '

||SQLERRM, 1, 255));

raise;

end;

END IF;

-- LR 06/30/2011

sql\_stmt:='insert into XNG\_GIGE\_DELIVERY\_TABLE\_WK

with valid\_giges as (

select distinct case when cpi.circ\_path\_inst\_id is null then cpi.next\_path\_inst\_id

else cpi.circ\_path\_inst\_id

end gige\_path\_inst\_id

from XNG\_CSR\_GIGE\_PATHS\_WK cpi

) select gige\_path\_inst\_id , cpe.segment\_inst\_id

, case when upper(ci.BANDWIDTH) like ''%GBPS'' then ''ETHERNET BW''

when upper(ci.BANDWIDTH) like ''%MBPS'' then ''ETHERNET BW''

when (upper(ci.BANDWIDTH)=''INFERRED'' and ci.type=''DARK FIBER'') then ''DARK FIBER BW''

else ci.BANDWIDTH

end delivered\_on,

cpe.SEQUENCE seg\_order from valid\_giges

join vzwnet.circ\_path\_element cpe

on cpe.CIRC\_PATH\_INST\_ID = gige\_path\_inst\_id

join vzwnet.circ\_inst ci

on cpe.SEGMENT\_INST\_ID = ci.CIRC\_INST\_ID

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_gige\_on\_native\_sonet(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_gige\_on\_native\_sonet;

procedure load\_bts\_paths\_for\_all\_csrs is

sql\_stmt varchar2(32000);

cursor CSRs is

select ei.equip\_inst\_id

from vzwnet.equip\_inst ei

where ei.type = 'CSR';

cnt number := 0;

begin

-- drop the table

sql\_stmt:='truncate table XNG\_CSR\_BTS\_PATHS\_WK';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error dropping the table: '

||'XNG\_CSR\_BTS\_PATHS\_WK '|| TO\_CHAR(SQLCODE)||': '

||SQLERRM, 1, 255));

raise;

end;

-- insert all CSR-BTS (both BTS and eNB) termination into this table

for rec in CSRs

LOOP

load\_bts\_paths\_for\_a\_csr(rec.equip\_inst\_id);

cnt := cnt + 1;

if (cnt = 1000)

THEN

dbms\_output.put\_line('Committing');

commit;

cnt := 0;

END IF;

END LOOP;

commit;

dbms\_output.put\_line('Added BTS/eNB terminations to XNG\_CSR\_BTS\_PATHS\_WK');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_bts\_paths\_for\_all\_csrs(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_bts\_paths\_for\_all\_csrs;

---------------------------NSP

-- there is a new path\_type 1XDO. That too should be considered here

-- AMPS should be removed from this mix.

---

---------------------------

Procedure load\_bts\_paths\_for\_a\_csr (CSR\_equip\_inst\_id in number) is

sql\_stmt varchar2(32000);

begin

sql\_stmt:='insert into XNG\_CSR\_BTS\_PATHS\_WK

WITH

GET\_ALL\_CSRS as (

select :y matched\_equip\_inst\_id

, ei.equip\_inst\_id, decode (ei.status, ''Live'', ''Y'', ''N'') csr\_live\_in\_xng

from XNG\_CSR\_PARSED\_WK csr, vzwnet.equip\_inst ei

where ei.EQUIP\_INST\_ID = csr.EQUIP\_INST\_ID

and csr.equip\_inst\_id = :x

)

,

GET\_CSR\_PORTS\_GOING\_TO\_BTS as

(

select p.site\_inst\_id, ei.\*, p.PORT\_INST\_ID, p.PORT\_HUM\_ID, p.BANDWIDTH

, p.CIRC\_PATH\_INST\_ID, cpi.BANDWIDTH path\_bw, cpi.CIRC\_PATH\_HUM\_ID path\_hum\_id

, p.NEXT\_PATH\_INST\_ID, cpi.BANDWIDTH next\_bw, cpi.CIRC\_PATH\_HUM\_ID next\_hum\_id

, decode(cpi.STATUS, ''Live'', ''Y'', ''N'') path\_live\_in\_xng

from GET\_ALL\_CSRS ei

join vzwnet.epa p

on ei.equip\_inst\_id = p.equip\_inst\_id

left outer join vzwnet.circ\_path\_inst cpi

on cpi.CIRC\_PATH\_INST\_ID = p.CIRC\_PATH\_INST\_ID

left outer join vzwnet.circ\_path\_inst npi

on npi.CIRC\_PATH\_INST\_ID = p.Next\_PATH\_INST\_ID

where

((cpi.BANDWIDTH = ''DS1'' or upper(cpi.BANDWIDTH) like ''%MBPS'')

and cpi.TYPE in ( ''CELL'', ''CDMA'', ''EVDO OA M'', ''EVDO'', ''IPBH'', ''AMPS''))

or

((npi.BANDWIDTH= ''DS1'' or upper(npi.BANDWIDTH) like ''%MBPS'')

and npi.TYPE in (''CELL'', ''CDMA'', ''EVDO OA M'', ''EVDO'', ''IPBH'', ''AMPS''))

)

-- get a set of all DS1 and MBPS paths that may or may not have a BTS port in them

select matched\_equip\_inst\_id, p.equip\_inst\_id, p.port\_inst\_id

, p.CIRC\_PATH\_INST\_ID

, p.NEXT\_PATH\_INST\_ID

, csr\_live\_in\_xng

, path\_live\_in\_xng

from GET\_CSR\_PORTS\_GOING\_TO\_BTS p

';

EXECUTE IMMEDIATE sql\_stmt using CSR\_equip\_inst\_id, CSR\_equip\_inst\_id ;

sql\_stmt:='insert into XNG\_CSR\_BTS\_PATHS\_WK

WITH

GET\_ALL\_CSR\_CHILD\_EQUIP as

(

select :y matched\_equip\_inst\_id

, ei.equip\_inst\_id, ei.descr, ei.status, ei.eq\_class

from vzwnet.equip\_inst ei

start with ei.equip\_inst\_id = :x --= 633557

connect by prior ei.equip\_inst\_id = ei.PARENT\_EQ\_INST\_ID

)

,

GET\_ALL\_CSR\_SHELVES as

(

select ei.\*, decode (ei.status, ''Live'', ''Y'', ''N'') csr\_live\_in\_xng

from GET\_ALL\_CSR\_CHILD\_EQUIP ei

where eq\_class = ''S''

)

,

GET\_CSR\_PORTS\_GOING\_TO\_eNB as

(

select p.site\_inst\_id, ei.\*, p.PORT\_INST\_ID, p.PORT\_HUM\_ID, p.BANDWIDTH

, p.CIRC\_PATH\_INST\_ID, cpi.BANDWIDTH path\_bw, cpi.CIRC\_PATH\_HUM\_ID path\_hum\_id

, p.NEXT\_PATH\_INST\_ID, cpi.BANDWIDTH next\_bw, cpi.CIRC\_PATH\_HUM\_ID next\_hum\_id

, decode(cpi.STATUS, ''Live'', ''Y'', ''N'') path\_live\_in\_xng

from GET\_ALL\_CSR\_SHELVES ei

-- get all ports on CSRs

join vzwnet.epa p

on ei.equip\_inst\_id = p.equip\_inst\_id

join vzwnet.circ\_path\_inst cpi

on cpi.CIRC\_PATH\_INST\_ID = p.CIRC\_PATH\_INST\_ID

join vzwnet.circ\_path\_inst npi

on npi.CIRC\_PATH\_INST\_ID = p.Next\_PATH\_INST\_ID

-- look at path element of all paths going thru CSR

join vzwnet.circ\_path\_element cpe

on cpe.CIRC\_PATH\_INST\_ID in (p.CIRC\_PATH\_INST\_ID, p.Next\_PATH\_INST\_ID)

-- find those GigE paths that also have eNB ports in them

join vzwnet.epa enbP

on cpe.PORT\_INST\_ID = enbP.port\_inst\_id

join vzwnet.equip\_inst enb

on enb.EQUIP\_INST\_ID = p.equip\_inst\_id

where

(upper(cpi.BANDWIDTH) like ''%GBPS'' or npi.BANDWIDTH like ''%GBPS'')

and enb.type in (''BASEBAND'',''ENODEB'',''MICROCELL'')

)

-- get a set of all GigE paths that have a eNB in them

select matched\_equip\_inst\_id, p.equip\_inst\_id, p.port\_inst\_id

, p.CIRC\_PATH\_INST\_ID

, p.NEXT\_PATH\_INST\_ID

, csr\_live\_in\_xng

, path\_live\_in\_xng

from GET\_CSR\_PORTS\_GOING\_TO\_eNB p

';

EXECUTE IMMEDIATE sql\_stmt using CSR\_equip\_inst\_id, CSR\_equip\_inst\_id ;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_bts\_paths\_for\_a\_csr(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_bts\_paths\_for\_a\_csr;

Procedure load\_gige\_vlans is

sql\_stmt varchar2(32000);

begin

-- get all vlans on these gige paths

sql\_stmt:='drop table XNG\_GIGE\_VLAN\_MAP\_WK';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error dropping the table: '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

-- LR 06/30/2011

sql\_stmt:='create table XNG\_GIGE\_VLAN\_MAP\_WK as

select distinct cpe.PATH\_INST\_ID gige\_path\_inst\_id --, cpe.CHANNEL\_USED, cpe.CHAN\_INST\_ID

, cpi.CIRC\_PATH\_INST\_ID vlan\_inst\_id

, decode(cpi.STATUS, ''Live'', ''Y'', ''N'') vlan\_live\_in\_xng

, vlan\_number

from XNG\_CSR\_GIGE\_PATHS\_WK xcp

-- get path where current GigE is the parent path

join vzwnet.circ\_path\_element cpe

on xcp.circ\_PATH\_INST\_ID = cpe.PATH\_INST\_ID

-- ensure that the path you got from prev join is a VLAN

join vzwnet.circ\_path\_inst cpi

on cpi.CIRC\_PATH\_INST\_ID = cpe.circ\_PATH\_INST\_ID

and cpi.BANDWIDTH = ''VLAN''

join XNG\_VLAN\_PARSED\_WK xvp

on xvp.CIRC\_PATH\_inst\_ID = cpi.CIRC\_PATH\_INST\_ID

'

;

EXECUTE IMMEDIATE sql\_stmt;

-- get all vlans on next gige paths

-- DONT truncate here.

--This is doing inserts in the same table that was created above

sql\_stmt:='insert into XNG\_GIGE\_VLAN\_MAP\_WK

select distinct cpe.PATH\_INST\_ID gige\_path\_inst\_id --, cpe.CHANNEL\_USED, cpe.CHAN\_INST\_ID

, cpi.CIRC\_PATH\_INST\_ID vlan\_inst\_id

, decode(cpi.STATUS, ''Live'', ''Y'', ''N'') vlan\_live\_in\_xng

, vlan\_number

from XNG\_CSR\_GIGE\_PATHS\_WK xcp

-- get path where next GigE is the parent path

join vzwnet.circ\_path\_element cpe

on xcp.NEXT\_PATH\_INST\_ID = cpe.PATH\_INST\_ID

-- ensure that the path you got from prev join is a VLAN

join vzwnet.circ\_path\_inst cpi

on cpi.CIRC\_PATH\_INST\_ID = cpe.circ\_PATH\_INST\_ID

and cpi.BANDWIDTH = ''VLAN''

join XNG\_VLAN\_PARSED\_WK xvp

on xvp.CIRC\_PATH\_inst\_ID = cpi.CIRC\_PATH\_INST\_ID

'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='CREATE INDEX GIGE\_VLAN\_MAP\_WK\_IDX1 ON XNG\_GIGE\_VLAN\_MAP\_WK (GIGE\_PATH\_INST\_ID)';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='CREATE INDEX GIGE\_VLAN\_MAP\_WK\_IDX2 ON XNG\_GIGE\_VLAN\_MAP\_WK (VLAN\_INST\_ID)';

EXECUTE IMMEDIATE sql\_stmt;

dbms\_output.put\_line('Update XNG\_GIGE\_VLAN\_MAP\_WK completed');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_gige\_vlans(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_gige\_vlans;

-- for giges with no vlans, then store its parents

procedure get\_gige\_parents is

CURSOR all\_giges is

select distinct GP.CIRC\_PATH\_INST\_ID gige\_inst\_id

from XNG\_CSR\_GIGE\_PATHS\_WK gp

union

select distinct GP.NEXT\_PATH\_INST\_ID gige\_inst\_id

from XNG\_CSR\_GIGE\_PATHS\_WK gp

;

sql\_stmt varchar2(200);

BEGIN

-- get all vlans on these gige paths

sql\_stmt:='truncate table xng\_gige\_parents\_wk';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error truncating the table: '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

for rec in all\_giges

loop

get\_gige\_parents\_in\_1st\_leg(rec.gige\_inst\_id, rec.gige\_inst\_id, '', 0);

commit;

end loop;

end get\_gige\_parents;

/\* this gets all parents ONLY in the first leg of the path

the code is written to not go past 10 parents up.

This is helpful in the case where due to problem-modelling in xng, there is a loop in data

without this you get an error "maximum open cursors exceeded"

\*/

procedure get\_gige\_parents\_in\_1st\_leg (start\_path in number, current\_path in number, p\_sys\_conn\_by\_path in varchar2, deep in number) is

CURSOR PARENTS\_IN\_1ST\_LEG (ARG\_PATH number) iS

select \*

from (

select distinct pl.circ\_path\_inst\_id, pl.leg\_inst\_id

, cpe.path\_inst\_id parent\_path\_inst\_id, pl.leg\_name, pl.rel\_order

, rank () over (partition by pl.circ\_path\_inst\_id order by rel\_order) as first\_leg

, cpi.TOPOLOGY

--, cpi.A\_SIDE\_SITE\_ID

--, cpi.z\_SIDE\_SITE\_ID

, cpe.element\_type

--, cpe.element\_inst\_id

--, plm.sequence element\_seq\_number

from vzwnet.circ\_path\_inst cpi

join vzwnet.circ\_path\_element cpe

on cpi.CIRC\_PATH\_INST\_ID = cpe.CIRC\_PATH\_INST\_ID

join vzwnet.path\_leg\_inst pl

on pl.CIRC\_PATH\_INST\_ID = cpe.CIRC\_PATH\_INST\_ID

/\*join vzwnet.path\_leg\_member plm

on pl.LEG\_INST\_ID = plm.LEG\_INST\_ID

and cpe.element\_inst\_id = plm.ELEMENT\_INST\_ID\*/

WHERE cpe.CIRC\_PATH\_INST\_ID = ARG\_PATH

and cpe.ELEMENT\_TYPE in ('P', 'K')

)

where first\_leg = 1

;

sql\_stmt varchar2(2000) := 'insert into xng\_gige\_parents\_wk columns

(gige\_inst\_id, leg\_inst\_id

, parent\_path\_inst\_id, min\_rel\_order

--, a\_side\_site\_id, z\_side\_site\_id

, sys\_conn\_by\_path

--, element\_inst\_id

--, element\_seq\_number

)

values(:gige\_inst\_id, :leg\_inst\_id

, :parent\_path\_inst\_id, :min\_rel\_order

--, :a\_side\_site\_id, :z\_side\_site\_id

, :sys\_conn\_by\_path

--, :element\_inst\_id

--, :element\_seq\_number

)';

sys\_conn\_by\_path varchar2(1000) := p\_sys\_conn\_by\_path;

BEGIN

sys\_conn\_by\_path := sys\_conn\_by\_path||'/'||current\_path;

for rec in PARENTS\_IN\_1ST\_LEG(current\_path)

loop

--insert parents in table

EXECUTE IMMEDIATE sql\_stmt using start\_path

, rec.leg\_inst\_id

, rec.parent\_path\_inst\_id, rec.first\_leg

--, rec.a\_side\_site\_id, rec.z\_side\_site\_id

, sys\_conn\_by\_path||'/'||rec.parent\_path\_inst\_id

--, rec.element\_inst\_id

--, rec.element\_seq\_number

;

--dbms\_output.PUT\_LINE (sys\_conn\_by\_path);

if (upper(rec.topology) <> 'U' -- Uni-Ring'

and upper(rec.topology) <> 'R' -- 'Bi-Ring'

and rec.element\_type <> 'K' -- path link

)

then

if (deep <= RECURSIVE\_DEPTH)

THEN

--recursively call the same procedure for the parent\_path

get\_gige\_parents\_in\_1st\_leg(start\_path, rec.parent\_path\_inst\_id, sys\_conn\_by\_path, deep+1);

end if;

END IF;

end loop;

end;

/\*-----------------------------------------------------------------\*/

procedure get\_vlan\_parents is

CURSOR VLANS is

select distinct vlan\_inst\_id

from XNG\_GIGE\_VLAN\_MAP\_WK vl

--where vl.CIRC\_PATH\_INST\_ID = '2556993' -- causes loop

;

/\*

CURSOR VLAN\_GP\_W\_MULT\_CSRs IS

WITH

PARENTS as

(

select distinct parent\_path\_inst\_id, rowid parent\_rowid

from xng\_vlan\_parents\_wk xvp

)

select parent\_rowid --parent\_path\_inst\_id, csr1.DESCR, csr2.DESCR

from PARENTS pp

join vzwnet.circ\_path\_element cpe1

on cpe1.CIRC\_PATH\_INST\_ID = pp.parent\_PATH\_INST\_ID

join vzwnet.epa cp1

on cpe1.PORT\_INST\_ID = cp1.PORT\_INST\_ID

join vzwnet.equip\_inst csr1

on csr1.EQUIP\_INST\_ID = cp1.equip\_inst\_id

join vzwnet.circ\_path\_element cpe2

on cpe2.CIRC\_PATH\_INST\_ID = pp.parent\_PATH\_INST\_ID

join vzwnet.epa cp2

on cpe2.PORT\_INST\_ID = cp2.PORT\_INST\_ID

join vzwnet.equip\_inst csr2

on csr2.EQUIP\_INST\_ID = cp2.equip\_inst\_id

where

csr1.rowid <> csr2.rowid

and csr1.TYPE = csr2.TYPE

and csr2.TYPE = 'CSR'

;

\*/

-- if the vlan's parents have more than 1 CSR combined (all parents together)

-- then treat it as spoke to hub

-- originally spoke-to-hub required more than 1 CSR on the same parent

CURSOR VLAN\_GP\_W\_MULT\_CSRs IS

WITH

PARENTS AS (

SELECT DISTINCT parent\_path\_inst\_id, xvp.vlan\_inst\_id

FROM xng\_reports.xng\_vlan\_parents\_wk xvp

),

multiple\_csr AS (

SELECT DISTINCT csr1.equip\_inst\_id, vlan\_inst\_id

FROM PARENTS pp

JOIN vzwnet.circ\_path\_element cpe1 ON cpe1.CIRC\_PATH\_INST\_ID = pp.parent\_PATH\_INST\_ID

JOIN vzwnet.epa cp1 ON cpe1.PORT\_INST\_ID = cp1.PORT\_INST\_ID

JOIN vzwnet.equip\_inst csr1 ON csr1.EQUIP\_INST\_ID = cp1.equip\_inst\_id

WHERE csr1.TYPE = 'CSR'

)

SELECT vlan\_inst\_id, COUNT ( \* )

FROM multiple\_csr c

GROUP BY vlan\_inst\_id

HAVING COUNT ( \* ) > 1

;

sql\_stmt varchar2(2000);

BEGIN

-- get all vlans on these gige paths

sql\_stmt:='truncate table xng\_vlan\_parents\_wk reuse storage';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error truncating the table: '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

for rec in VLANS

loop

get\_vlan\_parents\_in\_1st\_leg(rec.vlan\_inst\_id, rec.vlan\_inst\_id, '', 0);

commit;

end loop;

-- a\_side of the parent path is going to the mtso

sql\_stmt := 'update xng\_vlan\_parents\_wk xvp set

terminates\_on\_mtso = 1

where

exists (select 1

from vzwnet.site\_inst si

join vzwnet.circ\_path\_inst cpi

on si.site\_inst\_id in (cpi.A\_SIDE\_SITE\_ID, cpi.Z\_SIDE\_SITE\_ID)

where xvp.VLAN\_INST\_ID = cpi.CIRC\_PATH\_INST\_ID

and num like ''MTSO%''

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- if some vlan's parent has 2 CSRs in it then it means it is connecting spoke to hub site

-- identify those here

sql\_stmt := 'update xng\_vlan\_parents\_wk xvp set spoke\_to\_hub = 1 where vlan\_inst\_id = :vlan\_inst\_id';

for rec in VLAN\_GP\_W\_MULT\_CSRs

loop

-- update the parent path to say it goes from CSR to CSR

execute immediate sql\_stmt using rec.vlan\_inst\_id;

end loop;

commit;

-- take care of null nvl all to 0

sql\_stmt := 'update xng\_vlan\_parents\_wk xvp set

terminates\_on\_mtso = nvl(terminates\_on\_mtso, 0)

, has\_mspp = nvl(has\_mspp, 0)

, has\_ngMLS = nvl(has\_ngMLS, 0)

, spoke\_to\_hub = nvl(spoke\_to\_hub, 0)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in get\_vlan\_parents(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end get\_vlan\_parents;

/\* this gets all parents ONLY in the first leg of the path

the code is written to not go past 10 parents up.

This is helpful in the case where due to problem-modelling in xng, there is a loop in data

without this you get an error "maximum open cursors exceeded"

\*/

procedure get\_vlan\_parents\_in\_1st\_leg (start\_path in number, current\_path in number, p\_sys\_conn\_by\_path in varchar2, deep in number) is

CURSOR PARENTS\_IN\_1ST\_LEG (ARG\_PATH number) iS

select \*

from (

select distinct pl.circ\_path\_inst\_id, pl.leg\_inst\_id

, cpe.path\_inst\_id parent\_path\_inst\_id, pl.leg\_name, pl.rel\_order

, rank () over (partition by pl.circ\_path\_inst\_id order by rel\_order) as first\_leg

, cpi.TOPOLOGY

--, cpi.A\_SIDE\_SITE\_ID

--, cpi.z\_SIDE\_SITE\_ID

, cpe.element\_type

--, cpe.element\_inst\_id

--, plm.sequence element\_seq\_number

from vzwnet.circ\_path\_inst cpi

join vzwnet.circ\_path\_element cpe

on cpi.CIRC\_PATH\_INST\_ID = cpe.CIRC\_PATH\_INST\_ID

join vzwnet.path\_leg\_inst pl

on pl.CIRC\_PATH\_INST\_ID = cpe.CIRC\_PATH\_INST\_ID

/\*join vzwnet.path\_leg\_member plm

on pl.LEG\_INST\_ID = plm.LEG\_INST\_ID

and cpe.element\_inst\_id = plm.ELEMENT\_INST\_ID\*/

WHERE cpe.CIRC\_PATH\_INST\_ID = ARG\_PATH

and cpe.ELEMENT\_TYPE in ('P', 'K')

)

where first\_leg = 1

;

sql\_stmt varchar2(2000) := 'insert into xng\_vlan\_parents\_wk columns

(vlan\_inst\_id, leg\_inst\_id

, parent\_path\_inst\_id, min\_rel\_order

--, a\_side\_site\_id, z\_side\_site\_id

, sys\_conn\_by\_path

--, element\_inst\_id

--, element\_seq\_number

)

values(:vlan\_inst\_id, :leg\_inst\_id

, :parent\_path\_inst\_id, :min\_rel\_order

--, :a\_side\_site\_id, :z\_side\_site\_id

, :sys\_conn\_by\_path

--, :element\_inst\_id

--, :element\_seq\_number

)';

sys\_conn\_by\_path varchar2(1000) := p\_sys\_conn\_by\_path;

BEGIN

sys\_conn\_by\_path := sys\_conn\_by\_path||'/'||current\_path;

for rec in PARENTS\_IN\_1ST\_LEG(current\_path)

loop

--insert parents in table

EXECUTE IMMEDIATE sql\_stmt using start\_path

, rec.leg\_inst\_id

, rec.parent\_path\_inst\_id, rec.first\_leg

--, rec.a\_side\_site\_id, rec.z\_side\_site\_id

, sys\_conn\_by\_path||'/'||rec.parent\_path\_inst\_id

--, rec.element\_inst\_id

--, rec.element\_seq\_number

;

--dbms\_output.PUT\_LINE (sys\_conn\_by\_path);

if (upper(rec.topology) <> 'U' -- Uni-Ring'

and upper(rec.topology) <> 'R' -- 'Bi-Ring'

and rec.element\_type <> 'K' -- path link

)

then

if (deep <= RECURSIVE\_DEPTH)

THEN

--recursively call the same procedure for the parent\_path

get\_vlan\_parents\_in\_1st\_leg(start\_path, rec.parent\_path\_inst\_id, sys\_conn\_by\_path, deep+1);

end if;

END IF;

end loop;

end;

PROCEDURE load\_vlan\_ngMLS\_map is

sql\_stmt varchar2(2000) := '';

begin

-- store ngMLS path, equip, port mapping in a table

sql\_stmt := 'truncate table xng\_vlan\_ngMLS\_map\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt := 'insert into xng\_vlan\_ngMLS\_map\_wk columns

(CIRC\_PATH\_INST\_ID, PORT\_INST\_ID, EQUIP\_INST\_ID, card\_inst\_id)

WITH

PARENT\_PATHS as

(

select distinct xvp.parent\_path\_inst\_id

from xng\_vlan\_parents\_wk xvp

)

select cpe.CIRC\_PATH\_INST\_ID, p.PORT\_INST\_ID, ei.EQUIP\_INST\_ID, card\_inst\_id

from PARENT\_PATHS xvp

join vzwnet.circ\_path\_element cpe

on cpe.circ\_path\_inst\_id = xvp.parent\_path\_inst\_id

join vzwnet.epa p

on p.port\_inst\_id = cpe.port\_INST\_ID

join vzwnet.equip\_inst ei

on ei.EQUIP\_INST\_ID = p.EQUIP\_INST\_ID

where ei.TYPE = ''NGMLS''

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- does parent path have any ngMLS ports

sql\_stmt := 'update xng\_vlan\_parents\_wk xvp set

has\_ngmls = 1

where

exists (select 1

from xng\_vlan\_ngMLS\_map\_wk xvng

where xvng.circ\_path\_inst\_id = xvp.parent\_path\_inst\_id

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_vlan\_ngMLS\_map(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_vlan\_ngMLS\_map;

/\* this needs to be called after load\_vlan\_ngmls\_map \*/

PROCEDURE load\_ngMLS\_devices is

CURSOR ngMLS\_CARD\_TEST\_HEAD\_NAME is

WITH

MSPPs as

(

select distinct xmd.EQUIP\_INST\_ID

from xng\_ngMLS\_devices\_wk xmd

)

select ci.card\_inst\_id, ci.type card\_type, cas.ATTR\_VALUE

from MSPPs ms

join vzwnet.card\_inst ci

on ci.EQUIP\_INST\_ID = ms.EQUIP\_INST\_ID

left outer join vzwnet.card\_attr\_settings cas

on cas.CARD\_INST\_ID = ci.CARD\_INST\_ID

and cas.VAL\_ATTR\_INST\_ID = 7044 -- put the right QT UDA here??????

where (ci.TYPE like '10XGE MDA%' or ci.TYPE like 'A9K-SIP-700%')

;

sql\_stmt varchar2(2000) := '';

begin

--

sql\_stmt := 'truncate table xng\_ngmls\_devices\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt := 'insert into xng\_ngmls\_devices\_wk columns (equip\_inst\_id)

select distinct equip\_inst\_id

from xng\_vlan\_ngMLS\_map\_wk

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- store card to test head mapping

sql\_stmt := 'update xng\_vlan\_ngMLS\_map\_wk xmd set test\_head\_name = :test\_head\_name, card\_type = :card\_type where card\_inst\_id = :card\_inst\_id ';

for rec in ngMLS\_CARD\_TEST\_HEAD\_NAME

loop

-- update the test Head name on hte ngMLS card and its card type

execute immediate sql\_stmt using rec.attr\_value, rec.card\_type, rec.card\_inst\_id;

end loop;

commit;

sql\_stmt := 'update xng\_ngMLS\_devices\_wk xnd set test\_head\_name = (

WITH

ngMLS as

(

select distinct xmd.EQUIP\_INST\_ID

from xng\_ngMLS\_devices\_wk xmd

)

-- for these ngMLS get QT on the top-equip

select eas.ATTR\_VALUE test\_head\_name

from ngMLS ng

join vzwnet.equip\_attr\_settings eas

on eas.EQUIP\_INST\_ID = ng.EQUIP\_INST\_ID

where xnd.EQUIP\_INST\_ID = ng.EQUIP\_INST\_ID

and eas.VAL\_ATTR\_INST\_ID = 6220 -- Remote Test Unit / Test Head Name

)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_ngMLS\_devices;(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_ngMLS\_devices;

PROCEDURE load\_mspp\_devices is

sql\_stmt varchar2(2000) := '';

begin

--

sql\_stmt := 'truncate table xng\_mspp\_devices\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- for perfromance sake, dump all mspp\_devices in a table

sql\_stmt := 'truncate table xng\_mspp\_devices\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- getting alu from here

sql\_stmt := 'insert into xng\_mspp\_devices\_wk columns ( EQ\_CLASS\_TYPE, EQ\_CLASS, TYPE, STATUS,

ACCESS\_ID, NE\_ACCESS\_ID, INST\_ID, PARENT\_EQ\_INST\_ID,

NE\_INST\_ID, VENDOR, MODEL, DESCR, CLLI,SITE\_INST\_ID

)

select EQ\_CLASS\_TYPE, EQ\_CLASS, TYPE, STATUS,

ACCESS\_ID, NE\_ACCESS\_ID, INST\_ID, PARENT\_EQ\_INST\_ID,

NE\_INST\_ID, VENDOR, MODEL, DESCR, CLLI,SITE\_INST\_ID

from mspp\_devices\_v

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- getting fujitsu from here

-- 9500, the top level equip is being inserted here, should be of type MSPP

sql\_stmt :=' insert into xng\_mspp\_devices\_wk columns ( EQ\_CLASS\_TYPE, EQ\_CLASS, TYPE, STATUS,

INST\_ID, PARENT\_EQ\_INST\_ID,

NE\_INST\_ID, VENDOR, MODEL, DESCR, CLLI,SITE\_INST\_ID

)

select EQ\_CLASS\_TYPE, x.EQ\_CLASS, x.TYPE, x.STATUS,

ei.equip\_inst\_id INST\_ID, x.PARENT\_EQ\_INST\_ID,

NE\_INST\_ID, ei.VENDOR, ei.MODEL, ei.DESCR, CLLI,ei.SITE\_INST\_ID

from xng\_fw\_equip x, vzwnet.equip\_inst ei

where X.EQ\_CLASS\_TYPE=''NE''

and X.MODEL like ''%9500%''

and x.type = ''MSPP''

and EI.EQUIP\_INST\_ID = X.EQUIP\_INST\_ID

'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- 9500, the shelves are being inserted here, type for shelf can be anything

-- LR 04/23/2013 change MODEL to type

sql\_stmt := ' insert into xng\_mspp\_devices\_wk columns ( EQ\_CLASS\_TYPE, EQ\_CLASS, TYPE, STATUS,

INST\_ID, PARENT\_EQ\_INST\_ID,

NE\_INST\_ID, VENDOR, MODEL, DESCR, CLLI,SITE\_INST\_ID

)

with

all\_ne as

(

select Xfe.EQUIP\_INST\_ID

from xng\_fw\_equip xfe

where Xfe.EQ\_CLASS\_TYPE=''NE''

and Xfe.MODEL like ''%9500%''

and Xfe.type = ''MSPP''

)

select x.EQ\_CLASS\_TYPE, x.EQ\_CLASS, x.TYPE, x.STATUS,

x.equip\_INST\_ID,x.PARENT\_EQ\_INST\_ID,

x.NE\_INST\_ID, x.VENDOR, x.MODEL, x.DESCR,CLLI,

x.SITE\_INST\_ID

from xng\_fw\_equip x, all\_ne, vzwnet.equip\_inst ei

where X.EQ\_CLASS\_TYPE=''SHELF''

and all\_ne.equip\_inst\_id = X.PARENT\_EQ\_INST\_ID

and EI.EQUIP\_INST\_ID = x.equip\_inst\_id

'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt := 'update xng\_mspp\_devices\_wk xmd set test\_head\_name = (

WITH

MSPPs as

(

select distinct xmd.NE\_INST\_ID

from xng\_mspp\_devices\_wk xmd

where xmd.eq\_class\_type = ''NE''

)

-- for these mspp get QT on the top-equip

select eas.ATTR\_VALUE test\_head\_name

from MSPPs ms

join vzwnet.equip\_attr\_settings eas

on eas.EQUIP\_INST\_ID = ms.NE\_INST\_ID

where xmd.NE\_INST\_ID = ms.NE\_INST\_ID

and eas.VAL\_ATTR\_INST\_ID = 6220 -- Remote Test Unit / Test Head Name

)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_mspp\_devices;(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_mspp\_devices;

PROCEDURE load\_vlan\_mspp\_map is

CURSOR MSPP\_CARD\_TEST\_HEAD\_NAME is

WITH

MSPPs as

(

select distinct xmd.NE\_INST\_ID, inst\_id

from xng\_mspp\_devices\_wk xmd

where xmd.eq\_class\_type = 'NE'

)

select ci.card\_inst\_id, ci.type card\_type, cas.ATTR\_VALUE

from MSPPs ms

join vzwnet.card\_inst ci

on ci.EQUIP\_INST\_ID = ms.INST\_ID

left outer join vzwnet.card\_attr\_settings cas

on cas.CARD\_INST\_ID = ci.CARD\_INST\_ID

and cas.VAL\_ATTR\_INST\_ID = 7103 -- put the right QT UDA here??????

where (ci.TYPE like 'ES64%' or ci.TYPE like 'FC9565EXX1%')

;

sql\_stmt varchar2(32000) := '';

begin

-- store mspp path, equip, port mapping in a table

sql\_stmt := 'truncate table xng\_vlan\_mspp\_map\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt := 'insert into xng\_vlan\_mspp\_map\_wk columns (CIRC\_PATH\_INST\_ID, PORT\_INST\_ID, equip\_inst\_id, ne\_inst\_id, card\_inst\_id)

WITH

PARENT\_PATHS as

(

select distinct xvp.parent\_path\_inst\_id

from xng\_vlan\_parents\_wk xvp

)

select cpe.CIRC\_PATH\_INST\_ID, p.PORT\_INST\_ID, ei.inst\_id equip\_INST\_ID, ei.ne\_inst\_id, p.card\_inst\_id

from PARENT\_PATHS xvp

join vzwnet.circ\_path\_element cpe

on cpe.circ\_path\_inst\_id = xvp.parent\_path\_inst\_id

join vzwnet.epa p

on p.port\_inst\_id = cpe.port\_INST\_ID

join xng\_mspp\_devices\_wk ei

on ei.INST\_ID = p.equip\_INST\_ID

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- does parent path have any mspp ports

sql\_stmt := 'update xng\_vlan\_parents\_wk xvp set

has\_mspp = 1

where

exists (select 1

from xng\_vlan\_mspp\_map\_wk xvng

where xvng.circ\_path\_inst\_id = xvp.parent\_path\_inst\_id

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- store card to test head mapping

sql\_stmt := 'update xng\_vlan\_mspp\_map\_wk xmd set test\_head\_name = :test\_head\_name, card\_type = :card\_type where card\_inst\_id = :card\_inst\_id ';

for rec in MSPP\_CARD\_TEST\_HEAD\_NAME

loop

-- update the test Head name on hte card and card type

execute immediate sql\_stmt using rec.attr\_value, rec.card\_type, rec.card\_inst\_id;

end loop;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_vlan\_mspp\_map(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_vlan\_mspp\_map;

/\*

SCR000000028572

REQUEST UPDATE TO THE COMPLETE EBH REPORT FOR VLAN PATHS WITH FUJITSU MSPPS

TO ENSURE CARD SLOT AND PORT NAME MEET STANDARDS

SLOT x-xx, port\_hum\_id x

\*/

PROCEDURE load\_vlan\_mspp\_tam is

sql\_stmt varchar2(32000) := '';

begin

-- store mspp path, equip, port mapping in a table

sql\_stmt := 'truncate table xng\_vlan\_mspp\_tam\_wk';

-- Get the VLAN's parent that has MSPP and also ngMLS and ensure

-- that the slot and port follow the naming convention.

-- Does not have to be the VLAN's immediate parent

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt := 'insert into xng\_vlan\_mspp\_tam\_wk columns (vlan\_inst\_id,vlan\_number,

port\_inst\_id, card\_inst\_id, parent\_path\_inst\_id,

test\_head\_on\_card, test\_head\_on\_eq, port\_hum\_id, SLOT, VENDOR)

select distinct xvp.vlan\_inst\_id,gvm.vlan\_number, xvm.port\_inst\_id, xvm.card\_inst\_id, parent\_path\_inst\_id,

XVM.TEST\_HEAD\_NAME test\_head\_on\_card, EAS.ATTR\_VALUE test\_head\_on\_eq, epa.port\_hum\_id, SI.SLOT, XMD.VENDOR

from xng\_reports.xng\_vlan\_parents\_wk xvp

join xng\_reports.xng\_vlan\_mspp\_map\_wk xvm

on xvp.PARENT\_PATH\_INST\_ID = xvm.CIRC\_PATH\_INST\_ID

join xng\_reports.netsmart\_ne\_vs\_xng\_audit\_wk xmd

on xvm.NE\_INST\_ID = xmd.NE\_INST\_ID

join xng\_gige\_vlan\_map gvm

on GVM.VLAN\_INST\_ID = XVP.VLAN\_INST\_ID

join vzwnet.epa on EPA.PORT\_INST\_ID = XVM.PORT\_INST\_ID

and EPA.EQUIP\_INST\_ID = XVM.EQUIP\_INST\_ID

join vzwnet.slot\_inst si on SI.CARD\_INST\_ID = XVM.CARD\_INST\_ID

and EPA.CARD\_INST\_ID= SI.CARD\_INST\_ID

left join vzwnet.equip\_attr\_settings eas on EAS.EQUIP\_INST\_ID = XVM.NE\_INST\_ID

and EAS.VAL\_ATTR\_INST\_ID=6220 -- test head uda on eq

where has\_mspp = 1 and has\_ngmls =1

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt := '

update xng\_vlan\_mspp\_tam\_wk tam set TAM.PARSE\_STATUS = 21

where not exists (

select 1 from

(

select distinct W.VLAN\_INST\_ID

from xng\_vlan\_mspp\_tam\_wk w

where upper(VENDOR) = ''FUJITSU''

and regexp\_like(slot, ''^[[:digit:]]-[[:digit:]]{2}$'')

and regexp\_like(port\_hum\_id, ''^[[:digit:]]$'')

)matches

where TAM.VLAN\_INST\_ID = matches.vlan\_inst\_id

)'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt := '

update xng\_vlan\_mspp\_tam\_wk tam set TAM.TAM\_STRING = ''@@''||TAM.VLAN\_NUMBER||''@''||TAM.SLOT || ''-'' || TAM.PORT\_HUM\_ID

where exists (

select 1 from

(

select distinct W.VLAN\_INST\_ID

from xng\_vlan\_mspp\_tam\_wk w

where upper(w.VENDOR) = ''FUJITSU''

and

W.PARSE\_STATUS is null or W.PARSE\_STATUS <> 21

)matches

where TAM.VLAN\_INST\_ID = matches.vlan\_inst\_id

)'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_vlan\_mspp\_tam(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_vlan\_mspp\_tam;

PROCEDURE update\_ngMLS\_qt\_issues is

CURSOR ngMLS\_QT\_ISSUES is

WITH

-- get distinct ngMLS in xng and QT on the top-equip

GET\_QT\_ON\_NE as

(

select distinct xmd.equip\_inst\_id, test\_head\_name

from xng\_ngMLS\_devices\_wk xmd

)

,

-- for these ngMLS get QT all ?? cards on the top-equip

-- also get QT on these cards

GET\_QT\_ON\_CARDS as

(

select xvm.equip\_inst\_id, xvm.card\_INST\_ID, card\_type, xvm.test\_head\_name

from xng\_vlan\_ngMLS\_map\_wk xvm

where (card\_TYPE like '10XGE MDA%' or card\_TYPE like 'A9K-SIP-700%')

and xvm.test\_head\_name is not null

)

,

-- the a valid QT600 is found on both the top-eq and

-- cards on the ngMLS, then dont know which one to use.

-- put that as an error

ERR\_VALID\_QT\_ON\_CI\_AND\_EQ as

(

select ne.equip\_inst\_id, ne.test\_head\_name ne\_test\_head

, cas.card\_type, cas.test\_head\_name card\_test\_head\_name

, case when ne.test\_head\_name = cas.test\_head\_name then 2 -- same name found on both eq

else 3 -- different name found on both eq

end issue\_code

from GET\_QT\_ON\_NE ne

join GET\_QT\_ON\_CARDS cas

on cas.equip\_inst\_id = ne.equip\_inst\_id

where

ne.test\_head\_name is not null and cas.test\_head\_name is not null

)

,

-- get those ngMLS where QT missing on the top-equip

QT\_MISSING\_ON\_ngMLS\_EQ as

(

select equip\_inst\_id

from GET\_QT\_ON\_NE ms

where ms.test\_head\_name is null

)

,

-- if QT missing on the equip, check if QT found on every ES64 / EXX card

-- if valid QT not present on ALL ES64/EXX cards then it shld show up in this list

NO\_QT\_ON\_ngMLS\_CI\_OR\_EQ as

(

select distinct qtm.equip\_inst\_id --, cas.ATTR\_VALUE qt\_name

from QT\_MISSING\_ON\_ngMLS\_EQ qtm

left outer join GET\_QT\_ON\_CARDS gqc

on qtm.equip\_inst\_id = gqc.equip\_inst\_id

where gqc.card\_inst\_id is null

)

,

GUD\_QT\_ON\_ngMLS as

(

-- all ngMLS

select equip\_inst\_id

from GET\_QT\_ON\_NE

minus

-- minus all ngMLS that dont have any ngMLS on eq or card

select equip\_inst\_id

from NO\_QT\_ON\_ngMLS\_CI\_OR\_EQ

minus

-- minus all ngMLS that have valid QT on both eq and card

select equip\_inst\_id

from ERR\_VALID\_QT\_ON\_CI\_AND\_EQ

)

select equip\_inst\_id, 1 issue\_code

from GUD\_QT\_ON\_ngMLS

union

-- no QT on the device

select equip\_inst\_id, 0

from NO\_QT\_ON\_ngMLS\_CI\_OR\_EQ

union

-- minus all ngMLS that have QT on card and EQ

select equip\_inst\_id, issue\_code

from ERR\_VALID\_QT\_ON\_CI\_AND\_EQ

;

sql\_stmt varchar2(2000);

BEGIN

-- if (QT is there only ngMLS top-equip or on all cards of certain types) = 1

-- if (no QT on ngMLS and on all no cards) = 0

-- if (QT on BOTH top-equip and on all/some cards and same) = 2

-- if (QT on BOTH top-equip and on all/some cards and different) = 3

-- identify those here

sql\_stmt := 'update xng\_ngmls\_devices\_wk xmd set issue\_code = :issue\_code where equip\_inst\_id = :equip\_inst\_id';

for rec in ngMLS\_QT\_ISSUES

loop

-- update the ngMLS shelf to say it QT\_issues

execute immediate sql\_stmt using rec.issue\_code, rec.equip\_inst\_id;

end loop;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in update\_ngMLS\_qt\_issues(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end update\_ngMLS\_qt\_issues;

PROCEDURE update\_mspp\_qt\_issues is

CURSOR MSPP\_QT\_ISSUES is

WITH

-- get distinct mspp in xng and QT on the top-equip

GET\_QT\_ON\_NE as

(

select distinct xmd.NE\_INST\_ID, test\_head\_name

from xng\_mspp\_devices\_wk xmd

where xmd.eq\_class\_type = 'NE'

)

,

-- for these mspp get QT all ES64 / EXX cards on the top-equip

-- also get QT on these cards

GET\_QT\_ON\_CARDS as

(

select xvm.NE\_INST\_ID, xvm.equip\_inst\_id, xvm.card\_INST\_ID, card\_type, xvm.test\_head\_name

from xng\_vlan\_mspp\_map\_wk xvm

where (xvm.card\_TYPE like 'ES64%' or xvm.card\_TYPE like 'FC9565EXX1%')

and xvm.test\_head\_name is not null

)

,

-- the a valid QT600 is found on both the top-eq and

-- cards on the mspp, then dont know which one to use.

-- put that as an error

ERR\_VALID\_QT\_ON\_CI\_AND\_EQ as

(

select ne.NE\_INST\_ID, ne.test\_head\_name ne\_test\_head

, cas.card\_type, cas.test\_head\_name card\_test\_head\_name

, case when ne.test\_head\_name = cas.test\_head\_name then 2 -- same name found on both eq

else 3 -- different name found on both eq

end issue\_code

from GET\_QT\_ON\_NE ne

join GET\_QT\_ON\_CARDS cas

on cas.NE\_INST\_ID = ne.ne\_inst\_id

where

ne.test\_head\_name is not null and cas.test\_head\_name is not null

)

,

-- get those mspp where QT missing on the top-equip

QT\_MISSING\_ON\_MSPP\_EQ as

(

select NE\_INST\_ID

from GET\_QT\_ON\_NE ms

where ms.test\_head\_name is null

)

,

-- if QT missing on the equip, check if QT found on every ES64 / EXX card

-- if valid QT not present on ALL ES64/EXX cards then it shld show up in this list

NO\_QT\_ON\_MSPP\_CI\_OR\_EQ as

(

select distinct qtm.ne\_inst\_id --, cas.ATTR\_VALUE qt\_name

from QT\_MISSING\_ON\_MSPP\_EQ qtm

left outer join GET\_QT\_ON\_CARDS gqc

on qtm.NE\_INST\_ID = gqc.ne\_INST\_ID

where gqc.card\_inst\_id is null

)

,

GUD\_QT\_ON\_MSPP as

(

-- all MSPP

select NE\_INST\_ID

from GET\_QT\_ON\_NE

minus

-- minus all MSPP that dont have any MSPP on eq or card

select NE\_INST\_ID

from NO\_QT\_ON\_MSPP\_CI\_OR\_EQ

minus

-- minus all MSPP that have valid QT on both eq and card

select NE\_INST\_ID

from ERR\_VALID\_QT\_ON\_CI\_AND\_EQ

)

select ne\_inst\_id, 1 issue\_code

from GUD\_QT\_ON\_MSPP

union

-- no QT on the device

select NE\_INST\_ID, 0

from NO\_QT\_ON\_MSPP\_CI\_OR\_EQ

union

-- minus all MSPP that dont have valid QT on card and EQ

select NE\_INST\_ID, issue\_code

from ERR\_VALID\_QT\_ON\_CI\_AND\_EQ

;

sql\_stmt varchar2(2000);

BEGIN

-- if (QT is there only MSPP top-equip or on all cards of certain types) = 1

-- if (no QT on top-equip and on all no cards) = 0

-- if (QT on BOTH top-equip and on all/some cards) = 2

-- if (QT on BOTH top-equip and on all/some cards and different) = 3

-- identify those here

sql\_stmt := 'update xng\_mspp\_devices\_wk xmd set issue\_code = :issue\_code where ne\_inst\_id = :ne\_inst\_id';

for rec in MSPP\_QT\_ISSUES

loop

-- update the MSPP top-equip to say it QT\_issues

execute immediate sql\_stmt using rec.issue\_code, rec.ne\_inst\_id;

end loop;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in update\_mspp\_qt\_issues(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end update\_mspp\_qt\_issues;

/\*

store if

a\_side of the vlan is going to the mtso

z\_side of the vlan is going to the mtso

does parent path have any mspp ports

does parent path have any ngmls ports

does parent path have multiple CSRs

\*/

procedure qt\_audit is

sql\_stmt varchar2(32767);

BEGIN

sql\_stmt := 'truncate table ngt\_qt\_ne\_vs\_xng\_audit\_wk';

EXECUTE IMMEDIATE sql\_stmt;

-- insert all good test\_heads i.e. ones found on some eq or card

sql\_stmt := 'insert into ngt\_qt\_ne\_vs\_xng\_audit\_wk columns (ne\_TEST\_HEAD\_NAME, eq\_test\_head\_name, equip\_inst\_id

, equip\_type, match\_code, match\_status

, card\_inst\_id, card\_test\_head\_name)

-- match test head names found on MSPP to discovered test heads

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME, ms.TEST\_HEAD\_NAME eq\_test\_head\_name, ms.ne\_inst\_id

, ''MSPP'' equip\_type, ''BOTH'' match\_code, ''Found on NE'' match\_status

, null, null

from DACS\_QT\_TEST\_HEADS qt

join xng\_mspp\_devices\_wk ms

on upper(qt.QT\_NAME) = ms.TEST\_HEAD\_NAME

where issue\_code = 1

and ms.EQ\_CLASS\_TYPE = ''NE''

union

-- match test head names found on ngMLS to discovered test heads

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME, ng.TEST\_HEAD\_NAME eq\_test\_head\_name, ng.equip\_inst\_id

, ''NGMLS'' equip\_type, ''BOTH'' match\_code, ''Found on NE'' match\_status

, null, null

from DACS\_QT\_TEST\_HEADS qt

join xng\_ngMLS\_devices\_wk ng

on upper(qt.QT\_NAME) = ng.TEST\_HEAD\_NAME

where issue\_code = 1

union

-- discovered test head found only on card

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME, ms.TEST\_HEAD\_NAME eq\_test\_head\_name, ms.ne\_inst\_id

, ''MSPP'' equip\_type, ''BOTH'' match\_code, ''Found on card'' match\_status

, ci.card\_inst\_id, ci.test\_head\_name card\_test\_head\_name

from DACS\_QT\_TEST\_HEADS qt

join xng\_vlan\_mspp\_map\_wk ci

on upper(qt.QT\_NAME) = ci.TEST\_HEAD\_NAME

join xng\_mspp\_devices\_wk ms

on ci.EQUIP\_INST\_ID = ms.ne\_inst\_id

where issue\_code = 1

union

-- discovered test head found only on card

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME, ng.TEST\_HEAD\_NAME eq\_test\_head\_name, ng.equip\_inst\_id

, ''NGMLS'' equip\_type, ''BOTH'' match\_code, ''Found on card'' match\_status

, ci.card\_inst\_id, ci.test\_head\_name card\_test\_head\_name

from DACS\_QT\_TEST\_HEADS qt

join xng\_vlan\_ngmls\_map\_wk ci

on upper(qt.QT\_NAME) = ci.TEST\_HEAD\_NAME

join xng\_ngmls\_devices\_wk ng

on ci.EQUIP\_INST\_ID = ng.equip\_inst\_id

where issue\_code = 1

union

-- same test head name found on both MSPP eq and cards to discovered test heads

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME,ms.TEST\_HEAD\_NAME eq\_test\_head\_name, ms.ne\_inst\_id

, ''MSPP'' equip\_type, ''BOTH'' match\_code, ''Found on both equip/card'' match\_status

, ci.card\_inst\_id, ci.test\_head\_name card\_test\_head\_name

from DACS\_QT\_TEST\_HEADS qt

join xng\_vlan\_mspp\_map\_wk ci

on upper(qt.QT\_NAME) = ci.TEST\_HEAD\_NAME

join xng\_mspp\_devices\_wk ms

on ci.EQUIP\_INST\_ID = ms.ne\_inst\_id

where issue\_code = 2

union

-- different test head names found on both MSPP eq and cards to discovered test heads

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME,ms.TEST\_HEAD\_NAME eq\_test\_head\_name, ms.ne\_inst\_id

, ''MSPP'' equip\_type, ''BOTH'' match\_code, ''Different test head name on equip/card'' match\_status

, ci.card\_inst\_id, ci.test\_head\_name card\_test\_head\_name

from DACS\_QT\_TEST\_HEADS qt

join xng\_vlan\_mspp\_map\_wk ci

on upper(qt.QT\_NAME) = ci.TEST\_HEAD\_NAME

join xng\_mspp\_devices\_wk ms

on ci.EQUIP\_INST\_ID = ms.ne\_inst\_id

where issue\_code = 3

union

-- match test head names found on ngMLS/cards to discovered test heads

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME, ng.TEST\_HEAD\_NAME eq\_test\_head\_name, ng.equip\_inst\_id

, ''NGMLS'' equip\_type, ''BOTH'' match\_code, ''Found on both equip/card'' match\_status

, ci.card\_inst\_id, ng.test\_head\_name card\_test\_head\_name

from DACS\_QT\_TEST\_HEADS qt

join xng\_vlan\_ngMLS\_map\_wk ci

on upper(qt.QT\_NAME) = ci.TEST\_HEAD\_NAME

join xng\_ngMLS\_devices\_wk ng

on ci.EQUIP\_INST\_ID = ng.equip\_inst\_id

where issue\_code = 2

union

-- match test head names found on ngMLS/cards to discovered test heads

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME, ng.TEST\_HEAD\_NAME eq\_test\_head\_name, ng.equip\_inst\_id

, ''NGMLS'' equip\_type, ''BOTH'' match\_code, ''Different test head name on equip/card'' match\_status

, ci.card\_inst\_id, ng.test\_head\_name card\_test\_head\_name

from DACS\_QT\_TEST\_HEADS qt

join xng\_vlan\_ngMLS\_map\_wk ci

on upper(qt.QT\_NAME) = ci.TEST\_HEAD\_NAME

join xng\_ngMLS\_devices\_wk ng

on ci.EQUIP\_INST\_ID = ng.equip\_inst\_id

where issue\_code = 3

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- add QT 'NE Only' Test heads

sql\_stmt := 'insert into ngt\_qt\_ne\_vs\_xng\_audit\_wk columns ( NE\_TEST\_HEAD\_NAME, match\_code, match\_status)

select upper(qt.QT\_NAME) TEST\_HEAD\_NAME, ''NE Only'', ''Not found in Granite''

from DACS\_QT\_TEST\_HEADS qt

minus

select aud.card\_TEST\_HEAD\_NAME, ''NE Only'', ''Not found in Granite''

from ngt\_qt\_ne\_vs\_xng\_audit\_wk aud

minus

select aud.eq\_TEST\_HEAD\_NAME, ''NE Only'', ''Not found in Granite''

from ngt\_qt\_ne\_vs\_xng\_audit\_wk aud

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- if QT found on 2 distinct equip then it is an issue

sql\_stmt := 'update ngt\_qt\_ne\_vs\_xng\_audit\_wk aud set match\_code = ''BOTH''

, match\_status = ''QT found on multiple equip''

where exists (

WITH

EQ\_TEST\_HEAD as

(

select distinct equip\_inst\_id, ne\_test\_head\_name

from ngt\_qt\_ne\_vs\_xng\_audit\_wk aud1

)

select ne\_test\_head\_name, count(1) cnt

from EQ\_TEST\_HEAD aud1

where aud1.ne\_test\_head\_name = aud.ne\_test\_head\_name

group by ne\_test\_head\_name

having count(1) > 1

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in qt\_audit(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

PROCEDURE load\_qt\_summary\_by\_region is

sql\_stmt varchar2(32767);

BEGIN

sql\_stmt := 'truncate table ngt\_qt\_reg\_summ\_wk';

EXECUTE IMMEDIATE sql\_stmt;

-- insert all good test\_heads i.e. ones found on some eq or card

sql\_stmt := 'insert into ngt\_qt\_reg\_summ\_wk select \* from

(

WITH

DOMAINS as

(

select area, region

from domains\_regional\_reporting drr

where REGION not in (''NNO'', ''OSS'')

union

select ''unknown'', ''unknown''

from dual

)

,

CLLI\_DOMAINS as

(

select distinct substr(clli, 1, 6) clli\_6, area, region

from clli\_domain\_map\_v cdmv

)

,

TEST\_HEAD\_DOMAIN as

(

select distinct upper(th.qt\_name) test\_head\_name, nvl(area, ''unknown'') area, nvl(region, ''unknown'') region

from DACS\_QT\_TEST\_HEADS th

left outer join CLLI\_DOMAINS cdmv

on cdmv.clli\_6 = substr(upper(th.qt\_name), 1, 6)

)

,

GUD\_MATCH as

(

select distinct match\_code, ne\_test\_head\_name

from NGT\_QT\_NE\_VS\_XNG\_AUDIT\_WK ngt

where match\_code = ''BOTH''

)

select drr.area, drr.region, count(th.test\_head\_name) total, count(gud.ne\_test\_head\_name) good\_match

, round(count(gud.ne\_test\_head\_name) / decode(count(th.test\_head\_name),0, 1, count(th.test\_head\_name)) \*100, 2) compliance

from DOMAINS drr

left outer join TEST\_HEAD\_DOMAIN th

on drr.region = th.region

left outer join GUD\_MATCH gud

on th.test\_head\_name = gud.ne\_test\_head\_name

group by rollup (drr.area, drr.region)

order by area, region

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_qt\_summary\_by\_region(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

/\*

for every VLAN check

if VLAN is from spoke to a hub then dont process further ???

atleast one parent lands in a MTSO site.

it may have MSPP(s) and has to have ngMLS has. In this case QT has to be on the MSPP

container has a valid QT or

all cards of certain card\_types have valid QT

List of card types per device

9500: FC9565EXX1

1675: ES64

1678: ES64

if no MSPP(s) then the ngMLS has to have the QT

container/shelf has a valid QT or

all cards of certain card\_types have valid QT

List of card types per device

9010K: ??

7750: ??

in case of spoke CSR the VLAN WILL NOT go to MTSO and thus NO ngMLS, no MSPP etc. These VLAN will stop at the CSR.

need to check that. that is not yet coded ????

\*/

procedure load\_ebh\_summary\_by\_region is

sql\_stmt varchar2(32000);

begin

-- drop the table

sql\_stmt:='truncate table COMP\_EBH\_SUMMARY\_BY\_REGION\_WK';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='truncate table COMP\_EBH\_DETAILS\_BY\_CLLI\_WK';

EXECUTE IMMEDIATE sql\_stmt;

--this is the same query used by the details page in the report

sql\_stmt:='insert into COMP\_EBH\_DETAILS\_BY\_CLLI\_WK columns ( REGION,LEAF\_DOMAIN\_NAME,CSR\_DEVICE\_NAME,CSR\_IP,XNG\_CSR\_DEVICE\_NAME,MATCH\_CODE,

NE\_STATUS,NE\_SOURCE,ISSUES,IS\_CRITICAL,XNG\_EQUIPMENT,EQUIP\_INST\_ID,SITE\_HUM\_ID,SITE\_INST\_ID,CSR\_LIVE\_IN\_XNG,

GIGE\_PATH\_HUM\_ID, GIGE\_PATH\_INST\_ID, GIGE\_HAS\_ISSUES, GIGE\_LIVE\_IN\_XNG )

WITH ALL\_NE\_CSRs

AS (SELECT DISTINCT aud.csr\_id,

aud.ne\_csr\_device\_name csr\_device\_name,

cd.ne\_status,

cd.ne\_source

FROM xng\_reports.hpov\_vs\_xng\_csr\_audit\_wk aud

JOIN xng\_reports.CSR\_DEVICES\_wk cd ON aud.CSR\_ID = cd.CSR\_ID),

ALL\_CSR\_ISSUES

AS (SELECT DISTINCT aud.csr\_id,

ci.DESCRIPTION,

cai.xng\_equip\_inst\_id,

ci.is\_critical

FROM xng\_reports.hpov\_vs\_xng\_csr\_audit\_wk aud

JOIN xng\_reports.CSR\_AUDIT\_ISSUE\_wk cai

ON aud.CSR\_ID = cai.CSR\_ID

JOIN xng\_reports.csr\_issue ci

ON ci.CSR\_ISSUE\_ID = cai.CSR\_ISSUE\_ID

UNION

SELECT DISTINCT cai.csr\_id,

ci.DESCRIPTION,

cai.xng\_equip\_inst\_id,

ci.is\_critical

FROM xng\_reports.hpov\_vs\_xng\_csr\_audit\_wk aud

JOIN xng\_reports.CSR\_AUDIT\_ISSUE\_wk cai

ON aud.equip\_inst\_id = cai.xng\_equip\_inst\_id

JOIN xng\_reports.csr\_issue ci

ON ci.CSR\_ISSUE\_ID = cai.CSR\_ISSUE\_ID),

CLLI\_6\_AREA\_REGION

AS (SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, cdm.leaf\_DOMAIN\_INST\_ID

FROM xng\_reports.clli\_domain\_map cdm),

ALL\_CLLI\_FOR\_REGION

AS (SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, cdm.leaf\_DOMAIN\_INST\_ID

FROM xng\_reports.clli\_domain\_map cdm),

gige\_issues

AS (SELECT DISTINCT gvai.GIGE\_VLAN\_INST\_ID, xcgp.EQUIP\_INST\_ID

FROM xng\_reports.gige\_vlan\_audit\_issue\_wk gvai,

xng\_reports.XNG\_CSR\_GIGE\_PATHS\_wk xcgp

WHERE gvai.GIGE\_VLAN\_ISSUE\_ID IN -- (5, 10, 11, 12, 13, 15)

(SELECT gvi.GIGE\_VLAN\_ISSUE\_ID

FROM xng\_reports.gige\_vlan\_issue gvi

WHERE gvi.STORED\_VLAN\_INST\_ID = ''N'' and gvi.IS\_CRITICAL=''Y'' )

and (gvai.GIGE\_VLAN\_INST\_ID = xcgp.circ\_path\_inst\_id or

((gvai.GIGE\_VLAN\_INST\_ID= xcgp.next\_path\_inst\_id) and xcgp.circ\_path\_inst\_id is null ))

union

select distinct map.GIGE\_PATH\_INST\_ID, p.EQUIP\_INST\_ID from

xng\_reports.gige\_vlan\_audit\_issue\_wk gv

,xng\_reports.XNG\_GIGE\_VLAN\_MAP\_wk map

,xng\_reports.XNG\_CSR\_GIGE\_PATHS\_wk p

where

gv.GIGE\_VLAN\_issue\_ID in -- (5, 10, 11, 12, 13, 15)

(select gvi.GIGE\_VLAN\_ISSUE\_ID from xng\_reports.gige\_vlan\_issue gvi

where gvi.STORED\_VLAN\_INST\_ID = ''Y'' and gvi.IS\_CRITICAL=''Y'' )

and map.VLAN\_INST\_ID = gv.GIGE\_VLAN\_INST\_ID

and ( map.GIGE\_PATH\_INST\_ID = p.circ\_path\_inst\_id

or

((map.GIGE\_path\_INST\_ID = p.next\_path\_inst\_id ) and p.circ\_path\_inst\_id is null ))

),

ALL\_DETAILS as

(

select distinct

nvl(region, ''unknown'') region

, dlr.domain\_name leaf\_domain\_name

, cd.CSR\_DEVICE\_NAME

, dev.csr\_ip

, aud.xng\_CSR\_DEVICE\_NAME xng\_csr\_device\_name

, aud.match\_code

, cd.ne\_status

, cd.ne\_source

, cai.description issues

, cai.is\_critical

, aud.DESCR xng\_equipment

, aud.equip\_inst\_id

, si.site\_hum\_id

, si.site\_inst\_id

, aud.live\_in\_xng csr\_live\_in\_xng

, case when cpi.circ\_path\_hum\_id is null then npi.circ\_path\_hum\_id

else cpi.circ\_path\_hum\_id

end gige\_path\_hum\_id

, case when cpi.circ\_path\_inst\_id is null then npi.circ\_path\_inst\_id

else cpi.circ\_path\_inst\_id

end gige\_path\_inst\_id

, case when gi.gige\_vlan\_inst\_id > 0 then ''Y''

when gi.gige\_vlan\_inst\_id is null and cpi.CIRC\_PATH\_INST\_ID is null and npi.CIRC\_PATH\_INST\_ID is null then ''''

when gi.gige\_vlan\_inst\_id is null then ''N''

end gige\_has\_issues

,xcgp.gige\_live\_in\_xng

from ALL\_NE\_CSRs cd

left outer join CLLI\_6\_AREA\_REGION cdm

on cdm.CLLI\_6 = substr(cd.CSR\_DEVICE\_NAME, 1, 6)

left outer join xng\_reports.domains\_leaf\_reporting dlr

on dlr.DOMAIN\_INST\_ID = cdm.LEAF\_DOMAIN\_INST\_ID

left outer join xng\_reports.hpov\_vs\_xng\_csr\_audit\_wk aud

on aud.CSR\_ID = cd.CSR\_ID

left outer join xng\_reports.CSR\_DEVICES\_wk dev

on dev.CSR\_ID = aud.CSR\_ID

left join xng\_reports.XNG\_CSR\_GIGE\_PATHS\_wk xcgp

on xcgp.EQUIP\_INST\_ID = aud.equip\_inst\_id

left join vzwnet.circ\_path\_inst cpi

on cpi.circ\_path\_inst\_id = xcgp.CIRC\_PATH\_INST\_ID

left join vzwnet.circ\_path\_inst npi

on npi.circ\_path\_inst\_id = xcgp.NEXT\_PATH\_INST\_ID

left outer join ALL\_CSR\_ISSUES cai

on ( cai.CSR\_ID = cd.csr\_id or cai.xng\_equip\_inst\_id = aud.EQUIP\_INST\_ID)

left outer join vzwnet.equip\_inst ei on

aud.EQUIP\_INST\_ID = ei.EQUIP\_INST\_ID

left outer join vzwnet.site\_inst si on

ei.SITE\_INST\_ID = si.site\_inst\_id

left outer join gige\_issues gi on

gi.equip\_inst\_id = aud.equip\_inst\_id

and (gi.gige\_vlan\_inst\_id = CPI.CIRC\_PATH\_INST\_ID or gi.gige\_vlan\_inst\_id = npi.CIRC\_PATH\_INST\_ID)

join all\_clli\_for\_region car on

car.clli\_6 = cdm.clli\_6

union

select distinct dlr.region

, dlr.DOMAIN\_NAME leaf\_domain\_name

, aud.ne\_csr\_device\_name CSR\_DEVICE\_NAME

, '''' csr\_ip

, aud.xng\_CSR\_DEVICE\_NAME xng\_csr\_device\_name

, aud.match\_code

, cd.ne\_status

, cd.ne\_source

, cai.description issues

, cai.is\_critical

, aud.DESCR xng\_equipment

, aud.equip\_inst\_id

, si.site\_hum\_id

, si.site\_inst\_id

, aud.live\_in\_xng csr\_live\_in\_xng

, case when cpi.circ\_path\_hum\_id is null then npi.circ\_path\_hum\_id

else cpi.circ\_path\_hum\_id

end gige\_path\_hum\_id

, case when cpi.circ\_path\_inst\_id is null then npi.circ\_path\_inst\_id

else cpi.circ\_path\_inst\_id

end gige\_path\_inst\_id

, case when gi.gige\_vlan\_inst\_id > 0 then ''Y''

when gi.gige\_vlan\_inst\_id is null and cpi.CIRC\_PATH\_INST\_ID is null and npi.CIRC\_PATH\_INST\_ID is null then ''''

when gi.gige\_vlan\_inst\_id is null then ''N''

end gige\_has\_issues

,xcgp.gige\_live\_in\_xng

from xng\_reports.hpov\_vs\_xng\_csr\_audit\_wk aud

left outer join xng\_reports.CSR\_DEVICES\_wk cd

on aud.CSR\_ID = cd.CSR\_ID

join vzwnet.equip\_domain\_map cdm

on cdm.equip\_inst\_id = aud.EQUIP\_INST\_ID

join xng\_reports.domains\_leaf\_reporting dlr

on dlr.DOMAIN\_INST\_ID = cdm.DOMAIN\_INST\_ID

left join xng\_reports.XNG\_CSR\_GIGE\_PATHS\_wk xcgp

on xcgp.EQUIP\_INST\_ID = aud.EQUIP\_INST\_ID

left join vzwnet.circ\_path\_inst cpi

on cpi.circ\_path\_inst\_id = xcgp.CIRC\_PATH\_INST\_ID

left join vzwnet.circ\_path\_inst npi

on npi.circ\_path\_inst\_id = xcgp.NEXT\_PATH\_INST\_ID

left outer join ALL\_CSR\_ISSUES cai

on cai.xng\_equip\_inst\_ID = aud.equip\_inst\_id

join vzwnet.equip\_inst ei on

aud.EQUIP\_INST\_ID = ei.EQUIP\_INST\_ID

join vzwnet.site\_inst si on

ei.SITE\_INST\_ID = si.site\_inst\_id

left outer join gige\_issues gi on

gi.equip\_inst\_id = aud.equip\_inst\_id

and (gi.gige\_vlan\_inst\_id = CPI.CIRC\_PATH\_INST\_ID or gi.gige\_vlan\_inst\_id = npi.CIRC\_PATH\_INST\_ID)

where match\_code = ''Xng Only''

union

select distinct dlr.region

, dlr.DOMAIN\_NAME leaf\_domain\_name

, aud.ne\_csr\_device\_name CSR\_DEVICE\_NAME

, '''' csr\_ip

, aud.xng\_CSR\_DEVICE\_NAME xng\_csr\_device\_name

, aud.match\_code

, '''' ne\_status

, '''' ne\_source

, cai.description issues

, cai.is\_critical

, aud.DESCR xng\_equipment

, aud.equip\_inst\_id

, si.site\_hum\_id

, si.site\_inst\_id

, aud.live\_in\_xng csr\_live\_in\_xng

, case when cpi.circ\_path\_hum\_id is null then npi.circ\_path\_hum\_id

else cpi.circ\_path\_hum\_id

end gige\_path\_hum\_id

, case when cpi.circ\_path\_inst\_id is null then npi.circ\_path\_inst\_id

else cpi.circ\_path\_inst\_id

end gige\_path\_inst\_id

, case when gi.gige\_vlan\_inst\_id > 0 then ''Y''

when gi.gige\_vlan\_inst\_id is null and cpi.CIRC\_PATH\_INST\_ID is null and npi.CIRC\_PATH\_INST\_ID is null then ''''

when gi.gige\_vlan\_inst\_id is null then ''N''

end gige\_has\_issues ,xcgp.gige\_live\_in\_xng

from xng\_reports.hpov\_vs\_xng\_csr\_audit\_wk aud

join vzwnet.equip\_domain\_map cdm

on cdm.equip\_inst\_id = aud.EQUIP\_INST\_ID

join xng\_reports.domains\_leaf\_reporting dlr

on dlr.DOMAIN\_INST\_ID = cdm.DOMAIN\_INST\_ID

left join xng\_reports.XNG\_CSR\_GIGE\_PATHS\_wk xcgp

on xcgp.EQUIP\_INST\_ID = aud.EQUIP\_INST\_ID

left join vzwnet.circ\_path\_inst cpi

on cpi.circ\_path\_inst\_id = xcgp.CIRC\_PATH\_INST\_ID

left join vzwnet.circ\_path\_inst npi

on npi.circ\_path\_inst\_id = xcgp.NEXT\_PATH\_INST\_ID

left outer join ALL\_CSR\_ISSUES cai

on cai.xng\_equip\_inst\_ID = aud.equip\_inst\_id

join vzwnet.equip\_inst ei on

aud.EQUIP\_INST\_ID = ei.EQUIP\_INST\_ID

join vzwnet.site\_inst si on

ei.SITE\_INST\_ID = si.site\_inst\_id

left outer join gige\_issues gi on

gi.equip\_inst\_id = aud.equip\_inst\_id

and (gi.gige\_vlan\_inst\_id = CPI.CIRC\_PATH\_INST\_ID or gi.gige\_vlan\_inst\_id = npi.CIRC\_PATH\_INST\_ID)

left outer join ALL\_CLLI\_FOR\_REGION car

on car.CLLI\_6 = substr(xng\_CSR\_DEVICE\_NAME, 1, 6)

where match\_code = ''Xng Only'' and car.clli\_6 is null

)

select \* from all\_details'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating COMP\_EBH\_DETAILS\_BY\_CLLI\_WK');

-- LR 10/28/2014

--if a CSR has 2 giges one live without issues and the other non-live with issues then should be considered part of complete pathset

--if a CSR has 2 giges and both live one without issues and the other with issues, then it is still not good pathset.

sql\_stmt:= 'insert into COMP\_EBH\_SUMMARY\_BY\_REGION\_WK columns

(AREA,REGION,TOT\_CSR,BOTH\_LIVE\_CSR,BOTH\_NLIVE\_CSR,DUP\_LIVE\_CSR,LIVE\_CSR\_W\_LIVE\_GIGE,

LIVE\_CSR\_W\_NLIVE\_GIGE,LIVE\_CSR\_NO\_GIGE,LIVE\_CSR\_W\_EBH\_PATHSET,DEN\_FOR\_COMPLIANCE\_PER, COMPLIANCE)

WITH

ALL\_NE\_CSRs as

(

select distinct aud.csr\_id, aud.ne\_csr\_device\_name csr\_device\_name

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join CSR\_DEVICES\_WK cd on aud.csr\_id = cd.csr\_id

where aud.ne\_csr\_device\_name is not null

and cd.ne\_status = ''LIVE''

),

UNQ\_CLLI as

(

select distinct nvl(area, ''unknown'') area, nvl(region, ''unknown'') region

, dlr.domain\_name, substr(cdm.CLLI, 1, 6 ) clli

from xng\_reports.clli\_domain\_map cdm

join xng\_reports.domains\_leaf\_reporting dlr

on dlr.DOMAIN\_INST\_ID = cdm.LEAF\_DOMAIN\_INST\_ID

) ,

TOTAL\_CSRs\_PER\_REGION as

(

select nvl(area, ''unknown'') area, nvl(region, ''unknown'') region, clli

, cd.csr\_id

from ALL\_NE\_CSRs cd

left outer join UNQ\_CLLI cdm

on substr(cdm.CLLI, 1, 6) = substr(cd.CSR\_DEVICE\_NAME, 1, 6)

),

LIVE\_DUP\_CSRs as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join CSR\_AUDIT\_ISSUE\_WK cai

on aud.CSR\_ID = cai.CSR\_ID

join CSR\_DEVICES\_WK cd

on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG = ''Y''

and cai.CSR\_ISSUE\_ID = 1 -- Duplicate CSRs in Xng

-- all dups are turned into ''NE Only''

and match\_code = ''NE Only''

and cd.ne\_status = ''LIVE''

)

,

-- Discovered CSRs match uniquely in Xng (Both Live)

LIVE\_CSRs as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join CSR\_DEVICES\_WK cd on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG = ''Y''

and aud.MATCH\_CODE=''BOTH''

and cd.ne\_status = ''LIVE''

minus

select distinct aud.csr\_id

from LIVE\_DUP\_CSRs aud

)

,

-- Discovered CSRs match uniquely in Xng (Both Others)

NON\_LIVE\_CSRs as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join CSR\_DEVICES\_WK cd

on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG <> ''Y''

and aud.MATCH\_CODE=''BOTH''

and cd.ne\_status=''LIVE''

)

,

LIVE\_UNQ\_CSRs\_W\_L\_GIGE as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join XNG\_CSR\_GIGE\_PATHS\_WK xcgp

on xcgp.EQUIP\_INST\_ID = aud.equip\_inst\_id

join CSR\_DEVICES\_WK cd

on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG = ''Y''

-- all CSR with no issues only will be ''BOTH''

-- if Dup then it will be NE Only and not BOTH

and match\_code = ''BOTH''

and (xcgp.CIRC\_PATH\_INST\_ID is not null

or

xcgp.NEXT\_PATH\_INST\_ID is not null)

and xcgp.GIGE\_LIVE\_IN\_XNG = ''Y''

and cd.ne\_status=''LIVE''

)

,

LIVE\_UNQ\_CSRs\_W\_NL\_GIGE as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join XNG\_CSR\_GIGE\_PATHS\_WK xcgp

on xcgp.MATCHED\_EQUIP\_INST\_ID = aud.equip\_inst\_id

join CSR\_DEVICES\_WK cd

on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG = ''Y''

-- all CSR with no issues only will be ''BOTH''

and match\_code = ''BOTH''

and (xcgp.CIRC\_PATH\_INST\_ID is not null

or

xcgp.NEXT\_PATH\_INST\_ID is not null)

and xcgp.GIGE\_LIVE\_IN\_XNG = ''N''

and cd.ne\_status=''LIVE''

)

,

LIVE\_UNQ\_CSRs\_W\_NO\_GIGE as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

left outer join XNG\_CSR\_GIGE\_PATHS\_WK xcgp

on xcgp.MATCHED\_EQUIP\_INST\_ID = aud.equip\_inst\_id

join CSR\_DEVICES\_WK cd

on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG = ''Y''

-- all CSR with no issues only will be ''BOTH''

and match\_code = ''BOTH''

and xcgp.CIRC\_PATH\_INST\_ID is null

and xcgp.NEXT\_PATH\_INST\_ID is null

and cd.ne\_status=''LIVE''

)

,

GIGE\_VLAN\_CRITICAL\_ISSUES as

(

select gvi.GIGE\_VLAN\_ISSUE\_ID

from xng\_reports.gige\_vlan\_issue gvi

where gvi.STORED\_VLAN\_INST\_ID = ''N''

and gvi.IS\_CRITICAL=''Y''

)

,

LIVE\_GIGE\_W\_ISSUES as

(

select distinct hpov.csr\_id

from

GIGE\_VLAN\_AUDIT\_ISSUE\_WK gv

join XNG\_CSR\_GIGE\_PATHS\_WK p

on gv.GIGE\_VLAN\_INST\_ID = p.circ\_path\_inst\_id

or

(gv.GIGE\_VLAN\_INST\_ID = p.next\_path\_inst\_id

and p.circ\_path\_inst\_id is null

) -- LR 01//24/2012

join HPOV\_VS\_XNG\_CSR\_AUDIT\_WK hpov

on hpov.EQUIP\_INST\_ID = p.EQUIP\_INST\_ID

join LIVE\_CSRs csr

on csr.csr\_id = hpov.CSR\_ID

join CSR\_DEVICES\_WK cd

on cd.csr\_id = hpov.csr\_id

join GIGE\_VLAN\_CRITICAL\_ISSUES ciss

on gv.GIGE\_VLAN\_issue\_ID = ciss.gige\_vlan\_issue\_id

where cd.ne\_status=''LIVE''

union

select distinct hpov.csr\_id

from GIGE\_VLAN\_AUDIT\_ISSUE\_WK gv

, XNG\_GIGE\_VLAN\_MAP\_WK map

, XNG\_CSR\_GIGE\_PATHS\_WK p

, live\_csrs csr

, HPOV\_VS\_XNG\_CSR\_AUDIT\_WK hpov

, CSR\_DEVICES\_WK cd

where

gv.GIGE\_VLAN\_issue\_ID in

(select gvi.GIGE\_VLAN\_ISSUE\_ID

from xng\_reports.gige\_vlan\_issue gvi

where gvi.STORED\_VLAN\_INST\_ID = ''Y''

and gvi.IS\_CRITICAL=''Y'')

--(5, 10, 11, 12, 13, 15)

and map.VLAN\_INST\_ID = gv.GIGE\_VLAN\_INST\_ID

-- LR 09/01/2015

and ( map.GIGE\_PATH\_INST\_ID = p.circ\_path\_inst\_id

or

((map.GIGE\_path\_INST\_ID = p.next\_path\_inst\_id ) and p.circ\_path\_inst\_id is null ))

/\*and map.GIGE\_PATH\_INST\_ID in (p.circ\_path\_inst\_id, p.next\_path\_inst\_id) \*/

-- LR 09/01/2015

and hpov.EQUIP\_INST\_ID = p.EQUIP\_INST\_ID

and csr.csr\_id = hpov.CSR\_ID

and cd.csr\_id = hpov.csr\_id

and cd.ne\_status=''LIVE''

union

select distinct hpov.csr\_id

from

CSR\_AUDIT\_ISSUE\_WK cai

,live\_csrs csr

, HPOV\_VS\_XNG\_CSR\_AUDIT\_WK hpov

,CSR\_DEVICES\_WK cd

where cai.CSR\_ISSUE\_ID in( 4,5) -- csr with no gige, insuff VLANs

and hpov.EQUIP\_INST\_ID = cai.xng\_equip\_inst\_id

and csr.csr\_id = hpov.CSR\_ID

and cd.csr\_id = hpov.CSR\_ID

and cd.ne\_status=''LIVE''

)

,

DENOMINATOR\_FOR\_COM\_PER as

(

select distinct csr\_id from all\_ne\_csrs

minus

select distinct csr\_id from non\_live\_csrs

)

,

csr\_w\_1\_gige\_no\_iss as --LR 10/28/2014

(

select distinct te.equip\_inst\_id

from COMP\_EBH\_DETAILS\_BY\_CLLI\_WK te

where te.gige\_has\_issues=''N''

and gige\_live\_in\_xng=''Y''

and csr\_device\_name is not null

intersect

select distinct te.equip\_inst\_id

from COMP\_EBH\_DETAILS\_BY\_CLLI\_WK te

where te.gige\_has\_issues=''Y''

and gige\_live\_in\_xng<>''Y''

and csr\_device\_name is not null

),

LIVE\_CSRs\_W\_NO\_ISSUES as

(

(

select csr\_id

from live\_csrs

minus

-- get all live\_csrs\_with\_issues

select csr\_id

from LIVE\_GIGE\_W\_ISSUES

)

union

select csr\_id

from csr\_w\_1\_gige\_no\_iss csr, hpov\_vs\_xng\_csr\_audit\_wk wk

where csr.equip\_inst\_id = WK.EQUIP\_INST\_ID

)

select area, region, nvl( count(csr.csr\_id), 0) tot\_csr

, count(l.csr\_id) both\_live\_csr

, count(nl.csr\_id) both\_non\_live\_csr

-- , count(miss.csr\_id) ne\_only

, count(dup.csr\_id) dup\_live\_csr

, count(w\_gige.csr\_id) live\_csr\_w\_live\_gige

, count(w\_nl\_gige.csr\_id) live\_csr\_w\_nlive\_gige

, count(no\_gige.csr\_id) live\_csr\_no\_gige

, count(no\_issues.csr\_id) live\_csr\_ebh\_pathset

, count(den.csr\_id) den\_for\_comper

, case when count(den.csr\_id) <> 0 then

round((count(no\_issues.csr\_id)/count(den.csr\_id)), 4)\*100

else 0

end complicance

from TOTAL\_CSRs\_PER\_REGION csr

left outer join LIVE\_CSRs l

on l.csr\_id = csr.csr\_id

left outer join NON\_LIVE\_CSRs nl

on nl.csr\_id = csr.csr\_id

--left outer join MISSING\_IN\_XNG miss

-- on miss.csr\_id = csr.csr\_id

left outer join LIVE\_DUP\_CSRs dup

on dup.csr\_id = csr.csr\_id

left outer join LIVE\_UNQ\_CSRs\_W\_L\_GIGE w\_gige

on w\_gige.csr\_id = csr.csr\_id

left outer join LIVE\_UNQ\_CSRs\_W\_NL\_GIGE w\_nl\_gige

on w\_nl\_gige.csr\_id = csr.csr\_id

left outer join LIVE\_UNQ\_CSRs\_W\_NO\_GIGE no\_gige

on no\_gige.csr\_id = csr.csr\_id

left outer join LIVE\_CSRs\_W\_NO\_ISSUES no\_issues

on no\_issues.csr\_id = csr.csr\_id

left outer join DENOMINATOR\_FOR\_COM\_PER den

on den.csr\_id = csr.csr\_id

where AREA <> ''NNO''

group by rollup (area, region--, leaf\_domain\_inst\_id

)

order by area, region

--, leaf\_domain\_inst\_id

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating COMP\_EBH\_SUMMARY\_BY\_REGION\_WK');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_complete\_ebh\_summary\_by\_region(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_ebh\_summary\_by\_region;

procedure load\_csr\_summary\_by\_region is

sql\_stmt varchar2(32000);

begin

-- drop the table

sql\_stmt:='truncate table CSR\_SUMMARY\_BY\_REGION\_WK';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:= 'insert into CSR\_SUMMARY\_BY\_REGION\_WK columns

(AREA,REGION,TOT\_CSR,BOTH\_LIVE\_CSR,BOTH\_NLIVE\_CSR,DUP\_LIVE\_CSR,DEN\_FOR\_COMPER, COMPLIANCE)

WITH

ALL\_NE\_CSRs as

(

select distinct aud.csr\_id, aud.ne\_csr\_device\_name csr\_device\_name

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join CSR\_DEVICES\_WK cd on aud.csr\_id = cd.csr\_id

where aud.ne\_csr\_device\_name is not null

and cd.ne\_status = ''LIVE''

),

UNQ\_CLLI as

(

select distinct nvl(area, ''unknown'') area, nvl(region, ''unknown'') region

, dlr.domain\_name, substr(cdm.CLLI, 1, 6 ) clli

from xng\_reports.clli\_domain\_map cdm

join xng\_reports.domains\_leaf\_reporting dlr

on dlr.DOMAIN\_INST\_ID = cdm.LEAF\_DOMAIN\_INST\_ID

) ,

TOTAL\_CSRs\_PER\_REGION as

(

select nvl(area, ''unknown'') area, nvl(region, ''unknown'') region, clli

, cd.csr\_id

from ALL\_NE\_CSRs cd

left outer join UNQ\_CLLI cdm

on substr(cdm.CLLI, 1, 6) = substr(cd.CSR\_DEVICE\_NAME, 1, 6)

),

LIVE\_DUP\_CSRs as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join CSR\_AUDIT\_ISSUE\_WK cai

on aud.CSR\_ID = cai.CSR\_ID

join CSR\_DEVICES\_WK cd

on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG = ''Y''

and cai.CSR\_ISSUE\_ID = 1 -- Duplicate CSRs in Xng

-- all dups are turned into ''NE Only''

and match\_code = ''NE Only''

and cd.ne\_status = ''LIVE''

)

,

-- Discovered CSRs match uniquely in Xng (Both Live)

LIVE\_CSRs as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join CSR\_DEVICES\_WK cd on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG = ''Y''

and aud.MATCH\_CODE=''BOTH''

and cd.ne\_status = ''LIVE''

minus

select distinct aud.csr\_id

from LIVE\_DUP\_CSRs aud

)

,

-- Discovered CSRs match uniquely in Xng (Both Others)

NON\_LIVE\_CSRs as

(

select distinct aud.csr\_id

from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK aud

join CSR\_DEVICES\_WK cd

on aud.csr\_id = cd.csr\_id

where aud.LIVE\_IN\_XNG <> ''Y''

and aud.MATCH\_CODE=''BOTH''

and cd.ne\_status=''LIVE''

)

,

DENOMINATOR\_FOR\_COM\_PER as

(

select distinct csr\_id from all\_ne\_csrs

minus

select distinct csr\_id from non\_live\_csrs

)

select area, region, nvl( count(csr.csr\_id), 0) tot\_csr

, count(l.csr\_id) both\_live\_csr

, count(nl.csr\_id) both\_non\_live\_csr

-- , count(miss.csr\_id) ne\_only

, count(dup.csr\_id) dup\_live\_csr

, count(den.csr\_id) den\_for\_comper

, case when count(den.csr\_id) <> 0 then

round((count(l.csr\_id)/count(den.csr\_id)), 4)\*100

else 0

end compliance

from TOTAL\_CSRs\_PER\_REGION csr

left outer join LIVE\_CSRs l

on l.csr\_id = csr.csr\_id

left outer join NON\_LIVE\_CSRs nl

on nl.csr\_id = csr.csr\_id

left outer join LIVE\_DUP\_CSRs dup

on dup.csr\_id = csr.csr\_id

left outer join DENOMINATOR\_FOR\_COM\_PER den

on den.csr\_id = csr.csr\_id

where AREA <> ''NNO''

group by rollup (area, region--, leaf\_domain\_inst\_id

)

order by area, region

--, leaf\_domain\_inst\_id

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_SUMMARY\_BY\_REGION\_WK');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in load\_csr\_summary\_by\_region(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end load\_csr\_summary\_by\_region;

procedure move\_csr\_data\_from\_wk is

sql\_stmt varchar2(32000);

begin

sql\_stmt:='truncate table COMP\_EBH\_SUMMARY\_BY\_REGION';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into COMP\_EBH\_SUMMARY\_BY\_REGION select \* from COMP\_EBH\_SUMMARY\_BY\_REGION\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating COMP\_EBH\_SUMMARY\_BY\_REGION');

sql\_stmt:='truncate table CSR\_SUMMARY\_BY\_REGION';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into CSR\_SUMMARY\_BY\_REGION select \* from CSR\_SUMMARY\_BY\_REGION\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_SUMMARY\_BY\_REGION');

sql\_stmt:='truncate table HPOV\_VS\_XNG\_CSR\_AUDIT';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into HPOV\_VS\_XNG\_CSR\_AUDIT select \* from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating HPOV\_VS\_XNG\_CSR\_AUDIT');

sql\_stmt:='truncate table CSR\_MISSING\_IN\_HPOV';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into CSR\_MISSING\_IN\_HPOV select \* from CSR\_MISSING\_IN\_HPOV\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_MISSING\_IN\_HPOV');

sql\_stmt:='truncate table CSR\_AUDIT\_ISSUE';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into CSR\_AUDIT\_ISSUE select \* from CSR\_AUDIT\_ISSUE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_AUDIT\_ISSUE');

sql\_stmt:='truncate table GIGE\_VLAN\_AUDIT\_ISSUE';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into GIGE\_VLAN\_AUDIT\_ISSUE select \* from GIGE\_VLAN\_AUDIT\_ISSUE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating GIGE\_VLAN\_AUDIT\_ISSUE');

sql\_stmt:='truncate table XNG\_CSR\_PARSED';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_CSR\_PARSED select \* from XNG\_CSR\_PARSED\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_CSR\_PARSED');

sql\_stmt:='truncate table CSR\_DEVICES';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into CSR\_DEVICES select \* from CSR\_DEVICES\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_DEVICES');

sql\_stmt:='truncate table XNG\_CSR\_GIGE\_PATHS';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_CSR\_GIGE\_PATHS select \* from XNG\_CSR\_GIGE\_PATHS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_CSR\_GIGE\_PATHS');

sql\_stmt:='truncate table XNG\_CSR\_BTS\_PATHS';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_CSR\_BTS\_PATHS select \* from XNG\_CSR\_BTS\_PATHS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_CSR\_BTS\_PATHS');

sql\_stmt:='truncate table XNG\_GIGE\_DELIVERY\_TABLE';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_GIGE\_DELIVERY\_TABLE select \* from XNG\_GIGE\_DELIVERY\_TABLE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_GIGE\_DELIVERY\_TABLE');

sql\_stmt:='truncate table XNG\_GIGE\_PARENT\_MW';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_GIGE\_PARENT\_MW select \* from XNG\_GIGE\_PARENT\_MW\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_GIGE\_PARENT\_MW');

sql\_stmt:='truncate table XNG\_VLAN\_PARSED';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_VLAN\_PARSED select \* from XNG\_VLAN\_PARSED\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_PARSED');

sql\_stmt:='truncate table XNG\_GIGE\_VLAN\_MAP';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_GIGE\_VLAN\_MAP select \* from XNG\_GIGE\_VLAN\_MAP\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_GIGE\_VLAN\_MAP');

sql\_stmt:='truncate table XNG\_GIGE\_PARENTS';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_GIGE\_PARENTS select \* from XNG\_GIGE\_PARENTS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_GIGE\_PARENTS');

sql\_stmt:='truncate table XNG\_VLAN\_PARENTS';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_VLAN\_PARENTS select \* from XNG\_VLAN\_PARENTS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_PARENTS');

sql\_stmt:='truncate table XNG\_MSPP\_DEVICES';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_MSPP\_DEVICES select \* from XNG\_MSPP\_DEVICES\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_MSPP\_DEVICES');

sql\_stmt:='truncate table XNG\_NGMLS\_DEVICES';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_NGMLS\_DEVICES select \* from XNG\_NGMLS\_DEVICES\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_NGMLS\_DEVICES');

sql\_stmt:='truncate table XNG\_VLAN\_NGMLS\_MAP';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_VLAN\_NGMLS\_MAP select \* from XNG\_VLAN\_NGMLS\_MAP\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_NGMLS\_MAP');

sql\_stmt:='truncate table XNG\_VLAN\_MSPP\_MAP';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_VLAN\_MSPP\_MAP select \* from XNG\_VLAN\_MSPP\_MAP\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_MSPP\_MAP');

sql\_stmt:='truncate table XNG\_VLAN\_MSPP\_TAM';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into XNG\_VLAN\_MSPP\_TAM select \* from XNG\_VLAN\_MSPP\_TAM\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_MSPP\_TAM');

sql\_stmt:='delete from ngt\_qt\_ne\_vs\_xng\_audit';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into ngt\_qt\_ne\_vs\_xng\_audit select \* from ngt\_qt\_ne\_vs\_xng\_audit\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating ngt\_qt\_ne\_vs\_xng\_audit');

sql\_stmt:='delete from ngt\_qt\_reg\_summ';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into ngt\_qt\_reg\_summ select \* from ngt\_qt\_reg\_summ\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating ngt\_qt\_reg\_summ');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in move\_csr\_data\_from\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

WATCHDOG.logerror('CSR\_AUDIT',4000,SubStr('Error in move\_csr\_data\_from\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('CSR\_AUDIT','STATUS\_FAILURE','N');

--RAISE;

end move\_csr\_data\_from\_wk;

-- This procedure will be called by WD to Reinstate CSR-Audit

procedure move\_csr\_data\_from\_prod\_to\_wk is

sql\_stmt varchar2(32000);

begin

sql\_stmt:='delete from COMP\_EBH\_SUMMARY\_BY\_REGION\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into COMP\_EBH\_SUMMARY\_BY\_REGION\_WK select \* from COMP\_EBH\_SUMMARY\_BY\_REGION';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating COMP\_EBH\_SUMMARY\_BY\_REGION\_WK');

sql\_stmt:='delete from CSR\_SUMMARY\_BY\_REGION\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into CSR\_SUMMARY\_BY\_REGION\_WK select \* from CSR\_SUMMARY\_BY\_REGION';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_SUMMARY\_BY\_REGION\_WK');

sql\_stmt:='delete from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into HPOV\_VS\_XNG\_CSR\_AUDIT\_WK select \* from HPOV\_VS\_XNG\_CSR\_AUDIT';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating HPOV\_VS\_XNG\_CSR\_AUDIT');

sql\_stmt:='delete from CSR\_MISSING\_IN\_HPOV\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into CSR\_MISSING\_IN\_HPOV\_WK select \* from CSR\_MISSING\_IN\_HPOV';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_MISSING\_IN\_HPOV\_WK');

sql\_stmt:='delete from CSR\_AUDIT\_ISSUE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into CSR\_AUDIT\_ISSUE\_WK select \* from CSR\_AUDIT\_ISSUE';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_AUDIT\_ISSUE\_WK');

sql\_stmt:='delete from GIGE\_VLAN\_AUDIT\_ISSUE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into GIGE\_VLAN\_AUDIT\_ISSUE\_WK select \* from GIGE\_VLAN\_AUDIT\_ISSUE';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating GIGE\_VLAN\_AUDIT\_ISSUE\_WK');

sql\_stmt:='delete from XNG\_CSR\_PARSED\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_CSR\_PARSED\_WK select \* from XNG\_CSR\_PARSED';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_CSR\_PARSED\_WK');

sql\_stmt:='delete from CSR\_DEVICES\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='alter TRIGGER xng\_reports.put\_csr\_device\_id disable';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into CSR\_DEVICES\_WK select \* from CSR\_DEVICES';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='alter TRIGGER xng\_reports.put\_csr\_device\_id enable';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating CSR\_DEVICES\_WK');

sql\_stmt:='delete from XNG\_CSR\_GIGE\_PATHS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_CSR\_GIGE\_PATHS\_WK select \* from XNG\_CSR\_GIGE\_PATHS';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_CSR\_GIGE\_PATHS\_WK');

sql\_stmt:='delete from XNG\_CSR\_BTS\_PATHS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_CSR\_BTS\_PATHS\_WK select \* from XNG\_CSR\_BTS\_PATHS';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_CSR\_BTS\_PATHS\_WK');

sql\_stmt:='delete from XNG\_GIGE\_DELIVERY\_TABLE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_GIGE\_DELIVERY\_TABLE\_WK select \* from XNG\_GIGE\_DELIVERY\_TABLE';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_GIGE\_DELIVERY\_TABLE\_WK');

sql\_stmt:='delete from XNG\_GIGE\_PARENT\_MW\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_GIGE\_PARENT\_MW\_WK select \* from XNG\_GIGE\_PARENT\_MW';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_GIGE\_PARENT\_MW\_WK');

sql\_stmt:='delete from XNG\_VLAN\_PARSED\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_VLAN\_PARSED\_WK select \* from XNG\_VLAN\_PARSED';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_PARSED\_WK');

sql\_stmt:='delete from XNG\_GIGE\_VLAN\_MAP\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_GIGE\_VLAN\_MAP\_WK select \* from XNG\_GIGE\_VLAN\_MAP';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_GIGE\_VLAN\_MAP\_WK');

sql\_stmt:='delete from XNG\_GIGE\_PARENTS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_GIGE\_PARENTS\_WK select \* from XNG\_GIGE\_PARENTS';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_GIGE\_PARENTS\_WK');

sql\_stmt:='delete from XNG\_VLAN\_PARENTS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_VLAN\_PARENTS\_WK select \* from XNG\_VLAN\_PARENTS';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_PARENTS\_WK');

sql\_stmt:='delete from XNG\_MSPP\_DEVICES\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_MSPP\_DEVICES\_WK select \* from XNG\_MSPP\_DEVICES';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_MSPP\_DEVICES\_WK');

sql\_stmt:='delete from XNG\_NGMLS\_DEVICES\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_NGMLS\_DEVICES\_WK select \* from XNG\_NGMLS\_DEVICES';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_NGMLS\_DEVICES\_WK');

sql\_stmt:='delete from XNG\_VLAN\_NGMLS\_MAP\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_VLAN\_NGMLS\_MAP\_WK select \* from XNG\_VLAN\_NGMLS\_MAP';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_NGMLS\_MAP\_WK');

sql\_stmt:='delete from XNG\_VLAN\_MSPP\_MAP\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_VLAN\_MSPP\_MAP\_WK select \* from XNG\_VLAN\_MSPP\_MAP';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_MSPP\_MAP\_WK');

sql\_stmt:='delete from XNG\_VLAN\_MSPP\_TAM\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into XNG\_VLAN\_MSPP\_TAM\_WK select \* from XNG\_VLAN\_MSPP\_TAM';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating XNG\_VLAN\_MSPP\_MAP\_WK');

sql\_stmt:='delete from ngt\_qt\_ne\_vs\_xng\_audit\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into ngt\_qt\_ne\_vs\_xng\_audit\_wk select \* from ngt\_qt\_ne\_vs\_xng\_audit';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating ngt\_qt\_ne\_vs\_xng\_audit\_wk');

sql\_stmt:='delete from ngt\_qt\_reg\_summ\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into ngt\_qt\_reg\_summ\_wk select \* from ngt\_qt\_reg\_summ';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating ngt\_qt\_reg\_summ\_wk');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in move\_csr\_data\_from\_prod\_to\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

WATCHDOG.logerror('CSR\_AUDIT',4000,SubStr('Error in move\_csr\_data\_from\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('CSR\_AUDIT','STATUS\_FAILURE','N');

RAISE;

end move\_csr\_data\_from\_prod\_to\_wk;

procedure truncate\_csr\_wk is

sql\_stmt varchar2(32000);

begin

-- changed all truncates to deletes

sql\_stmt:='delete from COMP\_EBH\_SUMMARY\_BY\_REGION\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from COMP\_EBH\_SUMMARY\_BY\_REGION\_WK');

sql\_stmt:='delete from CSR\_SUMMARY\_BY\_REGION\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from CSR\_SUMMARY\_BY\_REGION\_WK');

sql\_stmt:='delete from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from HPOV\_VS\_XNG\_CSR\_AUDIT\_WK');

sql\_stmt:='delete from CSR\_MISSING\_IN\_HPOV\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from CSR\_MISSING\_IN\_HPOV\_WK');

sql\_stmt:='delete from CSR\_AUDIT\_ISSUE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from CSR\_AUDIT\_ISSUE');

sql\_stmt:='delete from GIGE\_VLAN\_AUDIT\_ISSUE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from GIGE\_VLAN\_AUDIT\_ISSUE\_WK');

sql\_stmt:='delete from XNG\_CSR\_PARSED\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from XNG\_CSR\_PARSED\_WK');

sql\_stmt:='delete from CSR\_DEVICES\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from CSR\_DEVICES\_WK');

sql\_stmt:='delete from XNG\_CSR\_GIGE\_PATHS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from XNG\_CSR\_GIGE\_PATHS\_WK');

sql\_stmt:='delete from XNG\_CSR\_BTS\_PATHS\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from XNG\_CSR\_BTS\_PATHS\_WK');

sql\_stmt:='delete from XNG\_GIGE\_DELIVERY\_TABLE\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from XNG\_GIGE\_DELIVERY\_TABLE\_WK');

sql\_stmt:='delete from XNG\_GIGE\_PARENT\_MW\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from XNG\_GIGE\_PARENT\_MW\_WK');

sql\_stmt:='delete from XNG\_VLAN\_PARSED\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from XNG\_VLAN\_PARSED\_WK');

sql\_stmt:='truncate table XNG\_GIGE\_VLAN\_MAP\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from XNG\_GIGE\_VLAN\_MAP\_WK');

sql\_stmt:='delete from xng\_gige\_parents\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from xng\_gige\_parents\_wk');

sql\_stmt:='delete from xng\_vlan\_parents\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from xng\_vlan\_parents\_wk');

sql\_stmt:='delete from xng\_mspp\_devices\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from xng\_mspp\_devices\_wk');

sql\_stmt:='delete from xng\_vlan\_ngMLS\_map\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from xng\_vlan\_ngMLS\_map\_wk');

sql\_stmt:='delete from xng\_ngMLS\_devices\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from xng\_ngMLS\_devices\_wk');

sql\_stmt:='delete from xng\_vlan\_mspp\_map\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from xng\_vlan\_mspp\_map\_wk');

sql\_stmt:='delete from gige\_vlan\_audit\_issue\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from gige\_vlan\_audit\_issue\_wk');

sql\_stmt:='delete from xng\_vlan\_mspp\_tam\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from xng\_vlan\_mspp\_tam\_wk');

sql\_stmt:='delete from ngt\_qt\_ne\_vs\_xng\_audit\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from ngt\_qt\_ne\_vs\_xng\_audit\_wk');

sql\_stmt:='delete from ngt\_qt\_reg\_summ\_wk';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Deleted from ngt\_qt\_reg\_summ\_wk');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in truncate\_csr\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

WATCHDOG.logerror('CSR\_AUDIT',4000,SubStr('Error in move\_csr\_data\_from\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('CSR\_AUDIT','STATUS\_FAILURE','N');

--RAISE;

end truncate\_csr\_wk;

/\*

Insert into COPY\_WK\_TABLES

(PROCESS\_ID, SEQ\_NO, SCHEMA\_NAME, WK\_TABLE\_NAME, PRD\_TABLE\_NAME,

LAST\_COPY\_DATE, WHERE\_CLAUSE)

Values

(4, 13, ' XNG\_REPORTS', 'xng\_gige\_parents\_wk', 'xng\_gige\_parents',

TO\_DATE('11/18/2011 13:43:40', 'MM/DD/YYYY HH24:MI:SS'), NULL);

Insert into COPY\_WK\_TABLES

(PROCESS\_ID, SEQ\_NO, SCHEMA\_NAME, WK\_TABLE\_NAME, PRD\_TABLE\_NAME,

LAST\_COPY\_DATE, WHERE\_CLAUSE)

Values

(4, 14, ' XNG\_REPORTS', 'xng\_vlan\_parents\_wk', 'xng\_vlan\_parents',

TO\_DATE('11/18/2011 13:43:40', 'MM/DD/YYYY HH24:MI:SS'), NULL);

Insert into COPY\_WK\_TABLES

(PROCESS\_ID, SEQ\_NO, SCHEMA\_NAME, WK\_TABLE\_NAME, PRD\_TABLE\_NAME,

LAST\_COPY\_DATE, WHERE\_CLAUSE)

Values

(4, 15, ' XNG\_REPORTS', 'xng\_mspp\_devices\_wk', 'xng\_mspp\_devices',

TO\_DATE('11/18/2011 13:43:40', 'MM/DD/YYYY HH24:MI:SS'), NULL);

Insert into COPY\_WK\_TABLES

(PROCESS\_ID, SEQ\_NO, SCHEMA\_NAME, WK\_TABLE\_NAME, PRD\_TABLE\_NAME,

LAST\_COPY\_DATE, WHERE\_CLAUSE)

Values

(4, 16, ' XNG\_REPORTS', 'xng\_ngmls\_devices\_wk', 'xng\_ngmls\_devices',

TO\_DATE('11/18/2011 13:43:40', 'MM/DD/YYYY HH24:MI:SS'), NULL);

Insert into COPY\_WK\_TABLES

(PROCESS\_ID, SEQ\_NO, SCHEMA\_NAME, WK\_TABLE\_NAME, PRD\_TABLE\_NAME,

LAST\_COPY\_DATE, WHERE\_CLAUSE)

Values

(4, 17, ' XNG\_REPORTS', 'xng\_vlan\_mspp\_map\_wk', 'xng\_vlan\_mspp\_map',

TO\_DATE('11/18/2011 13:43:40', 'MM/DD/YYYY HH24:MI:SS'), NULL);

\*/

END;

/

--------------------------------------------------------

-- DDL for Package Body DF\_STATUS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."DF\_STATUS"

IS

--

-- To modify this template, edit file PKGBODY.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package body

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter procedure, function bodies as shown below

PROCEDURE load\_df\_status\_details

IS

sql\_stmt varchar2(32000);

BEGIN

EXECUTE IMMEDIATE 'truncate table df\_status\_details\_wk reuse storage';

EXECUTE IMMEDIATE 'truncate table df\_status\_details\_new\_wk reuse storage';

sql\_stmt:= ' insert into DF\_STATUS\_DETAILS\_WK columns (AREA,REGION,LEAF\_DOMAIN\_NAME,SITE\_INST\_ID,SITE\_TYPE,LOC\_CAT,

PS\_ID,NWF\_ID,SITE\_STATUS,SITE\_NAME,CKT\_ID,CKT\_NAME,CKT\_VENDOR,CKT\_STATUS,BILL\_DATE,PATH\_INST\_ID, PATH\_NAME )

with sites as (

select distinct si.site\_inst\_id, si.site\_hum\_id site\_name, si.status site\_status,si.num site\_type,

ci.circ\_inst\_id ckt\_id,ci.circ\_hum\_id ckt\_name,ci.vendor ckt\_vendor,ci.status ckt\_status,

coalesce(ci.circ\_path\_inst\_id,ci.next\_path\_inst\_id) as path\_inst\_id,

lc.location\_category,

nwf.nwf\_id,

ps.ps\_id

from

vzwnet.circ\_inst ci,

vzwnet.site\_inst si

left join (select sas.site\_inst\_id, sas.attr\_value nwf\_id from vzwnet.site\_attr\_settings sas join vzwnet.val\_attr\_name van on sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id and van.group\_name = ''Site Information''

and van.attr\_name = ''NWF Site ID (NAURU)'' ) nwf

on (si.site\_inst\_id = nwf.site\_inst\_id)

left join (select sas.site\_inst\_id, sas.attr\_value location\_category from vzwnet.site\_attr\_settings sas join vzwnet.val\_attr\_name van on sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id and van.group\_name = ''Site Realignment''

and van.attr\_name = ''Location Category'' ) lc on (si.site\_inst\_id = lc.site\_inst\_id)

left join (select sas.site\_inst\_id, sas.attr\_value ps\_id from vzwnet.site\_attr\_settings sas join vzwnet.val\_attr\_name van on sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id and van.group\_name = ''Site Information''

and van.attr\_name = ''PeopleSoft Location Code #'' ) ps on (si.site\_inst\_id = ps.site\_inst\_id)

where ci.z\_site\_id = si.site\_inst\_id

and ci.type = ''DARK FIBER''

and ci.status = ''Live''

),

seginfo as (select circ\_inst\_id, bill\_date

FROM (SELECT ca.circ\_inst\_id,

CASE

WHEN LENGTH(ca.attr\_value) = 9

THEN TO\_DATE(ca.attr\_value,''dd-MON-yy'')

ELSE TO\_DATE(''0''||ca.attr\_value,''dd-MON-yy'')

END bill\_date

FROM vzwnet.circ\_attr\_settings ca,

vzwnet.val\_attr\_name v

WHERE ca.val\_attr\_inst\_id = v.val\_attr\_inst\_id

AND v.group\_name = ''Segment Information''

AND v.attr\_name = ''Bill Date'')

WHERE bill\_date >= SYSDATE - 60)

select distinct

area, region,d.domain\_name leaf\_domain\_name,

s.site\_inst\_id,

s.site\_type,

s.location\_category,

s.ps\_id,

s.nwf\_id,

s.site\_status,

s.site\_name,

ckt\_id,

ckt\_name,

ckt\_vendor,

ckt\_status,

si.bill\_date,

s.path\_inst\_id,

cpi.circ\_path\_hum\_id path\_name

from

xng\_reports.domains\_leaf\_reporting d,

vzwnet.site\_domain\_map sdi,

sites s

left join vzwnet.circ\_path\_inst cpi

on s.path\_inst\_id = cpi.circ\_path\_inst\_id

left join seginfo si on s.ckt\_id = si.circ\_inst\_id

where D.DOMAIN\_INST\_ID = SDI.DOMAIN\_INST\_ID

and s.site\_inst\_id = SDI.SITE\_INST\_ID and region != ''OSS''

order by area, region, site\_name'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:= ' insert into DF\_STATUS\_DETAILS\_new\_WK columns (TERRITORY, MARKET, SUB\_MARKET,LEAF\_DOMAIN\_NAME,SITE\_INST\_ID,SITE\_TYPE,LOC\_CAT,

PS\_ID,NWF\_ID,SITE\_STATUS,SITE\_NAME,CKT\_ID,CKT\_NAME,CKT\_VENDOR,CKT\_STATUS,BILL\_DATE,PATH\_INST\_ID, PATH\_NAME )

with sites as (

select distinct si.site\_inst\_id, si.site\_hum\_id site\_name, si.status site\_status,si.num site\_type,

ci.circ\_inst\_id ckt\_id,ci.circ\_hum\_id ckt\_name,ci.vendor ckt\_vendor,ci.status ckt\_status,

coalesce(ci.circ\_path\_inst\_id,ci.next\_path\_inst\_id) as path\_inst\_id,

lc.location\_category,

nwf.nwf\_id,

ps.ps\_id

from

vzwnet.circ\_inst ci,

vzwnet.site\_inst si

left join (select sas.site\_inst\_id, sas.attr\_value nwf\_id from vzwnet.site\_attr\_settings sas join vzwnet.val\_attr\_name van on sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id and van.group\_name = ''Site Information''

and van.attr\_name = ''NWF Site ID (NAURU)'' ) nwf

on (si.site\_inst\_id = nwf.site\_inst\_id)

left join (select sas.site\_inst\_id, sas.attr\_value location\_category from vzwnet.site\_attr\_settings sas join vzwnet.val\_attr\_name van on sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id and van.group\_name = ''Site Realignment''

and van.attr\_name = ''Location Category'' ) lc on (si.site\_inst\_id = lc.site\_inst\_id)

left join (select sas.site\_inst\_id, sas.attr\_value ps\_id from vzwnet.site\_attr\_settings sas join vzwnet.val\_attr\_name van on sas.val\_attr\_inst\_id = van.val\_attr\_inst\_id and van.group\_name = ''Site Information''

and van.attr\_name = ''PeopleSoft Location Code #'' ) ps on (si.site\_inst\_id = ps.site\_inst\_id)

where ci.z\_site\_id = si.site\_inst\_id

and ci.type = ''DARK FIBER''

and ci.status = ''Live''

),

seginfo as (select circ\_inst\_id, bill\_date

FROM (SELECT ca.circ\_inst\_id,

TO\_DATE(ca.attr\_value,''dd-MON-yyyy'') bill\_date

FROM vzwnet.circ\_attr\_settings ca,

vzwnet.val\_attr\_name v

WHERE ca.val\_attr\_inst\_id = v.val\_attr\_inst\_id

AND v.group\_name = ''Segment Information''

AND v.attr\_name = ''Bill Date'')

WHERE bill\_date >= SYSDATE - 60)

select distinct

--area, region,d.domain\_name leaf\_domain\_name,

territory, market\_territory MARKET, sub\_market,d.domain\_name leaf\_domain\_name,

s.site\_inst\_id,

s.site\_type,

s.location\_category,

s.ps\_id,

s.nwf\_id,

s.site\_status,

s.site\_name,

ckt\_id,

ckt\_name,

ckt\_vendor,

ckt\_status,

si.bill\_date,

s.path\_inst\_id,

cpi.circ\_path\_hum\_id path\_name

from

xng\_reports.domains\_leaf\_reporting d,

vzwnet.site\_domain\_map sdi,

sites s

left join vzwnet.circ\_path\_inst cpi

on s.path\_inst\_id = cpi.circ\_path\_inst\_id

left join seginfo si on s.ckt\_id = si.circ\_inst\_id

where D.DOMAIN\_INST\_ID = SDI.DOMAIN\_INST\_ID

and s.site\_inst\_id = SDI.SITE\_INST\_ID and region != ''OSS''

order by territory, market, sub\_market,site\_name'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in load\_df\_status\_details(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend (PROCESS\_NAME, 0, 'N');

RAISE;

END;

PROCEDURE load\_df\_status\_summary

IS

sql\_stmt varchar2(32000);

BEGIN

EXECUTE IMMEDIATE 'truncate table df\_status\_summary\_wk reuse storage';

EXECUTE IMMEDIATE 'truncate table df\_status\_summary\_NEW\_wk reuse storage';

sql\_stmt:= ' insert into DF\_STATUS\_SUMMARY\_WK columns (AREA,REGION,INBUILDING,MACRO,MICRO,NODE,VENUE,BLANK,TOTAL )

with sitetypes as (

select

distinct si.site\_inst\_id, SI.SITE\_HUM\_ID site\_name,

SI.NUM site\_type,

lc.attr\_value location\_category

from

vzwnet.circ\_inst ci,

vzwnet.site\_inst si,

(select SAS.ATTR\_VALUE, SAS.SITE\_INST\_ID

from vzwnet.site\_attr\_settings sas, vzwnet.val\_attr\_name van

where VAN.VAL\_ATTR\_INST\_ID = SAS.VAL\_ATTR\_INST\_ID

and VAN.ATTR\_NAME = ''Location Category'') lc

where CI.Z\_SITE\_ID = SI.SITE\_INST\_ID

and SI.SITE\_INST\_ID = lc.site\_inst\_id (+)

and CI.TYPE = ''DARK FIBER''

and CI.STATUS = ''Live''

)

select replace(area, ''\_Grand'', ''Grand'') area, region, INBUILDING, MACRO, MICRO, NODE, VENUE, BLANK, TOTAL from (

select nvl(area, ''\_Grand Total'') area, region,

sum (case when location\_category=''IN-BUILDING'' then 1 else 0 end) INBUILDING

,sum (case when location\_category=''MACRO'' then 1 else 0 end) MACRO

,sum (case when location\_category=''MICRO'' then 1 else 0 end) MICRO

,sum (case when location\_category=''NODE'' then 1 else 0 end) NODE

,sum (case when location\_category=''VENUE'' then 1 else 0 end) VENUE

,sum (case when ( location\_category is null) then 1 else 0 end) BLANK

,count (site\_type ) Total

from sitetypes, xng\_reports.domains\_leaf\_reporting d,

vzwnet.site\_domain\_map sdi

where D.DOMAIN\_INST\_ID = SDI.DOMAIN\_INST\_ID

and sitetypes.site\_inst\_id = SDI.SITE\_INST\_ID

and region != ''OSS''

group by rollup ( AREA, region) order by area, region

)

where region != ''NNO'' or region is null';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:= ' insert into DF\_STATUS\_SUMMARY\_NEW\_WK columns (TERRITORY, MARKET,SUB\_MARKET,INBUILDING,MACRO,MICRO,NODE,VENUE,BLANK,TOTAL )

with sitetypes as (

select

distinct si.site\_inst\_id, SI.SITE\_HUM\_ID site\_name,

SI.NUM site\_type,

lc.attr\_value location\_category

from

vzwnet.circ\_inst ci,

vzwnet.site\_inst si,

(select SAS.ATTR\_VALUE, SAS.SITE\_INST\_ID

from vzwnet.site\_attr\_settings sas, vzwnet.val\_attr\_name van

where VAN.VAL\_ATTR\_INST\_ID = SAS.VAL\_ATTR\_INST\_ID

and VAN.ATTR\_NAME = ''Location Category'') lc

where CI.Z\_SITE\_ID = SI.SITE\_INST\_ID

and SI.SITE\_INST\_ID = lc.site\_inst\_id (+)

and CI.TYPE = ''DARK FIBER''

and CI.STATUS = ''Live''

)

select replace(TERRITORY, ''\_Grand'', ''Grand'') TERRITORY, MARKET\_TERRITORY,SUB\_MARKET,INBUILDING, MACRO, MICRO, NODE, VENUE, BLANK, TOTAL from (

select nvl(D.TERRITORY, ''\_Grand Total'') TERRITORY, D.MARKET\_TERRITORY ,SUB\_MARKET,

sum (case when location\_category=''IN-BUILDING'' then 1 else 0 end) INBUILDING

,sum (case when location\_category=''MACRO'' then 1 else 0 end) MACRO

,sum (case when location\_category=''MICRO'' then 1 else 0 end) MICRO

,sum (case when location\_category=''NODE'' then 1 else 0 end) NODE

,sum (case when location\_category=''VENUE'' then 1 else 0 end) VENUE

,sum (case when ( location\_category is null) then 1 else 0 end) BLANK

,count (site\_type ) Total

from sitetypes, xng\_reports.domains\_leaf\_reporting d,

vzwnet.site\_domain\_map sdi

where D.DOMAIN\_INST\_ID = SDI.DOMAIN\_INST\_ID

and sitetypes.site\_inst\_id = SDI.SITE\_INST\_ID

and region != ''OSS''

group by rollup ( TERRITORY,MARKET\_TERRITORY,SUB\_MARKET) order by TERRITORY,MARKET\_TERRITORY,SUB\_MARKET

)

where MARKET\_TERRITORY != ''NNO'' or MARKET\_TERRITORY is null';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:= 'update df\_status\_summary\_new\_wk wk set wk.TERRITORY\_MARKET\_SUB = SUB\_MARKET where sub\_market is not null';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:= 'update df\_status\_summary\_new\_wk wk set wk.TERRITORY\_MARKET\_SUB = MARKET where sub\_market is null and wk.market is not null';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:= 'update df\_status\_summary\_new\_wk wk set wk.TERRITORY\_MARKET\_SUB = TERRITORY where sub\_market is null and wk.market is null' ;

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in load\_df\_status\_summary(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend (PROCESS\_NAME, 0, 'N');

RAISE;

END;

PROCEDURE do\_all

IS

sql\_stmt varchar2(32000);

BEGIN

watchdog.updateProcessStart(PROCESS\_NAME);

load\_df\_status\_details();

load\_df\_status\_summary();

watchdog.updateProcessEnd(PROCESS\_NAME,'STATUS\_SUCCESS','Y');

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in do\_all(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend (PROCESS\_NAME, 0, 'N');

END;

END;

/

--------------------------------------------------------

-- DDL for Package Body EBH\_DISCONNECT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."EBH\_DISCONNECT"

IS

--

-- To modify this template, edit file PKGBODY.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package body

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter procedure, function bodies as shown below

PROCEDURE load\_df\_seg\_wo\_billing

IS

sql\_stmt varchar2(32000);

BEGIN

-- df has no bill, ebh disconnect dates (duration)

sql\_stmt:= 'insert into ebh\_disconnect\_details\_wk columns(area, region, domain\_name, df\_seg\_id, df\_has\_bill, ndf\_seg\_id, ebh\_disconnect\_date, duration, df\_site\_id, audit\_date, ebh\_bill\_date)

with

LEAF\_DOMAINS AS (

SELECT domain\_inst\_id

FROM vzwnet.domain\_inst di

where domain\_inst\_id <> 1012 --exclude NNO domain

MINUS -- minus the parents

SELECT parent\_domain\_inst\_id

FROM vzwnet.domain\_inst

)

,BILL\_DATE\_UDA AS (

SELECT van.VAL\_ATTR\_INST\_ID

FROM vzwnet.val\_attr\_name van

where van.attr\_name = ''Bill Date''

and van.group\_name = ''Segment Information''

)

,SEG\_W\_BILL\_DATE as ( select -- cas.attr\_value as bill\_date,

CAS.CIRC\_INST\_ID

from vzwnet.circ\_attr\_settings cas

join BILL\_DATE\_UDA van on van.val\_attr\_inst\_id = cas.val\_attr\_inst\_id

where (sysdate - to\_date(attr\_value,''dd-mon-yy'')) <= 90

)

,all\_df\_segments as (

select distinct x.CIRC\_INST\_ID df\_seg\_id, x.CIRC\_HUM\_ID df\_seg\_name,x.type df\_seg\_type, x.BANDWIDTH df\_seg\_bw,x.STATUS df\_seg\_status

,x.vendor df\_seg\_vendor, nvl(sbd.CIRC\_INST\_ID,0) as df\_w\_bill

,x.Z\_SITE\_ID df\_site\_id, SI.SITE\_HUM\_ID df\_site\_name, x.recur\_costs df\_recur\_costs, x.non\_recur\_costs df\_nonrecur\_costs

from

vzwnet.site\_inst si

,VZWNET.CIRC\_INST x

left outer join vzwnet.circ\_path\_inst cpi on (x.circ\_path\_inst\_id=CPI.CIRC\_PATH\_INST\_ID or X.NEXT\_PATH\_INST\_ID =cpi.circ\_path\_inst\_id)

left outer join SEG\_W\_BILL\_DATE sbd on x.CIRC\_INST\_ID=sbd.CIRC\_INST\_ID

where ( x.type=''DARK FIBER'')

-- and x.STATUS not in (''Decommissioned'',''Cancelled'')

and ((x.Z\_SITE\_ID = SI.SITE\_INST\_ID and x.Z\_SITE\_ID is not null)

or (x.Z\_SITE\_ID is not null and CPI.Z\_SIDE\_SITE\_ID = SI.SITE\_INST\_ID)

)

and si.num <> ''MTSO''

)--select \* from all\_df\_segments where df\_in\_btp <> 0;

,all\_ebh\_segments as (

select distinct x.CIRC\_INST\_ID ndf\_seg\_id, x.CIRC\_HUM\_ID ndf\_seg\_name,x.type ndf\_seg\_type, x.BANDWIDTH ndf\_seg\_bw,x.STATUS ndf\_seg\_status

,x.vendor ndf\_seg\_vendor

,x.Z\_SITE\_ID ndf\_site\_id, SI.SITE\_HUM\_ID ndf\_site\_name, x.recur\_costs ndf\_recur\_costs, x.non\_recur\_costs ndf\_nonrecur\_costs

from

vzwnet.site\_inst si

,VZWNET.CIRC\_INST x

left outer join vzwnet.circ\_path\_inst cpi on (x.circ\_path\_inst\_id=CPI.CIRC\_PATH\_INST\_ID or X.NEXT\_PATH\_INST\_ID =cpi.circ\_path\_inst\_id)

where ( x.type=''LEASED SERVICE'')

and ((upper(x.BANDWIDTH) like ''% MBPS'') or (upper(x.BANDWIDTH) like ''% GBPS''))

-- and x.STATUS not in (''Decommissioned'',''Cancelled'')

and ((x.Z\_SITE\_ID = SI.SITE\_INST\_ID and x.Z\_SITE\_ID is not null)

or (x.Z\_SITE\_ID is not null and CPI.Z\_SIDE\_SITE\_ID = SI.SITE\_INST\_ID))

)

,DISCONN\_DATE\_UDA AS (

SELECT van.VAL\_ATTR\_INST\_ID

FROM vzwnet.val\_attr\_name van

where van.attr\_name = ''Disconnect Date''

and van.group\_name = ''Disconnect Information''

)

,EBH\_SEG\_DISCON\_DATES as

(select dd.circ\_inst\_id, dd.uda\_disconnect\_date ebh\_disconnect\_date,

case

when (sysdate - dd.uda\_disconnect\_date) <= 30 then ''1''

when (sysdate - dd.uda\_disconnect\_date) <= 60 then ''2''

when (sysdate - dd.uda\_disconnect\_date) <= 90 then ''3''

when (sysdate - dd.uda\_disconnect\_date) <= 120 then ''4''

when (sysdate - dd.uda\_disconnect\_date) <= 180 then ''5''

when (sysdate - dd.uda\_disconnect\_date) > 180 then ''6''

end ebh\_disconnect\_duration

from all\_ebh\_segments ebh,

(select circ\_inst\_id,

case

when length(uda\_disconnect\_date) = 9 then to\_date(uda\_disconnect\_date,''dd-mon-yy'')

when length(uda\_disconnect\_date) = 11 then to\_date(uda\_disconnect\_date,''dd-mon-yyyy'')

else null

end uda\_disconnect\_date

from (select cas.circ\_inst\_id, substr(cas.attr\_value,1,11) as uda\_disconnect\_date

from vzwnet.circ\_attr\_settings cas

join DISCONN\_DATE\_UDA van on van.val\_attr\_inst\_id = cas.val\_attr\_inst\_id)) dd

where dd.CIRC\_INST\_ID = ebh.NDF\_SEG\_ID

)-- select \* from EBH\_SEG\_DISCON\_DATES;

,EBH\_BILL\_DATE as

(select dd.circ\_inst\_id, dd.ebh\_bill\_date

from all\_ebh\_segments ebh,

(select circ\_inst\_id,

case

when length(uda\_bill\_date) = 9 then to\_date(uda\_bill\_date,''dd-mon-yy'')

when length(uda\_bill\_date) = 11 then to\_date(uda\_bill\_date,''dd-mon-yyyy'')

else null

end ebh\_bill\_date

from (select cas.circ\_inst\_id, substr(cas.attr\_value,1,11) as uda\_bill\_date

from vzwnet.circ\_attr\_settings cas

join BILL\_DATE\_UDA van on van.val\_attr\_inst\_id = cas.val\_attr\_inst\_id)) dd

where dd.CIRC\_INST\_ID = ebh.NDF\_SEG\_ID

)

select distinct area, region, domain\_name,

df.df\_seg\_id, ''N'' df\_has\_bill, ebh.ndf\_seg\_id, edd.ebh\_disconnect\_date, nvl(edd.ebh\_disconnect\_duration,0) duration, df.df\_site\_id, sysdate

,ebd.ebh\_bill\_date from all\_df\_segments df,all\_ebh\_segments ebh , LEAF\_DOMAINS li, vzwnet.circ\_domain\_map cdm, xng\_reports.domains\_leaf\_reporting dlr

, EBH\_SEG\_DISCON\_DATES edd ,EBH\_BILL\_DATE ebd

where df.df\_site\_id = ebh.ndf\_site\_id

and cdm.domain\_inst\_id = li.domain\_inst\_id

and df.df\_seg\_id = CDM.CIRC\_INST\_ID

and DLR.DOMAIN\_INST\_ID = li.domain\_inst\_id

and ebh.ndf\_seg\_id = edd.circ\_inst\_id (+)

and ebh.ndf\_seg\_id = ebd.circ\_inst\_id (+)

and df\_w\_bill = 0

order by area, region, domain\_name';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in load\_df\_seg\_w\_billing(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

-- watchdog.updateprocessend (PROCESS\_NAME, 0, 'N');

RAISE;

END;

PROCEDURE load\_df\_seg\_w\_billing

IS

sql\_stmt varchar2(32000);

BEGIN

-- df has billing, df in service dates(duration)

sql\_stmt:= 'insert into ebh\_disconnect\_details\_wk columns(area, region, domain\_name, df\_seg\_id, df\_has\_bill, ndf\_seg\_id, ebh\_disconnect\_date, df\_in\_service\_date ,duration, df\_site\_id, audit\_date, ebh\_bill\_date)

with

LEAF\_DOMAINS AS (

SELECT domain\_inst\_id

FROM vzwnet.domain\_inst di

where domain\_inst\_id <> 1012 --exclude NNO domain

MINUS -- minus the parents

SELECT parent\_domain\_inst\_id

FROM vzwnet.domain\_inst

)

,BILL\_DATE\_UDA AS (

SELECT van.VAL\_ATTR\_INST\_ID

FROM vzwnet.val\_attr\_name van

where van.attr\_name = ''Bill Date''

and van.group\_name = ''Segment Information''

)

,SEG\_W\_BILL\_DATE as ( select -- cas.attr\_value as bill\_date,

CAS.CIRC\_INST\_ID

from vzwnet.circ\_attr\_settings cas

join BILL\_DATE\_UDA van on van.val\_attr\_inst\_id = cas.val\_attr\_inst\_id

where (sysdate - to\_date(attr\_value,''dd-mon-yy'')) <= 90

)

,all\_df\_segments as (

select distinct x.CIRC\_INST\_ID df\_seg\_id, x.CIRC\_HUM\_ID df\_seg\_name,x.type df\_seg\_type, x.BANDWIDTH df\_seg\_bw,x.STATUS df\_seg\_status

,x.vendor df\_seg\_vendor, nvl(sbd.CIRC\_INST\_ID,0) as df\_w\_bill

,x.Z\_SITE\_ID df\_site\_id, SI.SITE\_HUM\_ID df\_site\_name, x.recur\_costs df\_recur\_costs, x.non\_recur\_costs df\_nonrecur\_costs

from

vzwnet.site\_inst si

,VZWNET.CIRC\_INST x

left outer join vzwnet.circ\_path\_inst cpi on (x.circ\_path\_inst\_id=CPI.CIRC\_PATH\_INST\_ID or X.NEXT\_PATH\_INST\_ID =cpi.circ\_path\_inst\_id)

left outer join SEG\_W\_BILL\_DATE sbd on x.CIRC\_INST\_ID=sbd.CIRC\_INST\_ID

where ( x.type=''DARK FIBER'')

-- and x.STATUS not in (''Decommissioned'',''Cancelled'')

and ((x.Z\_SITE\_ID = SI.SITE\_INST\_ID and x.Z\_SITE\_ID is not null)

or (x.Z\_SITE\_ID is not null and CPI.Z\_SIDE\_SITE\_ID = SI.SITE\_INST\_ID)

)

and si.num <> ''MTSO''

)--select \* from all\_df\_segments where df\_in\_btp <> 0;

,all\_ebh\_segments as (

select distinct x.CIRC\_INST\_ID ndf\_seg\_id, x.CIRC\_HUM\_ID ndf\_seg\_name,x.type ndf\_seg\_type, x.BANDWIDTH ndf\_seg\_bw,x.STATUS ndf\_seg\_status

,x.vendor ndf\_seg\_vendor

,x.Z\_SITE\_ID ndf\_site\_id, SI.SITE\_HUM\_ID ndf\_site\_name, x.recur\_costs ndf\_recur\_costs, x.non\_recur\_costs ndf\_nonrecur\_costs

from

vzwnet.site\_inst si

,VZWNET.CIRC\_INST x

left outer join vzwnet.circ\_path\_inst cpi on (x.circ\_path\_inst\_id=CPI.CIRC\_PATH\_INST\_ID or X.NEXT\_PATH\_INST\_ID =cpi.circ\_path\_inst\_id)

where ( x.type=''LEASED SERVICE'')

and ((upper(x.BANDWIDTH) like ''% MBPS'') or (upper(x.BANDWIDTH) like ''% GBPS''))

-- and x.STATUS not in (''Decommissioned'',''Cancelled'')

and ((x.Z\_SITE\_ID = SI.SITE\_INST\_ID and x.Z\_SITE\_ID is not null)

or (x.Z\_SITE\_ID is not null and CPI.Z\_SIDE\_SITE\_ID = SI.SITE\_INST\_ID))

)

,DISCONN\_DATE\_UDA AS (

SELECT van.VAL\_ATTR\_INST\_ID

FROM vzwnet.val\_attr\_name van

where van.attr\_name = ''Disconnect Date''

and van.group\_name = ''Disconnect Information''

)

,EBH\_DISCONNECT\_DATES as

(select dd.circ\_inst\_id, dd.ebh\_disconnect\_date

from all\_ebh\_segments ebh,

(select circ\_inst\_id,

case

when length(uda\_disconnect\_date) = 9 then to\_date(uda\_disconnect\_date,''dd-mon-yy'')

when length(uda\_disconnect\_date) = 11 then to\_date(uda\_disconnect\_date,''dd-mon-yyyy'')

else null

end ebh\_disconnect\_date

from (select cas.circ\_inst\_id, substr(cas.attr\_value,1,11) as uda\_disconnect\_date

from vzwnet.circ\_attr\_settings cas

join DISCONN\_DATE\_UDA van on van.val\_attr\_inst\_id = cas.val\_attr\_inst\_id)) dd

where dd.CIRC\_INST\_ID = ebh.NDF\_SEG\_ID

),EBH\_BILL\_DATE as

(select dd.circ\_inst\_id, dd.ebh\_bill\_date

from all\_ebh\_segments ebh,

(select circ\_inst\_id,

case

when length(uda\_bill\_date) = 9 then to\_date(uda\_bill\_date,''dd-mon-yy'')

when length(uda\_bill\_date) = 11 then to\_date(uda\_bill\_date,''dd-mon-yyyy'')

else null

end ebh\_bill\_date

from (select cas.circ\_inst\_id, substr(cas.attr\_value,1,11) as uda\_bill\_date

from vzwnet.circ\_attr\_settings cas

join BILL\_DATE\_UDA van on van.val\_attr\_inst\_id = cas.val\_attr\_inst\_id)) dd

where dd.CIRC\_INST\_ID = ebh.NDF\_SEG\_ID

)

,DF\_INSERVICE\_DATES as

(select df.df\_seg\_id, BXA.IN\_SERVICE df\_in\_service\_date ,

case

when (sysdate - BXA.IN\_SERVICE ) <= 30 then ''1''

when (sysdate - BXA.IN\_SERVICE ) <= 60 then ''2''

when (sysdate - BXA.IN\_SERVICE ) <= 90 then ''3''

when (sysdate - BXA.IN\_SERVICE ) <= 120 then ''4''

when (sysdate - BXA.IN\_SERVICE ) <= 180 then ''5''

when (sysdate - BXA.IN\_SERVICE ) > 180 then ''6''

end df\_inservice\_duration

from all\_df\_segments df, vzwnet.circ\_inst bxa --xng\_reports.btp\_xng\_audit bxa

where df.DF\_SEG\_ID = BXA.CIRC\_INST\_ID

)

select distinct area, region, domain\_name,

df.df\_seg\_id, ''Y'' df\_has\_bill,

ebh.ndf\_seg\_id, edd.ebh\_disconnect\_date, did.df\_in\_service\_date, nvl(did.df\_inservice\_duration,0) duration,

df.df\_site\_id, sysdate, ebd.ebh\_bill\_date

from all\_df\_segments df,all\_ebh\_segments ebh , LEAF\_DOMAINS li, vzwnet.circ\_domain\_map cdm, xng\_reports.domains\_leaf\_reporting dlr

, EBH\_DISCONNECT\_DATES edd , DF\_INSERVICE\_DATES did,EBH\_BILL\_DATE ebd

where df.df\_site\_id = ebh.ndf\_site\_id

and cdm.domain\_inst\_id = li.domain\_inst\_id

and df.df\_seg\_id = CDM.CIRC\_INST\_ID

and DLR.DOMAIN\_INST\_ID = li.domain\_inst\_id

and ebh.ndf\_seg\_id = edd.circ\_inst\_id (+)

and ebh.ndf\_seg\_id = ebd.circ\_inst\_id (+)

and df.df\_seg\_id = did.df\_seg\_id(+)

and df\_w\_bill > 0

order by area, region, domain\_name';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in load\_df\_seg\_wo\_billing(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

-- watchdog.updateprocessend (PROCESS\_NAME, 0, 'N');

RAISE;

END;

PROCEDURE do\_all

IS

sql\_stmt varchar2(32000);

BEGIN

watchdog.updateProcessStart(PROCESS\_NAME);

EXECUTE IMMEDIATE 'truncate table ebh\_disconnect\_details\_wk reuse storage';

load\_df\_seg\_w\_billing();

load\_df\_seg\_wo\_billing();

ebh\_disconnect\_summary();

watchdog.updateProcessEnd(PROCESS\_NAME,'STATUS\_SUCCESS','Y');

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in do\_all(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend (PROCESS\_NAME, 0, 'N');

END;

PROCEDURE ebh\_disconnect\_summary

IS

sql\_stmt varchar2(32000);

BEGIN

EXECUTE IMMEDIATE 'truncate table ebh\_disconnect\_summary\_wk reuse storage';

-- df has a bill , disconnect date is null

sql\_stmt := 'insert into ebh\_disconnect\_summary\_wk columns(AREA,REGION,NOINSERV,DUR0TO30,DUR30TO60,DUR60TO90,DUR90TO120,DUR120TO180,

DUROVER180,CTREGIONTOTAL,AMTNOINSERV, AMTDUR0TO30,AMTDUR30TO60 , AMTDUR60TO90 , AMTDUR90TO120 , AMTDUR120TO180,

AMTDUROVER180 , AMTREGIONTOTAL , FLAG )

select area, region, count (case when dur = 0 then dur else null end) noInServ,

count (case when dur = 1 then dur else null end) dur0to30,

count (case when dur = 2 then dur else null end) dur30to60,

count (case when dur = 3 then dur else null end) dur60to90,

count (case when dur = 4 then dur else null end) dur90to120,

count (case when dur = 5 then dur else null end) dur120to180,

count (case when dur = 6 then dur else null end) durOver180,

count (case when dur > 0 then dur else null end) ctRegionTotal,

sum (case when dur = 0 then recurCost else null end) amtNoInserv,

sum (case when dur = 1 then recurCost else null end) amtDur0to30,

sum (case when dur = 2 then recurCost else null end) amtDur30to60,

sum (case when dur = 3 then recurCost else null end) amtDur60to90,

sum (case when dur = 4 then recurCost else null end) amtDur90to120,

sum (case when dur = 5 then recurCost else null end) amtDur120to180,

sum (case when dur = 6 then recurCost else null end) amtDurOver180,

sum (case when dur > 0 then recurCost else 0 end) amtRegionTotal

,''df\_has\_bill\_ebh\_no\_disconnect'' flag

from ( select l.area, l.region, tdm.duration dur, ci.recur\_costs recurCost

from xng\_reports.domains\_leaf\_reporting l

left outer join xng\_reports.ebh\_disconnect\_details\_wk tdm on l.domain\_name = tdm.domain\_name

and tdm.df\_has\_bill = ''Y'' and tdm.ebh\_disconnect\_date is null

left outer join vzwnet.circ\_inst ci on tdm.ndf\_seg\_id = ci.circ\_inst\_id

where l.area <> ''OSS'' )

group by rollup (area, region) order by area, region ';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- df has a bill , disconnect date is not null

sql\_stmt := 'insert into ebh\_disconnect\_summary\_wk columns(AREA,REGION,NOINSERV,DUR0TO30,DUR30TO60,DUR60TO90,DUR90TO120,DUR120TO180,

DUROVER180,CTREGIONTOTAL,AMTNOINSERV, AMTDUR0TO30,AMTDUR30TO60 , AMTDUR60TO90 , AMTDUR90TO120 , AMTDUR120TO180,

AMTDUROVER180 , AMTREGIONTOTAL , FLAG )

select area, region, count (case when dur = 0 then dur else null end) noInServ,

count (case when dur = 1 then dur else null end) dur0to30,

count (case when dur = 2 then dur else null end) dur30to60,

count (case when dur = 3 then dur else null end) dur60to90,

count (case when dur = 4 then dur else null end) dur90to120,

count (case when dur = 5 then dur else null end) dur120to180,

count (case when dur = 6 then dur else null end) durOver180,

count (case when dur > 0 then dur else null end) ctRegionTotal,

sum (case when dur = 0 then recurCost else null end) amtNoInserv,

sum (case when dur = 1 then recurCost else null end) amtDur0to30,

sum (case when dur = 2 then recurCost else null end) amtDur30to60,

sum (case when dur = 3 then recurCost else null end) amtDur60to90,

sum (case when dur = 4 then recurCost else null end) amtDur90to120,

sum (case when dur = 5 then recurCost else null end) amtDur120to180,

sum (case when dur = 6 then recurCost else null end) amtDurOver180,

sum (case when dur > 0 then recurCost else 0 end) amtRegionTotal

,''df\_has\_bill\_ebh\_has\_disconnect'' flag

from ( select l.area, l.region, tdm.duration dur, ci.recur\_costs recurCost

from xng\_reports.domains\_leaf\_reporting l

left outer join xng\_reports.ebh\_disconnect\_details\_wk tdm on l.domain\_name = tdm.domain\_name

and tdm.df\_has\_bill = ''Y'' and tdm.ebh\_disconnect\_date is not null

left outer join vzwnet.circ\_inst ci on tdm.ndf\_seg\_id = ci.circ\_inst\_id

where l.area <> ''OSS'' )

group by rollup (area, region) order by area, region

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- df does not have a bill

sql\_stmt:= 'insert into ebh\_disconnect\_summary\_wk columns(AREA,REGION,NODISCONNECT,DUR0TO30,DUR30TO60,DUR60TO90,DUR90TO120,DUR120TO180,

DUROVER180,CTREGIONTOTAL,AMTNODISCONNECT, AMTDUR0TO30,AMTDUR30TO60 , AMTDUR60TO90 , AMTDUR90TO120 , AMTDUR120TO180,

AMTDUROVER180 , AMTREGIONTOTAL , FLAG )

select area, region, count (case when dur = 0 then dur else null end) noDiscon,

count (case when dur = 1 then dur else null end) dur0to30,

count (case when dur = 2 then dur else null end) dur30to60,

count (case when dur = 3 then dur else null end) dur60to90,

count (case when dur = 4 then dur else null end) dur90to120,

count (case when dur = 5 then dur else null end) dur120to180,

count (case when dur = 6 then dur else null end) durOver180,

count (case when dur > 0 then dur else null end) ctRegionTotal,

sum (case when dur = 0 then recurCost else null end) amtNoDisconnect,

sum (case when dur = 1 then recurCost else null end) amtDur0to30,

sum (case when dur = 2 then recurCost else null end) amtDur30to60,

sum (case when dur = 3 then recurCost else null end) amtDur60to90,

sum (case when dur = 4 then recurCost else null end) amtDur90to120,

sum (case when dur = 5 then recurCost else null end) amtDur120to180,

sum (case when dur = 6 then recurCost else null end) amtDurOver180,

sum (case when dur > 0 then recurCost else 0 end) amtRegionTotal

,''df\_no\_bill'' flag

from ( select l.area, l.region, ebh.duration dur, ci.recur\_costs recurCost

from xng\_reports.domains\_leaf\_reporting l

left outer join xng\_reports.ebh\_disconnect\_details\_wk ebh on l.domain\_name = ebh.domain\_name

and ebh.df\_has\_bill = ''N''

left outer join vzwnet.circ\_inst ci on ebh.ndf\_seg\_id = ci.circ\_inst\_id

where l.area <> ''OSS'' )

group by rollup (area, region) order by area, region

' ;

EXECUTE IMMEDIATE sql\_stmt;

commit;

END;

END;

/

--------------------------------------------------------

-- DDL for Package Body EHEALTH\_TRAFFIC\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."EHEALTH\_TRAFFIC\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: Ehealth\_traffic\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/9/2013 ramamla 1. Created this package body.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE runEhealthAudit IS

sql\_stmt varchar2(32767);

BEGIN

-- MPLS PERFORMANCE

execute immediate ('truncate table vzw\_ehealth\_traffic\_wk');

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC\_WK

with

mpls\_rtrs\_1 as (

select x.\*, regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(x.port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

from XNG\_MPLS\_RTR\_PORTS\_PARSED x

),

mpls\_rtrs\_2 as (select router\_name, port\_inst\_id, port\_hum\_id,

case when circ\_path\_inst\_id is null then next\_path\_inst\_id

else circ\_path\_inst\_id

end circ\_path\_inst\_id,

case when fix2 like ''% TX/RX'' then replace(fix2, '' TX/RX'','''')

when fix2 like ''% RX'' then replace(fix2, '' RX'','''')

when fix2 like ''% TX'' then replace(fix2, '' TX'','''')

else fix2

end fix2

from mpls\_rtrs\_1

),

ehealth\_data as (select distinct hostname, port\_name,''Y'' has\_traffic , w.extract\_date

from EHEALTH\_PERF\_WK w

where upper(W.ELEMENT\_TYPE)=''MPLS''

and W.TR\_TOKEN > 0

)select distinct hostname, port\_hum\_id, CIRC\_PATH\_INST\_ID, PORT\_INST\_ID, has\_traffic, extract\_date

from mpls\_rtrs\_2, ehealth\_data

where router\_name=hostname

and fix2 = port\_name

'

;

execute immediate sql\_stmt;

commit;

-- CRS\_GIGE PERFORMANCE

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC\_WK

with

crs\_rtrs\_1 as (

select x.\*, regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(x.port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

from XNG\_MPLS\_RTR\_PORTS\_PARSED x

),

crs\_rtrs\_2 as (select router\_name, port\_inst\_id, port\_hum\_id,

case when circ\_path\_inst\_id is null then next\_path\_inst\_id

else circ\_path\_inst\_id

end circ\_path\_inst\_id,

case when fix2 like ''% TX/RX'' then replace(fix2, '' TX/RX'','''')

when fix2 like ''% RX'' then replace(fix2, '' RX'','''')

when fix2 like ''% TX'' then replace(fix2, '' TX'','''')

else fix2

end fix2

from crs\_rtrs\_1

),

ehealth\_data as (select distinct hostname, port\_name,''Y'' has\_traffic , w.extract\_date

from EHEALTH\_PERF\_WK w

where upper(W.ELEMENT\_TYPE)=''CRS\_GIGE''

and W.TR\_TOKEN > 0

)select distinct hostname, port\_hum\_id, CIRC\_PATH\_INST\_ID, PORT\_INST\_ID, has\_traffic, extract\_date

from crs\_rtrs\_2, ehealth\_data

where router\_name=hostname

and fix2 = port\_name

'

;

execute immediate sql\_stmt;

commit;

-- NCM NGMLS PERFORMANCE

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date )

with

paths\_w\_ngmls\_devices as (

select distinct XNE.NGMLS\_DEVICE\_NAME, XNE.EQUIP\_INST\_ID, x.port\_inst\_id, descr,

x.PORT\_HUM\_ID,

regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(x.port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

,cpi.circ\_path\_inst\_id, x.bandwidth

from xng\_reports.xng\_ngMLS\_equip\_wk xne,

vzwnet.epa x,

vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe

where

XNE.VENDOR = ''CISCO''

and x.EQUIP\_INST\_ID = xne.EQUIP\_INST\_ID

and cpi.CIRC\_PATH\_INST\_ID in ( x.CIRC\_PATH\_INST\_ID, x.NEXT\_PATH\_INST\_ID)

and CPE.PORT\_INST\_ID=x.port\_inst\_id

),

details as

( select distinct np.circ\_path\_inst\_id,np.port\_inst\_id,e.hostname,e.port\_name,''Y'' has\_traffic,e.extract\_date, E.BANDWIDTH

,port\_hum\_id, np.bandwidth, fix2

from paths\_w\_ngmls\_devices np

,ehealth\_perf\_wk e

where

E.ELEMENT\_TYPE = ''9010'' and hostname is not null

and NGMLS\_DEVICE\_NAME = E.HOSTNAME

and upper(np.BANDWIDTH) like upper(''1 Gbps%'')

and upper(E.BANDWIDTH) = upper(''GigabitEthernet'')

and tr\_token > 0

and (fix2 like ''%''||port\_name

or fix2 like ''%''||port\_name||'' ''||''%'' )

union

select distinct np.circ\_path\_inst\_id,np.port\_inst\_id,e.hostname,e.port\_name,''Y'' has\_traffic,e.extract\_date, E.BANDWIDTH

,port\_hum\_id, np.bandwidth, fix2

from paths\_w\_ngmls\_devices np

,ehealth\_perf\_wk e

where

E.ELEMENT\_TYPE = ''9010'' and hostname is not null

and NGMLS\_DEVICE\_NAME = E.HOSTNAME

and upper(np.BANDWIDTH) like upper(''10 Gbps%'')

and upper(E.BANDWIDTH) like upper(''TenGi%'')

and tr\_token > 0

and (fix2 like ''%''||port\_name

or fix2 like ''%''||port\_name||'' ''||''%'' )

)select distinct hostname, port\_hum\_id, --fix2, port\_name,

circ\_path\_inst\_id, port\_inst\_id, has\_traffic, extract\_date from details

';

execute immediate sql\_stmt;

commit;

-- CISCO CSR

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date )

with

paths\_w\_cisco\_csr as (

select distinct XNE.CSR\_DEVICE\_NAME, XNE.EQUIP\_INST\_ID,epa.PORT\_INST\_ID,epa.port\_hum\_id

,CPi.CIRC\_PATH\_INST\_ID, CPI.CIRC\_PATH\_HUM\_ID

from xng\_reports.xng\_csr\_parsed xne

,vzwnet.epa

,VZWNET.CIRC\_PATH\_ELEMENT cpe

,vzwnet.circ\_path\_inst cpi

where

XNE.CSR\_DEVICE\_NAME like ''%-CI-%''

and upper(EPA.BANDWIDTH) like ''%GBPS%''

and xne.EQUIP\_INST\_ID = epa.EQUIP\_INST\_ID

and CPE.PORT\_INST\_ID = epa.PORT\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID in (EPA.CIRC\_PATH\_INST\_ID, EPA.NEXT\_PATH\_INST\_ID)

)select distinct e.hostname, port\_hum\_id, circ\_path\_inst\_id, PORT\_INST\_ID, ''Y'' has\_traffic, extract\_date

-- port\_name, csr\_device\_name, , w.equip\_inst\_id, , parse\_status

from

paths\_w\_cisco\_csr w

,xng\_reports.ehealth\_csr\_perf\_wk e

where

parse\_status is null

and tr\_token > 0

and port\_name <> ''1''

and e.hostname = w.csr\_device\_name

and (port\_hum\_id like ''%''||port\_name

or port\_hum\_id like ''%''||port\_name||'' ''||''%'')

'

;

execute immediate sql\_stmt;

commit;

-- SAM CSR

sql\_stmt := ' insert into XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date )

with

paths\_w\_sam\_csr as (

select distinct XNE.CSR\_DEVICE\_NAME, XNE.EQUIP\_INST\_ID,epa.PORT\_INST\_ID,epa.port\_hum\_id,replace(slot,''-'',''/'')||''/''||port\_hum\_id xng\_port\_name

,CPi.CIRC\_PATH\_INST\_ID, CPI.CIRC\_PATH\_HUM\_ID

from xng\_reports.xng\_csr\_parsed xne

,vzwnet.epa

,VZWNET.CIRC\_PATH\_ELEMENT cpe

,vzwnet.circ\_path\_inst cpi

,vzwnet.card\_inst ci

where

XNE.CSR\_DEVICE\_NAME like ''%-AL-%''

and xne.EQUIP\_INST\_ID = epa.EQUIP\_INST\_ID

and CPE.PORT\_INST\_ID = epa.PORT\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID in (EPA.CIRC\_PATH\_INST\_ID, EPA.NEXT\_PATH\_INST\_ID)

and CI.CARD\_INST\_ID = epa.CARD\_INST\_ID

)select distinct e.hostname ,xng\_port\_name, circ\_path\_inst\_id, PORT\_INST\_ID, ''Y'' has\_traffic, extract\_date

-- ,port\_hum\_id,port\_name, csr\_device\_name, , w.equip\_inst\_id, , parse\_status

from

paths\_w\_sam\_csr w

,xng\_reports.ehealth\_csr\_perf\_wk e

where

parse\_status is null

and tr\_token > 0

and port\_name <> ''1''

and e.hostname = w.csr\_device\_name

and xng\_port\_name = port\_name

';

execute immediate sql\_stmt;

commit;

sql\_stmt := ' insert into XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date )

with

paths\_w\_6500 as (

select distinct ei.descr, ei.equip\_inst\_id,epa.PORT\_INST\_ID,epa.port\_hum\_id,slot||''/''||port\_hum\_id xng\_port\_name,

regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

,CPi.CIRC\_PATH\_INST\_ID, CPI.CIRC\_PATH\_HUM\_ID

from vzwnet.equip\_inst ei

,vzwnet.epa

,VZWNET.CIRC\_PATH\_ELEMENT cpe

,vzwnet.circ\_path\_inst cpi

,vzwnet.card\_inst ci

where

ei.descr like ''%-6500-%''

and ei.EQUIP\_INST\_ID = epa.EQUIP\_INST\_ID

and CPE.PORT\_INST\_ID = epa.PORT\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID in (EPA.CIRC\_PATH\_INST\_ID, EPA.NEXT\_PATH\_INST\_ID)

and CI.CARD\_INST\_ID = epa.CARD\_INST\_ID

)

select distinct e.hostname ,xng\_port\_name, circ\_path\_inst\_id, PORT\_INST\_ID, ''Y'' has\_traffic, extract\_date

--,port\_hum\_id,port\_name, fix2

from

paths\_w\_6500 w

,xng\_reports.ehealth\_perf\_wk e

where

parse\_status is null

and tr\_token > 0

and e.ELEMENT\_TYPE = ''6500\_Ethernet''

and w.descr like ''%''||e.HOSTNAME||''%''

and fix2 = port\_name

';

execute immediate sql\_stmt;

commit;

sql\_stmt := 'delete from XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC';

execute immediate sql\_stmt;

commit;

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC select \* from XNG\_REPORTS.VZW\_EHEALTH\_TRAFFIC\_WK';

execute immediate sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in runEhealthAudit(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

END Ehealth\_traffic\_audit;

/

--------------------------------------------------------

-- DDL for Package Body ENB\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."ENB\_AUDIT"

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: eNBAudit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/25/2010 Neeharika 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--- Foundation of eNB Audit

-- In order to avoid using oracle regex , pre parsing some stuff and keeping them in tables

-- pre parsed (/home2/wnea/xng/perljobs/bin/parse\_xng\_eNB on d1) all xng eNB

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure eNbNeVsXngAudit is

CURSOR eNB\_Dups\_in\_Xng is

WITH

eNB\_Equip as

(

select distinct ENB\_NUMBER, equip\_inst\_id, leaf\_domain\_inst\_id

from XNG\_ENB\_PARSED enb

)

select distinct ENB\_NUMBER, count(equip\_inst\_id)

from eNB\_Equip enb

group by ENB\_NUMBER, leaf\_domain\_inst\_id

having count(equip\_inst\_id) > 1

;

/\* Declaring variables\*/

sql\_stmt varchar2(32000);

begin

sql\_stmt:='truncate table '||AUDIT\_TABLE\_NAME;

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: Could not truncate table: '||AUDIT\_TABLE\_NAME ||

'Error: '|| TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

-- create a full audit of discoverd eNB (e/// and alu) vs Xng

sql\_stmt := 'insert into '|| AUDIT\_TABLE\_NAME ||'

WITH

ERIC\_ENB as

(

select enb.\*, ''OSS-RC'' from\_ems

from '|| OSS\_RC\_ENB ||' enb

where regexp\_like( enb.NODE,''^[[:digit:]]{6}'' )

)

,

ALU\_ENB as

(

select enb.\*, ''SAM'' from\_ems

from '|| SAM\_ENB ||' enb

where regexp\_like( enb.NODE,''^[[:digit:]]{6}'' )

--and enb.enb\_status=''LIVE''

)

,

XNG\_EQUIP as

(

select xpe.\*

from xng\_enb\_parsed xpe, market\_prefix\_enb\_naming mkt, clli\_domain\_map cli

where substr(xpe.enb\_number, 1, 3) = mkt.market\_id

and substr(mkt.clli, 1, 8) = cli.clli

and cli.leaf\_domain\_inst\_id = xpe.leaf\_domain\_inst\_id

)

select

substr(enb.node, 1,3) ne\_market\_id

, enb.node

, from\_ems

, IP, connection\_status, synch\_status

, ''BOTH'' match\_code

, parse\_status match\_status

, descr xng\_equip\_name

, case when descr is not null and enb\_number is null then ''Unparseable''

when descr is not null and enb\_number is not null then substr(enb\_number, 1,3)

end xng\_market\_id

, ENB\_NUMBER xng\_ENB\_NUMBER

, equip\_inst\_id

, eq\_class

, has\_multiples

, vendor

, model

, status

,enb.server\_name

,case when enb.connection\_status = ''Connected'' and synch\_status = ''Synchronized'' then ''Up''

else ''Down''

end ne\_status

, ei.type

, null comments

from ERIC\_ENB enb

--full outer

join XNG\_EQUIP ei

on ei.enb\_number = substr(enb.node, 1,6)

union

select substr(enb.node, 1,3) ne\_market\_id

, enb.node

, from\_ems

, null IP

, enb\_status connection\_status

, null synch\_status

, ''BOTH'' match\_code

, parse\_status match\_status

, descr xng\_equip\_name

, case when descr is not null and enb\_number is null then ''Unparseable''

when descr is not null and enb\_number is not null then substr(enb\_number, 1,3)

end xng\_market\_id

, ENB\_NUMBER xng\_ENB\_NUMBER

, equip\_inst\_id

, eq\_class

, has\_multiples

, vendor

, model

, status

,enb.server\_name

,case when enb.enb\_status = ''LIVE'' then ''Up''

else ''Down''

end ne\_status

, ei.type

, null comments

from ALU\_ENB enb

--full outer

join XNG\_EQUIP ei

on ei.enb\_number = substr(enb.node, 1,6)

';

EXECUTE IMMEDIATE sql\_stmt;

-- update the ones that have any problem to NE ONLY

--or (xng\_enb\_number is null and match\_status is not null )

-- eNB should be built as shelves in Xng. So anything that is not a shelf mark it bad match

-- i.e. NE Only

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_code = ''NE Only''

where

match\_code = ''BOTH''

and (aud.eq\_class <> ''S''

or has\_multiples is not null

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated the matched equip match\_code, which are not shelf, to NE Only');

-- eNB should be built as shelves in Xng. So anything that is not a shelf mark it bad match

-- i.e. NE Only

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_status = ''Equipment not built as Shelf in Xng''

where aud.eq\_class <> ''S'' AND match\_status is null

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated the matched equip match\_status, which are not shelf');

--insert NE ONLY from E///

sql\_stmt:='

insert into '|| AUDIT\_TABLE\_NAME ||'

select substr(enb.node, 1,3) ne\_market\_id

, enb.node

, ''OSS-RC''

, IP, connection\_status, synch\_status

, ''NE Only'' match\_code

, null match\_status

, null xng\_equip\_name

, null xng\_market\_id

, null xng\_ENB\_NUMBER

, null equip\_inst\_id

, null eq\_class

, null has\_multiples

, null vendor

, null model

, null status

,enb.server\_name

,case when enb.CONNECTION\_STATUS = ''Connected'' and synch\_status = ''Synchronized'' then ''Up''

else ''Down''

end ne\_status

, '''' type

, null comments

from

'|| OSS\_RC\_ENB ||' enb

,

(select node

FROM '|| OSS\_RC\_ENB ||' enb

where connection\_status = ''Connected''

and synch\_status = ''Synchronized''

minus

select aud.NODE

from '|| AUDIT\_TABLE\_NAME ||' aud

) miss

where miss.node = enb.NODE

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Inserted NE Only e///');

--insert NE ONLY from ALU

sql\_stmt:='

insert into '|| AUDIT\_TABLE\_NAME ||'

select substr(enb.node, 1,3) ne\_market\_id

, enb.node

, ''SAM''

, null IP, enb\_status connection\_status, null synch\_status

, ''NE Only'' match\_code

, null match\_status

, null xng\_equip\_name

, null xng\_market\_id

, null xng\_ENB\_NUMBER

, null equip\_inst\_id

, null eq\_class

, null has\_multiples

, null vendor

, null model

, null status

,enb.server\_name

,case when enb.enb\_status = ''LIVE'' then ''Up''

else ''Down''

end ne\_status

, '''' type

, null comments

from

'|| SAM\_ENB ||' enb

,

(select node

FROM '|| SAM\_ENB ||' enb

--where enb.enb\_status=''LIVE''

minus

select aud.NODE

from '|| AUDIT\_TABLE\_NAME ||' aud

) miss

where miss.node = enb.NODE

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Inserted NE Only ALU');

-- insert Xng\_only. here both ALU and E/// will be inserted

sql\_stmt:='

insert into '|| AUDIT\_TABLE\_NAME ||'

select null ne\_market\_id

, null node

, null from\_ems

, null IP

, null connection\_status

, null synch\_status

, ''Xng Only'' match\_code

, parse\_status match\_status

, xng.descr xng\_equip\_name

, substr(xng.ENB\_NUMBER, 1,3) xng\_market\_id

, xng.ENB\_NUMBER xng\_ENB\_NUMBER

, xng.equip\_inst\_id

, xng.eq\_class

, has\_multiples

, xng.vendor

, xng.model

, xng.STATUS

, null ne\_server\_name

, null ne\_status

, xng.type

, null comments

from xng\_enb\_parsed xng,

(

select EQUIP\_INST\_ID

from xng\_enb\_parsed xng

minus

select AUD.EQUIP\_INST\_ID

from '|| AUDIT\_TABLE\_NAME ||' aud

) miss

where miss.equip\_inst\_id = xng.equip\_inst\_id';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Inserted NE Only ALU and E///');

-- if a new market id is discovered then mark the record to reflect it

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_status = ''New Market Id was discovered''

where

exists ( select 1

from

(

select NE\_MARKET\_ID

from '|| AUDIT\_TABLE\_NAME ||'

minus

select market\_id

from MARKET\_PREFIX\_ENB\_NAMING m

)

where ne\_market\_id = aud.ne\_market\_id

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated rec to indicate that a new Mkt\_id was discovered');

-- an undiscovered Mkt\_id was found in Xng

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_status = ''Unknown Market Id found in Xng''

where

exists ( select 1

from

(

select Xng\_MARKET\_ID

from '|| AUDIT\_TABLE\_NAME ||'

minus

select market\_id

from MARKET\_PREFIX\_ENB\_NAMING m

)

where xng\_market\_id = aud.xng\_market\_id

and aud.XNG\_EQUIP\_NAME is not null

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated record to indicate that an undiscovered Mkt\_id was found in Xng');

-- multiple equip in Xng have the same enb id

/\* if the equipment is in > 1 domain then counting equip\_inst\_id in the table

gives wrong answer . So count enb\_number, count(equip\_inst\_id) \*/

sql\_stmt := 'update '|| AUDIT\_TABLE\_NAME ||' aud

set has\_multiples = ''Y''

, match\_status = nvl2(match\_status, match\_status||'', Multiple equipments have same eNB identfier in Xng'', ''Multiple equipments have same eNB identfier in Xng'')

, match\_code = ''NE Only''

where substr(node, 1,6) = :ENB\_NUMBER';

for rec in eNB\_Dups\_in\_Xng

LOOP

EXECUTE IMMEDIATE sql\_stmt using rec.ENB\_NUMBER;

END LOOP;

commit;

dbms\_output.put\_line('Updated records where discovered enb id was found on multiple equipments in Xng');

-- update the match\_status to indicate duplicates and match\_code NE Only

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_code = ''NE Only'',

match\_status = nvl2(match\_status, match\_status||'', Duplicate in NE'', ''Duplicate in NE'')

where exists (

select node, count(1) dups from '|| SAM\_ENB ||'

where node = aud.node group by node

having count(1) > 1

union all

select node, count(1) dups from '|| OSS\_RC\_ENB ||'

where node = aud.node group by node

having count(1) > 1

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated the match\_code to NE Only [and match\_status] for DUPLICATEs');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in enb\_audit.eNbNeVsXngAudit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

Procedure generateMarketIdSummary is

/\* Declaring variables\*/

sql\_stmt varchar2(32767);

begin

sql\_stmt := 'truncate table eNB\_SUMMARY\_BY\_Mkt\_id\_wrk' ;

execute immediate(sql\_stmt);

sql\_stmt:=' insert into eNB\_SUMMARY\_BY\_Mkt\_id\_wrk

WITH

eNB\_MKT as

(

-- eliminate reserved blocks from the defined eNB market codes

select e.MARKET\_ID, e.lte\_market\_name, e.clli

from MARKET\_PREFIX\_ENB\_NAMING e

)

,

REPORTING\_DOMAINS as

(

-- eliminate domains that will not own an eNB

-- and add Unknown JIC we find stuff that does not belong anywhere else

select cdv.area, cdv.region, cdv.LEAF\_DOMAIN\_INST\_ID

, cdv.PARENT\_DOMAIN\_INST\_ID, cdv.LEAF\_DOMAIN\_NAME

, cdv.clli

from clli\_domain\_map\_v cdv

where cdv.REGION not in (''NNO'', ''OSS'')

)

,

MKT\_DOMAIN\_MAP as

(

select market\_id, e.lte\_market\_name

, e.clli

, nvl(cdv.area, ''Unknown'') area, nvl(cdv.region, ''Unknown'') region

, cdv.leaf\_domain\_inst\_id, cdv.leaf\_domain\_name

from eNB\_MKT e

-- OUR MASTER TABLE might be missing some records.

-- So do a full outer join to capture those

left outer join REPORTING\_DOMAINS cdv

on substr(cdv.clli, 1,6) = substr(e.clli, 1,6)

)

,

REPORTING\_MARKET\_IDS as

(

select case when market\_id is null then substr(e.NODE, 1, 3)

else market\_id

end market\_id

, m.lte\_market\_name

--, m.region

, nvl(area, ''Unknown'') area

, nvl(region, ''Unknown'') region

, leaf\_domain\_inst\_id, leaf\_domain\_name

from MKT\_DOMAIN\_MAP m

full OUTER join '|| OSS\_RC\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where

e.CONNECTION\_STATUS = ''Connected''

and e.SYNCH\_STATUS = ''Synchronized''

UNION

select case when market\_id is null then substr(e.NODE, 1, 3)

else market\_id

end market\_id

, m.lte\_market\_name

, nvl(area, ''Unknown'') area

, nvl(region, ''Unknown'') region

, leaf\_domain\_inst\_id, leaf\_domain\_name

from MKT\_DOMAIN\_MAP m

full OUTER join '|| SAM\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where e.enb\_status=''LIVE''

)

,

eNB\_MKT\_MATCH as

(

select case when market\_id is null then substr(e.NODE, 1, 3)

else market\_id

end market\_id

, m.lte\_market\_name

, leaf\_domain\_inst\_id, leaf\_domain\_name

, e.NODE, e.IP

from MKT\_DOMAIN\_MAP m

right OUTER join '|| OSS\_RC\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where

e.CONNECTION\_STATUS = ''Connected''

and e.SYNCH\_STATUS = ''Synchronized''

UNION

select distinct case when market\_id is null then substr(e.NODE, 1, 3)

else market\_id

end market\_id

, m.lte\_market\_name

, leaf\_domain\_inst\_id, leaf\_domain\_name

, e.NODE, NULL IP

from MKT\_DOMAIN\_MAP m

right OUTER join '|| SAM\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where e.enb\_status=''LIVE''

)

,

TOT as

(

-- since we might discover some new market\_ids that are not in our

-- master table, we need to do put nvl on area and region

select e.market\_id

, count(1) total

from eNB\_MKT\_MATCH e

group by e.market\_id

)

,

BUILT\_AS\_SHELF as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select enb.market\_id, count(1) as\_shelf

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.NODE

where match\_code = ''BOTH''

and eq\_class = ''S''

and ne\_status=''Up''

group by enb.market\_id

)

,

MATCHED\_LIVE as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select enb.market\_id

, count(1) live

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.NODE

-- no need to call check that these are shelves since non-shelf matches

-- are not marked BOTH in the audit

where match\_code = ''BOTH''

and aud.STATUS = ''Live''

and ne\_status=''Up''

group by enb.market\_id

)

,

MATCHED\_N\_LIVE as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select enb.market\_id

, count(1) n\_live

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.node

where match\_code = ''BOTH''

and aud.STATUS <> ''Live''

and ne\_status=''Up''

group by enb.market\_id

)

,

DUP as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select market\_id

, count(1) multiples

from

(

select distinct enb.market\_id, aud.NODE

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.node

where match\_code = ''NE Only''

and aud.HAS\_MULTIPLES = ''Y''

-- and match\_status = ''Multiple equipments have same eNB identfier''

)

group by market\_id

)

,

STATS\_BY\_MKT\_ID as

(

select area

, region

, mkt.market\_id

, mkt.lte\_market\_name

, mkt.leaf\_domain\_name

, sum(total) Total\_eNB

, sum(as\_shelf) shelves, sum(live) Live\_shelves

, sum(n\_live) n\_live\_shelves, sum(multiples) multiple\_shelves

--, round( nvl(sum(live)-sum(multiples),0)/( sum(total)-sum(n\_live)),4)"% Compliant"

from REPORTING\_MARKET\_IDS mkt

left outer join TOT t

on t.market\_id = mkt.market\_id

left outer join BUILT\_AS\_SHELF bas

on bas.market\_id = mkt.market\_id

left outer join MATCHED\_LIVE ml

on ml.market\_id = mkt.market\_id

left outer join MATCHED\_N\_LIVE mnl

on mnl.market\_id = mkt.market\_id

left outer join DUP d

on d.market\_id = mkt.market\_id

group by mkt.area, mkt.region, mkt.MARKET\_ID, mkt.lte\_market\_name, mkt.leaf\_domain\_name

)

select area, region, '''''''' || market\_id

, lte\_market\_name

, leaf\_domain\_name

, total\_eNB

, shelves Built\_as\_Shelf

, live\_shelves Live

, n\_live\_shelves Other

, multiple\_shelves Are\_Duplicated

, case when (nvl(TOTAL\_ENB, 0) - nvl(n\_live\_shelves, 0) > 0) then round (nvl(live\_shelves, 0)/nullif (nvl(TOTAL\_ENB, 0) - nvl(n\_live\_shelves, 0), 0) \*100, 2) else 100 end total\_compliance

from STATS\_BY\_MKT\_ID

ORDER BY area, region, market\_id

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in enb\_audit.generateMarketIdSummary(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

End generateMarketIdSummary;

Procedure generateRegionalSummary is

/\* Declaring variables\*/

sql\_stmt varchar2(32767);

begin

sql\_stmt := 'truncate table ENB\_SUMMARY\_BY\_REGION\_WRK' ;

execute immediate(sql\_stmt);

sql\_stmt := 'insert into ENB\_SUMMARY\_BY\_REGION\_WRK columns

(area, region, total\_enb, built\_as\_shelf, live, other, are\_duplicated, enb\_reported, total\_compliance)

select area, region, sum( mkt.TOTAL\_ENB)

, sum(mkt.BUILT\_AS\_SHELF ), sum(mkt.LIVE ), sum(mkt.OTHER), sum(mkt.ARE\_DUPLICATED)

, sum( nvl(mkt.TOTAL\_ENB, 0)) - sum(nvl(mkt.OTHER, 0)) enb\_reported

,round (sum(nvl(mkt.LIVE, 0))/decode (sum( nvl(mkt.TOTAL\_ENB, 0)) - sum(nvl(mkt.OTHER, 0)), 0, 1, sum(nvl(mkt.TOTAL\_ENB, 0)) - sum(nvl(mkt.OTHER, 0))) \*100, 2) total\_compliance

from ENB\_SUMMARY\_BY\_MKT\_ID\_WRK mkt

group by rollup(area, region)

order by area, region

';

EXECUTE IMMEDIATE sql\_stmt;

-- Added so missing area/regions will show with 0 count

sql\_stmt := 'insert into ENB\_SUMMARY\_BY\_REGION\_WRK SELECT d.area, d.region, 0, 0, 0, 0, 0, 0, 0 FROM DOMAINS\_REGIONAL\_REPORTING d left join enb\_summary\_by\_region\_wrk e on d.area = e.area and d.region = e.region WHERE d.area NOT IN (''OSS'', ''NNO'') and e.area is null';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in enb\_audit.generateRegionalSummary(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

End generateRegionalSummary;

PROCEDURE copy\_enb\_wrk IS

sqlStmt varchar2(32000);

v\_ready xng\_reports.all\_processes.is\_ready%type;

BEGIN

select is\_ready

into v\_ready

from xng\_reports.all\_processes

where process\_name = 'ENB\_AUDIT';

IF v\_ready = 'Y' then

sqlStmt:='DELETE FROM NE\_VS\_XNG\_ENB\_AUDIT';

execute immediate sqlStmt;

sqlStmt := 'insert into NE\_VS\_XNG\_ENB\_AUDIT select \* from '||AUDIT\_TABLE\_NAME;

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating NE\_VS\_XNG\_ENB\_AUDIT');

sqlStmt := 'truncate table ENB\_SUMMARY\_BY\_REGION';

execute immediate sqlStmt;

sqlStmt := 'insert into ENB\_SUMMARY\_BY\_REGION select \* from ENB\_SUMMARY\_BY\_REGION\_WRK';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating ENB\_SUMMARY\_BY\_REGION');

sqlStmt := 'truncate table eNB\_SUMMARY\_BY\_Mkt\_id';

execute immediate sqlStmt;

sqlStmt := 'insert into eNB\_SUMMARY\_BY\_Mkt\_id select \* from eNB\_SUMMARY\_BY\_Mkt\_id\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating eNB\_SUMMARY\_BY\_Mkt\_id');

END IF;

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('ENB\_AUDIT',4000,SubStr('Error in copy\_enb\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('ENB\_AUDIT','STATUS\_FAILURE','N');

END;

PROCEDURE truncate\_enb\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from '||AUDIT\_TABLE\_NAME;

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting '||AUDIT\_TABLE\_NAME);

sqlStmt := 'delete from ENB\_SUMMARY\_BY\_REGION\_WRK';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting ENB\_SUMMARY\_BY\_REGION\_WRK');

sqlStmt := 'delete from eNB\_SUMMARY\_BY\_Mkt\_id\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting eNB\_SUMMARY\_BY\_Mkt\_id\_wrk');

END;

PROCEDURE restore\_enb\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt:='DELETE FROM '||AUDIT\_TABLE\_NAME;

execute immediate sqlStmt;

sqlStmt := 'insert into '||AUDIT\_TABLE\_NAME||' select \* from NE\_VS\_XNG\_ENB\_AUDIT ';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating NE\_VS\_XNG\_ENB\_AUDIT');

sqlStmt := 'truncate table ENB\_SUMMARY\_BY\_REGION';

execute immediate sqlStmt;

sqlStmt := 'insert into ENB\_SUMMARY\_BY\_REGION select \* from ENB\_SUMMARY\_BY\_REGION\_WRK';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating nortel\_cdma\_audit\_reg\_summ\_wrk');

sqlStmt := 'truncate table eNB\_SUMMARY\_BY\_Mkt\_id';

execute immediate sqlStmt;

sqlStmt := 'insert into eNB\_SUMMARY\_BY\_Mkt\_id select \* from eNB\_SUMMARY\_BY\_Mkt\_id\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating eNB\_SUMMARY\_BY\_Mkt\_id');

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('ENB\_AUDIT',4000,SubStr('Error in restore\_enb\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('ENB\_AUDIT','STATUS\_FAILURE','N');

END;

END eNB\_Audit;

/

--------------------------------------------------------

-- DDL for Package Body ENB\_AUDIT\_WD

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."ENB\_AUDIT\_WD"

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: eNBAudit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/25/2010 Neeharika 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--- Foundation of eNB Audit

-- In order to avoid using oracle regex , pre parsing some stuff and keeping them in tables

-- pre parsed (/home2/wnea/xng/perljobs/bin/parse\_xng\_eNB on d1) all xng eNB

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure eNbNeVsXngAudit is

CURSOR DUPS\_IN\_XNG is

WITH

eNB\_Equip as

(

select distinct ENB\_NUMBER, equip\_inst\_id

from XNG\_ENB\_PARSED enb

)

select distinct ENB\_NUMBER, count(equip\_inst\_id)

from eNB\_Equip enb

group by ENB\_NUMBER

having count(equip\_inst\_id) > 1

;

/\* Declaring variables\*/

sql\_stmt varchar2(32000);

begin

sql\_stmt:='truncate table '||AUDIT\_TABLE\_NAME;

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: Could not truncate table: '||AUDIT\_TABLE\_NAME ||

'Error: '|| TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

-- create a full audit of discoverd eNB (e/// and alu) vs Xng

sql\_stmt := 'insert into '|| AUDIT\_TABLE\_NAME ||'

WITH

ERIC\_ENB as

(

select enb.\*, ''OSS-RC'' from\_ems

from '|| OSS\_RC\_ENB ||' enb

where regexp\_like( enb.NODE,''^[[:digit:]]{6}'' )

)

,

ALU\_ENB as

(

select enb.\*, ''XMS'' from\_ems

from '|| XMS\_ENB ||' enb

where regexp\_like( enb.NODE,''^[[:digit:]]{6}'' )

and enb.enb\_status=''LIVE''

)

,

XNG\_EQUIP as

(

select xpe.\*, ei.VENDOR, ei.MODEL, status

from xng\_enb\_parsed xpe

join vzwnet.equip\_inst ei

on ei.equip\_inst\_id = xpe.EQUIP\_INST\_ID

)

select

substr(enb.node, 1,3) ne\_market\_id

, enb.node

, from\_ems

, IP, connection\_status, synch\_status

, ''BOTH'' match\_code

, parse\_status match\_status

, descr xng\_equip\_name

, case when descr is not null and enb\_number is null then ''Unparseable''

when descr is not null and enb\_number is not null then substr(enb\_number, 1,3)

end xng\_market\_id

, ENB\_NUMBER xng\_ENB\_NUMBER

, equip\_inst\_id

, eq\_class

, has\_multiples

, vendor

, model

, status

,enb.server\_name

,case when enb.connection\_status = ''Connected'' and synch\_status = ''Synchronized'' then ''Up''

else ''Down''

end ne\_status

from ERIC\_ENB enb

--full outer

join XNG\_EQUIP ei

on ei.enb\_number = substr(enb.node, 1,6)

union

select substr(enb.node, 1,3) ne\_market\_id

, enb.node

, from\_ems

, null IP

, enb\_status connection\_status

, null synch\_status

, ''BOTH'' match\_code

, parse\_status match\_status

, descr xng\_equip\_name

, case when descr is not null and enb\_number is null then ''Unparseable''

when descr is not null and enb\_number is not null then substr(enb\_number, 1,3)

end xng\_market\_id

, ENB\_NUMBER xng\_ENB\_NUMBER

, equip\_inst\_id

, eq\_class

, has\_multiples

, vendor

, model

, status

,enb.server\_name

,case when enb.enb\_status = ''LIVE'' then ''Up''

else ''Down''

end ne\_status

from ALU\_ENB enb

--full outer

join XNG\_EQUIP ei

on ei.enb\_number = substr(enb.node, 1,6)

';

EXECUTE IMMEDIATE sql\_stmt;

-- update the ones that have any problem to NE ONLY

--or (xng\_enb\_number is null and match\_status is not null )

-- eNB should be cuilt as shelves in Xng. So anything that is not a shelf mark it bad match

-- i.e. NE Only

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_code = ''NE Only''

where

match\_code = ''BOTH''

and (aud.eq\_class <> ''S''

or has\_multiples is not null

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated the matched equip match\_code, which are not shelf, to NE Only');

-- eNB should be cuilt as shelves in Xng. So anything that is not a shelf mark it bad match

-- i.e. NE Only

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_status = ''Equipment not built as Shelf in Xng''

where aud.eq\_class <> ''S'' AND match\_status is null

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated the matched equip match\_status, which are not shelf');

--insert NE ONLY from E///

sql\_stmt:='

insert into '|| AUDIT\_TABLE\_NAME ||'

select substr(enb.node, 1,3) ne\_market\_id

, enb.node

, ''OSS-RC''

, IP, connection\_status, synch\_status

, ''NE Only'' match\_code

, null match\_status

, null xng\_equip\_name

, null xng\_market\_id

, null xng\_ENB\_NUMBER

, null equip\_inst\_id

, null eq\_class

, null has\_multiples

, null vendor

, null model

, null status

,enb.server\_name

,case when enb.CONNECTION\_STATUS = ''Connected'' and synch\_status = ''Synchronized'' then ''Up''

else ''Down''

end ne\_status

from

'|| OSS\_RC\_ENB ||' enb

,

(select node

FROM '|| OSS\_RC\_ENB ||' enb

where connection\_status = ''Connected''

and synch\_status = ''Synchronized''

minus

select aud.NODE

from '|| AUDIT\_TABLE\_NAME ||' aud

) miss

where miss.node = enb.NODE

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Inserted NE Only e///');

--insert NE ONLY from ALU

sql\_stmt:='

insert into '|| AUDIT\_TABLE\_NAME ||'

select substr(enb.node, 1,3) ne\_market\_id

, enb.node

, ''XMS''

, null IP, enb\_status connection\_status, null synch\_status

, ''NE Only'' match\_code

, null match\_status

, null xng\_equip\_name

, null xng\_market\_id

, null xng\_ENB\_NUMBER

, null equip\_inst\_id

, null eq\_class

, null has\_multiples

, null vendor

, null model

, null status

,enb.server\_name

,case when enb.enb\_status = ''LIVE'' then ''Up''

else ''Down''

end ne\_status

from

'|| XMS\_ENB ||' enb

,

(select node

FROM '|| XMS\_ENB ||' enb

where enb.enb\_status=''LIVE''

minus

select aud.NODE

from '|| AUDIT\_TABLE\_NAME ||' aud

) miss

where miss.node = enb.NODE

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Inserted NE Only ALU');

-- insert Xng\_only. here both ALU and E/// will be inserted

sql\_stmt:='

insert into '|| AUDIT\_TABLE\_NAME ||'

select null ne\_market\_id

, null node

, null from\_ems

, null IP, null connection\_status, null synch\_status

, ''Xng Only'' match\_code

, parse\_status match\_status

, ei.descr xng\_equip\_name

, substr(xng.ENB\_NUMBER, 1,3) xng\_market\_id

, xng.ENB\_NUMBER xng\_ENB\_NUMBER

, ei.equip\_inst\_id

, ei.eq\_class

, has\_multiples

, ei.vendor

, ei.model

, ei.STATUS

,null ne\_server\_name

, null ne\_status

from

xng\_enb\_parsed xng

,

(

select EQUIP\_INST\_ID

from xng\_enb\_parsed xng

minus

select AUD.EQUIP\_INST\_ID

from '|| AUDIT\_TABLE\_NAME ||' aud

) miss

, vzwnet.equip\_inst ei

where miss.equip\_inst\_id = xng.equip\_inst\_id

and ei.EQUIP\_INST\_ID = xng.EQUIP\_INST\_ID

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Inserted NE Only ALU and E///');

-- if a new market id is discovered then mark the record to reflect it

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_status = ''New Market Id was discovered''

where

exists ( select 1

from

(

select NE\_MARKET\_ID

from '|| AUDIT\_TABLE\_NAME ||'

minus

select market\_id

from MARKET\_PREFIX\_ENB\_NAMING m

)

where ne\_market\_id = aud.ne\_market\_id

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated rec to indicate that a new Mkt\_id was discovered');

-- an undiscovered Mkt\_id was found in Xng

sql\_stmt:='

update '|| AUDIT\_TABLE\_NAME ||' aud

set match\_status = ''Unknown Market Id found in Xng''

where

exists ( select 1

from

(

select Xng\_MARKET\_ID

from '|| AUDIT\_TABLE\_NAME ||'

minus

select market\_id

from MARKET\_PREFIX\_ENB\_NAMING m

)

where xng\_market\_id = aud.xng\_market\_id

and aud.XNG\_EQUIP\_NAME is not null

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Updated record to indicate that an undiscovered Mkt\_id was found in Xng');

-- multiple equip in Xng have the same enb id

sql\_stmt:='update '|| AUDIT\_TABLE\_NAME ||' aud

set has\_multiples = ''Y''

, match\_status = ''Multiple equipments have same eNB identfier''

, match\_code = ''NE Only''

where substr(node, 1,6) = upper(:enb\_number)

';

for rec in DUPS\_IN\_XNG

loop

EXECUTE IMMEDIATE sql\_stmt using rec.enb\_number;

dbms\_output.put\_line('Dup:'||rec.enb\_number);

end loop;

commit;

dbms\_output.put\_line('Updated records where discovered enb id was found on multiple equipments in Xng');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in enb\_audit.eNbNeVsXngAudit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

Procedure generateRegionalSummary is

/\* Declaring variables\*/

sql\_stmt varchar2(32767);

begin

sql\_stmt := 'truncate table ENB\_SUMMARY\_BY\_REGION\_WRK' ;

execute immediate(sql\_stmt);

sql\_stmt:=' insert into ENB\_SUMMARY\_BY\_REGION\_WRK

WITH

-- master list queried from FDW

eNB\_MKT as

(

select e.MARKET\_ID, e.lte\_market\_name, e.clli

from MARKET\_PREFIX\_ENB\_NAMING e

)

,

DOMAINS\_BY\_CLLI as

(

-- eliminate domains that will not own an eNB

-- and add Unknown JIC we find stuff that does not belong anywhere else

select cdv.area, cdv.region, cdv.LEAF\_DOMAIN\_INST\_ID

, cdv.PARENT\_DOMAIN\_INST\_ID, cdv.LEAF\_DOMAIN\_NAME

, cdv.clli

from clli\_domain\_map\_v cdv

where cdv.REGION not in (''NNO'', ''OSS'')

union

select ''Unknown'' area, ''Unknown'' region

, 0 domain\_inst\_id, 0, ''Unknown'' domain\_name, ''Unknown'' clli

from dual

)

,

MKT\_DOMAIN\_MAP as

(

-- map eNB markets to xng leaf\_domains based on CLLI

select distinct market\_id, e.lte\_market\_name

, e.clli

, nvl(cdv.area, ''Unknown'') area, nvl(cdv.region, ''Unknown'') region

, cdv.leaf\_domain\_inst\_id, cdv.leaf\_domain\_name

from eNB\_MKT e

left outer join DOMAINS\_BY\_CLLI cdv

on substr(cdv.clli,1,6) = substr(e.clli, 1,6)

)

,

eNB\_MKT\_MATCH as

(

select area, region

, m.market\_id, m.lte\_market\_name

, leaf\_domain\_inst\_id, leaf\_domain\_name

, e.NODE--, NULL IP

from MKT\_DOMAIN\_MAP m

right OUTER

join '|| XMS\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where e.enb\_status=''LIVE''

UNION

select area, region

, m.market\_id, m.lte\_market\_name

, leaf\_domain\_inst\_id, leaf\_domain\_name

, e.NODE--, e.IP

from MKT\_DOMAIN\_MAP m

right OUTER

join '|| OSS\_RC\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where

e.CONNECTION\_STATUS = ''Connected''

and e.SYNCH\_STATUS = ''Synchronized''

)

,

TOT as

(

-- since we might discover some new market\_ids that are not in our

-- master table, we need to do all this jugglery

select nvl(leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, count(1) total

from eNB\_MKT\_MATCH

group by leaf\_domain\_inst\_id

--order by area, region;

)

,

BUILT\_AS\_SHELF as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select nvl(leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, count(1) as\_shelf

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.NODE

where match\_code = ''BOTH''

and eq\_class = ''S''

and ne\_status=''Up''

group by leaf\_domain\_inst\_id

)

,

MATCHED\_LIVE as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select nvl(leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, count(1) live

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.NODE

-- no need to call check that these are shelves since non-shelf matches

-- are not marked BOTH in the audit

where match\_code = ''BOTH''

and aud.STATUS = ''Live''

and ne\_status=''Up''

group by leaf\_domain\_inst\_id

)

,

MATCHED\_N\_LIVE as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select nvl(leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, count(1) n\_live

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.node

where match\_code = ''BOTH''

and aud.STATUS <> ''Live''

and ne\_status=''Up''

group by leaf\_domain\_inst\_id

)

,

DUP as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select nvl(leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, count(1) multiples

from

(

select distinct leaf\_domain\_inst\_id, aud.NODE

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.node

where aud.HAS\_MULTIPLES = ''Y''

)

group by leaf\_domain\_inst\_id

)

,

REPORTING\_DOMAINS as

(

select distinct cdv.area, cdv.region, cdv.LEAF\_DOMAIN\_INST\_ID

, cdv.PARENT\_DOMAIN\_INST\_ID, cdv.LEAF\_DOMAIN\_NAME

from DOMAINS\_BY\_CLLI cdv

)

select drr.area, drr.region

, sum(total) Total\_eNB

, sum(as\_shelf) Built\_as\_Shelf

, sum(live) Live

, sum(n\_live) Other

, sum(multiples) Are\_Duplicated

, sum(nvl(total,0)) - nvl(sum(n\_live),0) enb\_reported -- nsp added

, 0 -- % compliance will be updated below by a update stmt. -- nsp added

from REPORTING\_DOMAINS drr

left outer join TOT t

On t.Leaf\_Domain\_Inst\_Id = Drr.leaf\_domain\_inst\_id

left outer join BUILT\_AS\_SHELF bas

on bas.leaf\_domain\_inst\_id = drr.leaf\_domain\_inst\_id

left outer join MATCHED\_LIVE ml

on ml.leaf\_domain\_inst\_id = drr.leaf\_domain\_inst\_id

left outer join MATCHED\_N\_LIVE mnl

on mnl.leaf\_domain\_inst\_id = drr.leaf\_domain\_inst\_id

left outer join DUP d

on d.leaf\_domain\_inst\_id = drr.leaf\_domain\_inst\_id

group by rollup (drr.area, drr.region)

order by drr.area, drr.region

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

update ENB\_SUMMARY\_BY\_REGION\_WRK r

set r.total\_compliance = round(nvl(live,0)/enb\_reported \* 100 ,2)

where r.ENB\_REPORTED <> 0

and r.ENB\_REPORTED is not null;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in enb\_audit.generateRegionalSummary(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

End;

Procedure generateMarketIdSummary is

/\* Declaring variables\*/

sql\_stmt varchar2(32767);

begin

sql\_stmt := 'truncate table eNB\_SUMMARY\_BY\_Mkt\_id\_wrk' ;

execute immediate(sql\_stmt);

sql\_stmt:=' insert into eNB\_SUMMARY\_BY\_Mkt\_id\_wrk

WITH

eNB\_MKT as

(

-- eliminate reserved blocks from the defined eNB market codes

select e.MARKET\_ID, e.lte\_market\_name, e.clli

from MARKET\_PREFIX\_ENB\_NAMING e

)

,

REPORTING\_DOMAINS as

(

-- eliminate domains that will not own an eNB

-- and add Unknown JIC we find stuff that does not belong anywhere else

select cdv.area, cdv.region, cdv.LEAF\_DOMAIN\_INST\_ID

, cdv.PARENT\_DOMAIN\_INST\_ID, cdv.LEAF\_DOMAIN\_NAME

, cdv.clli

from clli\_domain\_map\_v cdv

where cdv.REGION not in (''NNO'', ''OSS'')

)

,

MKT\_DOMAIN\_MAP as

(

select market\_id, e.lte\_market\_name

, e.clli

, nvl(cdv.area, ''Unknown'') area, nvl(cdv.region, ''Unknown'') region

, cdv.leaf\_domain\_inst\_id, cdv.leaf\_domain\_name

from eNB\_MKT e

-- OUR MASTER TABLE might be missing some records.

-- So do a full outer join to capture those

left outer join REPORTING\_DOMAINS cdv

on substr(cdv.clli, 1,6) = substr(e.clli, 1,6)

)

,

REPORTING\_MARKET\_IDS as

(

select case when market\_id is null then substr(e.NODE, 1, 3)

else market\_id

end market\_id

, m.lte\_market\_name

--, m.region

, nvl(area, ''Unknown'') area

, nvl(region, ''Unknown'') region

, leaf\_domain\_inst\_id, leaf\_domain\_name

from MKT\_DOMAIN\_MAP m

full OUTER join '|| OSS\_RC\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where

e.CONNECTION\_STATUS = ''Connected''

and e.SYNCH\_STATUS = ''Synchronized''

UNION

select case when market\_id is null then substr(e.NODE, 1, 3)

else market\_id

end market\_id

, m.lte\_market\_name

, nvl(area, ''Unknown'') area

, nvl(region, ''Unknown'') region

, leaf\_domain\_inst\_id, leaf\_domain\_name

from MKT\_DOMAIN\_MAP m

full OUTER join '|| XMS\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where e.enb\_status=''LIVE''

)

,

eNB\_MKT\_MATCH as

(

select case when market\_id is null then substr(e.NODE, 1, 3)

else market\_id

end market\_id

, m.lte\_market\_name

, leaf\_domain\_inst\_id, leaf\_domain\_name

, e.NODE, e.IP

from MKT\_DOMAIN\_MAP m

right OUTER join '|| OSS\_RC\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where

e.CONNECTION\_STATUS = ''Connected''

and e.SYNCH\_STATUS = ''Synchronized''

UNION

select distinct case when market\_id is null then substr(e.NODE, 1, 3)

else market\_id

end market\_id

, m.lte\_market\_name

, leaf\_domain\_inst\_id, leaf\_domain\_name

, e.NODE, NULL IP

from MKT\_DOMAIN\_MAP m

right OUTER join '|| XMS\_ENB ||' e

on substr(e.NODE, 1, 3) = m.market\_id

where e.enb\_status=''LIVE''

)

,

TOT as

(

-- since we might discover some new market\_ids that are not in our

-- master table, we need to do put nvl on area and region

select e.market\_id

, count(1) total

from eNB\_MKT\_MATCH e

group by e.market\_id

)

,

BUILT\_AS\_SHELF as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select enb.market\_id, count(1) as\_shelf

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.NODE

where match\_code = ''BOTH''

and eq\_class = ''S''

and ne\_status=''Up''

group by enb.market\_id

)

,

MATCHED\_LIVE as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select enb.market\_id

, count(1) live

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.NODE

-- no need to call check that these are shelves since non-shelf matches

-- are not marked BOTH in the audit

where match\_code = ''BOTH''

and aud.STATUS = ''Live''

and ne\_status=''Up''

group by enb.market\_id

)

,

MATCHED\_N\_LIVE as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select enb.market\_id

, count(1) n\_live

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.node

where match\_code = ''BOTH''

and aud.STATUS <> ''Live''

and ne\_status=''Up''

group by enb.market\_id

)

,

DUP as

(

-- since we are reporting the discovered spans that matched

-- we dont need to worry about "mkt ids, that were not discovered"

select market\_id

, count(1) multiples

from

(

select distinct enb.market\_id, aud.NODE

from '|| AUDIT\_TABLE\_NAME ||' aud

join eNB\_MKT\_MATCH enb

on enb.node = aud.node

where match\_code = ''NE Only''

and aud.HAS\_MULTIPLES = ''Y''

-- and match\_status = ''Multiple equipments have same eNB identfier''

)

group by market\_id

)

,

STATS\_BY\_MKT\_ID as

(

select area

, region

, mkt.market\_id

, mkt.lte\_market\_name

, mkt.leaf\_domain\_name

, sum(total) Total\_eNB

, sum(as\_shelf) shelves, sum(live) Live\_shelves

, sum(n\_live) n\_live\_shelves, sum(multiples) multiple\_shelves

--, round( nvl(sum(live)-sum(multiples),0)/( sum(total)-sum(n\_live)),4)"% Compliant"

from REPORTING\_MARKET\_IDS mkt

left outer join TOT t

on t.market\_id = mkt.market\_id

left outer join BUILT\_AS\_SHELF bas

on bas.market\_id = mkt.market\_id

left outer join MATCHED\_LIVE ml

on ml.market\_id = mkt.market\_id

left outer join MATCHED\_N\_LIVE mnl

on mnl.market\_id = mkt.market\_id

left outer join DUP d

on d.market\_id = mkt.market\_id

group by mkt.area, mkt.region, mkt.MARKET\_ID, mkt.lte\_market\_name, mkt.leaf\_domain\_name

)

select area, region, '''''''' || market\_id

, lte\_market\_name

, leaf\_domain\_name

, total\_eNB

, shelves Built\_as\_Shelf

, live\_shelves Live

, n\_live\_shelves Other

, multiple\_shelves Are\_Duplicated

from STATS\_BY\_MKT\_ID

ORDER BY area, region, market\_id

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in enb\_audit.generateMarketIdSummary(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

End;

PROCEDURE copy\_enb\_wrk IS

sqlStmt varchar2(32000);

v\_ready xng\_reports.all\_processes.is\_ready%type;

BEGIN

select is\_ready

into v\_ready

from xng\_reports.all\_processes

where process\_name = 'ENB\_AUDIT';

IF v\_ready = 'Y' then

sqlStmt:='DELETE FROM NE\_VS\_XNG\_ENB\_AUDIT';

execute immediate sqlStmt;

sqlStmt := 'insert into NE\_VS\_XNG\_ENB\_AUDIT select \* from '||AUDIT\_TABLE\_NAME;

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating NE\_VS\_XNG\_ENB\_AUDIT');

sqlStmt := 'truncate table ENB\_SUMMARY\_BY\_REGION';

execute immediate sqlStmt;

sqlStmt := 'insert into ENB\_SUMMARY\_BY\_REGION select \* from ENB\_SUMMARY\_BY\_REGION\_WRK';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating ENB\_SUMMARY\_BY\_REGION');

sqlStmt := 'truncate table eNB\_SUMMARY\_BY\_Mkt\_id';

execute immediate sqlStmt;

sqlStmt := 'insert into eNB\_SUMMARY\_BY\_Mkt\_id select \* from eNB\_SUMMARY\_BY\_Mkt\_id\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating eNB\_SUMMARY\_BY\_Mkt\_id');

END IF;

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('ENB\_AUDIT',4000,SubStr('Error in copy\_enb\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('ENB\_AUDIT','STATUS\_FAILURE','N');

END;

PROCEDURE truncate\_enb\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from '||AUDIT\_TABLE\_NAME;

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting '||AUDIT\_TABLE\_NAME);

sqlStmt := 'delete from ENB\_SUMMARY\_BY\_REGION\_WRK';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting ENB\_SUMMARY\_BY\_REGION\_WRK');

sqlStmt := 'delete from eNB\_SUMMARY\_BY\_Mkt\_id\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting eNB\_SUMMARY\_BY\_Mkt\_id\_wrk');

END;

PROCEDURE restore\_enb\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt:='DELETE FROM '||AUDIT\_TABLE\_NAME;

execute immediate sqlStmt;

sqlStmt := 'insert into '||AUDIT\_TABLE\_NAME||' select \* from NE\_VS\_XNG\_ENB\_AUDIT ';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating NE\_VS\_XNG\_ENB\_AUDIT');

sqlStmt := 'truncate table ENB\_SUMMARY\_BY\_REGION';

execute immediate sqlStmt;

sqlStmt := 'insert into ENB\_SUMMARY\_BY\_REGION select \* from ENB\_SUMMARY\_BY\_REGION\_WRK';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating nortel\_cdma\_audit\_reg\_summ\_wrk');

sqlStmt := 'truncate table eNB\_SUMMARY\_BY\_Mkt\_id';

execute immediate sqlStmt;

sqlStmt := 'insert into eNB\_SUMMARY\_BY\_Mkt\_id select \* from eNB\_SUMMARY\_BY\_Mkt\_id\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating eNB\_SUMMARY\_BY\_Mkt\_id');

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('ENB\_AUDIT',4000,SubStr('Error in restore\_enb\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('ENB\_AUDIT','STATUS\_FAILURE','N');

END;

END eNB\_Audit\_wd;

/

--------------------------------------------------------

-- DDL for Package Body EXECUTIVE\_SUMMARY\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_PKG" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate exec\_summ\_rpt

\* first in\_progress will be generated and then moved to rptd

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure executiveSummaryReported is

BEGIN

processName := 'EXECUTIVE\_SUMMARY\_RPTD';

BEGIN

-- Tell watchDdog the process has started

watchdog.updateprocessstart (processName);

BEGIN

-- create a snapshot of previous month data

createReportedSnapshot ();

-- copy current reported data to previous

copyFromReportedToPrevious ();

-- previous data over 13 months old

deleteOldPrevious ();

-- copy current previous data to previous bak

createPreviousBackup ();

-- Generate Executive Suumary

genExecutiveSummary();

-- copy generated in\_progress to reported

copyFromInProgressToReported ();

-- mark successful completion of the task

watchdog.updateprocessend (processName, 'STATUS\_SUCCESS', 'Y');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

-- mark failed completion of the task

watchdog.updateprocessend (processName, 'STATUS\_FAILURE', 'N');

END;

END;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate exec\_summ

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure executiveSummaryInProgress is

BEGIN

processName := 'EXECUTIVE\_SUMMARY';

-- Tell watchDdog the process has started

BEGIN

watchdog.updateprocessstart (processName);

BEGIN

genExecutiveSummary();

-- tell watchDog process completed successfully

watchdog.updateprocessend (processName, 'STATUS\_SUCCESS', 'Y');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.updateprocessend (processName, 'STATUS\_FAILURE', 'N');

END;

END;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate a snapshot of the previous month data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure createSnapshot is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from EXECUTIVE\_SUMMARY to EXECUTIVE\_SUMMARY\_SNAPSHOT

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMMARY\_SNAPSHOT';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMMARY\_SNAPSHOT columns (

area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting EXECUTIVE\_SUMMARY\_SNAPSHOT '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into EXECUTIVE\_SUMMARY\_SNAPSHOT table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating EXECUTIVE\_SUMMARY\_SNAPSHOT '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc EXECUTIVE\_SUMMARY\_SNAPSHOT table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END createSnapshot;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate a snapshot of the previous month data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure createReportedSnapshot is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from EXECUTIVE\_SUMMARY\_REPORTED to EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT columns (

area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_reported';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

-- Copy data from EXECUTIVE\_SUMMARY\_REPORTED\_BAK to EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT columns (

area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_reported\_bak';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

-- Copy data from EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK to EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT ';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT columns (

area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_previous\_bak';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END createReportedSnapshot;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* procedure that populates exec\_summ table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure genExecutiveSummary is

-- get distinct areas

cursor getArea is

select distinct area

from xng\_reports.domains\_regional\_reporting

where area<>'OSS'

order by area

;

sql\_stmt varchar2(32000);

begin

-- backup executive\_summary table

createSnapshot();

sql\_stmt:='truncate table xng\_reports.executive\_summary\_work';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- insert all area region rows in the exec summary table.

-- later metrics for area/region will be populated

BEGIN

sql\_Stmt := 'insert into xng\_reports.executive\_summary\_work columns(area, region, extract\_date)

select distinct area, region, to\_date(sysdate)

from xng\_reports.domains\_regional\_reporting

where area<>''OSS''

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- populate regional metrics (col) for the following in the exec\_summ table

-- these have to be calculated since these are not persisted anywhere

for area\_row in getArea

loop

-- calc VSM equip data

populateVSMData(area\_row.area);

-- calc Segments with Path data

populateSegmentData(area\_row.area);

end loop;

-- this only stores regional and area totals and not grand totals

-- get data from xngstds audit,

populateXNGAuditData();

-- create aggregate rollup of all entered metrics so far and store in exec\_summ

populateMetricTotals();

-- get data from btp audit. Calc area and grand totals as well

populateBTPAuditData();

-- get data from cross connect audit

populateCrossConAuditData;

-- processes below already compute and put stats in table with grand totals

-- just populate the columns from these tables

-- add % OBS for the enterprise. This has region/area/enterprise #.

-- so not calculating them again.

populateOBSData();

-- Now all metric col are fully populated.

-- calc Overall Compliance % for the region/area/enterprise

populateOverallPct();

-- insert prev months metric grand total as a row

populatePrevGrandTotalsRow();

-- for each metric calc deltas between grand total of the metric of current month vs prev month

-- insert these as a row

populateDeltasRow();

-- populate prev month compliance % in a col and

-- delta between current month's total\_compliance and prev month's total compliance

populatePrevAndDeltasCol();

--Now copy from Work to InProgress

copyFromWorkToInProgress ();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end genExecutiveSummary;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates data for '% Switch/DACS Audited in XNG vs VSM' and

\* 'Equip Count' cols. in ES. report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateVSMData(area in varchar2) is

cursor getRegions(ARG\_IN\_AREA varchar2) is

select distinct region from

xng\_reports.domains\_regional\_reporting drr

where drr.area=''||ARG\_IN\_AREA||'';

cursor getDacsData(ARG\_IN\_AREA varchar2, ARG\_IN\_REGION varchar2) is

with

disc\_nid as

(

select distinct attr\_value node\_id

from vzwnet.EQUIP\_ATTR\_SETTINGS

where VAL\_ATTR\_INST\_ID=(select val\_attr\_inst\_id

from vzwnet.val\_attr\_name

where group\_name='Fault Management'

and attr\_name='Node Id')

) ,

xng\_dacs\_match as

(

select area,region,count(\*) dacs\_match

from xng\_reports.vzw\_vsm\_xng\_xref vvxx, disc\_nid

where

upper(vvxx.VSM\_DEVICE\_NAME) = upper(NODE\_ID)

and upper(vvxx.VSM\_CLASS) in ('DACS\_ALCATEL','DACSII','DACSII\_LA','ER\_DNX','TELLABS\_532L','TITAN')

group by area, region

) ,

vsm\_dacs as

(

select area,region, count(\*) dacs\_count

from xng\_reports.vzw\_vsm\_xng\_xref vvxx

where upper(vvxx.vsm\_class) in ('DACS\_ALCATEL','DACSII','DACSII\_LA','TELLABS\_532L','TITAN','ER\_DNX')

group by area,region

)

select vsm\_dacs.area, vsm\_dacs.region,

vsm\_dacs.dacs\_count AS equip\_count,

decode(xng\_dacs\_match.dacs\_match, 0, 0

, trunc((xng\_dacs\_match.dacs\_match)/(vsm\_dacs.dacs\_count)\*100, 2)) AS equip\_match,

xng\_dacs\_match.dacs\_match

from vsm\_dacs, xng\_dacs\_match

where vsm\_dacs.area=''||ARG\_IN\_AREA||''

and vsm\_dacs.region=''||ARG\_IN\_REGION||''

and vsm\_dacs.area = xng\_dacs\_match.area (+)

and vsm\_dacs.region = xng\_dacs\_match.region (+)

;

cursor getSwitchData(ARG\_IN\_AREA varchar2, ARG\_IN\_REGION varchar2) is

with

disc\_nid2 as (

select distinct attr\_value node\_id

from vzwnet.EQUIP\_ATTR\_SETTINGS

where VAL\_ATTR\_INST\_ID=(select val\_attr\_inst\_id

from vzwnet.val\_attr\_name

where group\_name='Fault Management'

and attr\_name='Node Id')

) ,

xng\_sw\_match as

(

select area,region,count(\*) switch\_match

from xng\_reports.vzw\_vsm\_xng\_xref vvxx,

disc\_nid2

where

upper(vvxx.VSM\_DEVICE\_NAME) = upper(NODE\_ID)

and upper(vvxx.VSM\_CLASS) in ('5ESS','MOTOROLA\_OMCR','MOTOROLA\_EMX','NTBSM','NTDMSMSC', 'NORTEL\_CNM')

group by area, region

) ,

vsm\_switch as

(

select area,region, count(\*) switch\_count

from xng\_reports.vzw\_vsm\_xng\_xref vvxx

where upper(vvxx.vsm\_class) in ('5ESS','MOTOROLA\_OMCR','MOTOROLA\_EMX','NTBSM','NTDMSMSC', 'NORTEL\_CNM')

group by area,region

)

select vsm\_switch.area, vsm\_switch.region,

vsm\_switch.switch\_count AS equip\_count,

decode(xng\_sw\_match.switch\_match, 0, 0,

trunc((xng\_sw\_match.switch\_match)/(vsm\_switch.switch\_count)\*100, 2)) AS equip\_match,

xng\_sw\_match.switch\_match

from vsm\_switch, xng\_sw\_match

where vsm\_switch.area=''||ARG\_IN\_AREA||''

and vsm\_switch.region=''||ARG\_IN\_REGION||''

and vsm\_switch.area = xng\_sw\_match.area (+)

and vsm\_switch.region = xng\_sw\_match.region (+)

;

l\_area\_equip\_count number:=0;

l\_area\_equip\_match\_pct number:=0;

l\_area\_equip\_match number:=0;

l\_region\_count number:=0;

l\_dacs\_count number:=0;

l\_dacs\_match\_pct number:=0;

l\_dacs\_match number:=0;

l\_switch\_count number:=0;

l\_switch\_match\_pct number:=0;

l\_switch\_match number:=0;

l\_region\_equip\_count number:=0;

l\_region\_equip\_match\_pct number:=0;

l\_region\_equip\_match number:=0;

l\_area varchar2(100);

l\_region varchar2(100);

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work es

set es.equip\_count= :l\_region\_equip\_count

, es.equip\_match\_pct = :l\_region\_equip\_match\_pct

where es.area= :area

and es.region= :region

and to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

--dbms\_output.put\_line(sql\_stmt);

for data in getRegions(area)

loop

l\_region\_count:=l\_region\_count +1;

--dbms\_output.put\_line(area||data.region);

open getSwitchData(area, data.region);

loop

fetch getSwitchData into l\_area, l\_region, l\_switch\_count, l\_switch\_match\_pct, l\_switch\_match;

exit when getSwitchData%notfound;

end loop;

close getSwitchData;

open getDacsData(area, data.region);

loop

fetch getDacsData into l\_area, l\_region, l\_dacs\_count, l\_dacs\_match\_pct, l\_dacs\_match;

exit when getDacsData%notfound;

end loop;

close getDacsData;

l\_region\_equip\_count:=l\_switch\_count+l\_dacs\_count;

l\_region\_equip\_match:=l\_switch\_match+l\_dacs\_match;

l\_region\_equip\_match\_pct:=round(((l\_region\_equip\_match/l\_region\_equip\_count)\*100),2);

l\_area\_equip\_count:=l\_area\_equip\_count+l\_region\_equip\_count;

l\_area\_equip\_match:=l\_area\_equip\_match+l\_region\_equip\_match;

EXECUTE IMMEDIATE sql\_stmt using l\_region\_equip\_count, l\_region\_equip\_match\_pct, area, data.region;

commit;

end loop;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateVSMData(): Populating Regional Stats'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateVSMData(): can''t populate VSM DACS/Switch numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populateVSMData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates data for '% Segments w/Path' col. in

\* ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateSegmentData(area in varchar2) is

cursor getRegions(ARG\_IN\_AREA varchar2) is

select drr.region from xng\_reports.domains\_regional\_reporting drr

where (drr.area) = ARG\_IN\_AREA

;

l\_area\_seg\_count number:=0;

l\_area\_seg\_w\_paths number:=0;

l\_area\_seg\_w\_paths\_pct number:=0;

l\_region\_count number:=0;

l\_region\_seg\_count number:=0;

l\_region\_seg\_w\_paths number:=0;

l\_region\_seg\_w\_paths\_pct number:=0;

l\_area varchar2(100);

l\_region varchar2(100);

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work es set es.seg\_with\_path\_pct= :l\_region\_seg\_w\_paths\_pct

where es.area= :area

and es.region= :region

and to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

for data in getRegions(area)

loop

l\_region\_count:=l\_region\_count +1;

l\_region:=data.region;

l\_region\_seg\_count:=getSegmentCount(l\_region);

l\_region\_seg\_w\_paths:=getSegmentWithPathCount(l\_region);

l\_region\_seg\_w\_paths\_pct:=round((l\_region\_seg\_w\_paths/l\_region\_seg\_count)\*100,2);

l\_area\_seg\_count:=l\_area\_seg\_count+l\_region\_seg\_count;

l\_area\_seg\_w\_paths:=l\_area\_seg\_w\_paths+l\_region\_seg\_w\_paths;

--dbms\_output.put\_line(sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt using l\_region\_seg\_w\_paths\_pct, area, data.region;

commit;

end loop;

l\_area\_seg\_w\_paths\_pct:=round((l\_area\_seg\_w\_paths/l\_area\_seg\_count\*100),2);

sql\_stmt:= 'update xng\_reports.executive\_summary\_work es set es.seg\_with\_path\_pct='||l\_area\_seg\_w\_paths\_pct||' where es.area='''

||area||''' and es.region=''Totals'''||' and to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

BEGIN

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateSegmentData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateSegmentData(): can''t update ''Xng Segments with Paths'' numbers for '|| area

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateSegmentData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateSegmentData(): can''t populate ''Xng Segments with Paths'' numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateSegmentData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc is called by populateSegmentData to help in calculating data

\* for '% Segments w/Path' col. in ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION getSegmentCount(p\_region IN varchar2

) RETURN NUMBER

IS

retval NUMBER := 0;

sqlStmt varchar2(32767);

BEGIN

sqlStmt := '

WITH

SEG\_PER\_DOMAIN as

(

select distinct cdm.circ\_inst\_id

from vzwnet.circ\_domain\_map cdm

join xng\_reports.DOMAINS\_REGIONAL\_REPORTING drr

on cdm.domain\_inst\_id = drr.domain\_inst\_id

where drr.region = :p\_region

)

,

SEGS as

(

select distinct ci.circ\_inst\_id

from vzwnet.circ\_inst ci

JOIN SEG\_PER\_DOMAIN ci\_ids

ON ci.circ\_inst\_id=ci\_ids.circ\_inst\_id

where ci.status = ''Live''

and upper(ci.bandwidth) !=''DS0''

)

select count(circ\_inst\_id)

from SEGS

';

execute immediate sqlStmt into retval using p\_region ;

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: getSegmentCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateSegmentData(): can''t get counts of segments for '||p\_region

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

raise;

END getSegmentCount;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc is called by populateSegmentData to help in calculating data

\* for '% Segments w/Path' col. in ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION getSegmentWithPathCount( p\_region IN varchar2

) RETURN NUMBER

IS

retval NUMBER := 0;

sqlStmt varchar2(32767);

BEGIN

sqlStmt := '

WITH

SEG\_PER\_DOMAIN as

(

select distinct cdm.circ\_inst\_id

from vzwnet.circ\_domain\_map cdm

join xng\_reports.DOMAINS\_REGIONAL\_REPORTING drr

on cdm.domain\_inst\_id = drr.domain\_inst\_id

where drr.region = :p\_region

)

,

SEGS as

(

select distinct ci.circ\_inst\_id

from vzwnet.circ\_inst ci

JOIN SEG\_PER\_DOMAIN ci\_ids

ON ci.circ\_inst\_id=ci\_ids.circ\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id IN (ci.CIRC\_PATH\_INST\_ID, ci.NEXT\_PATH\_INST\_ID)

where ci.status = ''Live''

and upper(ci.bandwidth) !=''DS0''

)

select count(circ\_inst\_id)

from SEGS

';

execute immediate sqlStmt into retval using p\_region;

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: getSegmentWithPathCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: getSegmentWithPathCount(): can''t get counts of segments with paths for '||p\_region

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END getSegmentWithPathCount;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc populates data for 'Compliance %' col. in ES. report

\* merely populates area/region # in here. Does not calc anything

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateXNGAuditData is

sqlStmt varchar2(32767);

begin

-- populates regional

-- NOT area totals in the exec\_summ table and Grand totals

sqlStmt:= 'update xng\_reports.executive\_summary\_work es

set es.OVERALL\_FAULT\_PCT = ( select round((xsa.overall\_fault\_pct)\*100,2) as overall\_fault\_pct

from xng\_reports.xngstd\_fault\_sum xsa

where to\_char(xsa.creation\_date, ''mm-dd-yyyy'')=(select to\_char(max(creation\_date),''mm-dd-yyyy'')

from xng\_reports.xngstd\_fault\_sum)

and xsa.region = es.region

)

';

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateXNGAuditData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateXNGAuditData(): can''t populate XngStdsCompliance numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateXNGAuditData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc updates '% BTP Circs Matched in XNG' col. in ES. report

\* merely populates regional # in here. Does not calc anything

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateBTPAuditData is

sql\_stmt varchar2(32000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work es

set es.ckt\_match\_pct= (

WITH

ROLLUPS as

(

select nvl(area, ''Grand'') area, nvl(region, ''Totals'') region

, round((sum(ckts\_matched)/sum(ckt\_count) \* 100), 2) as ckts\_matched\_pct\_num

from xng\_reports.btp\_xng\_audit\_summary

group by rollup(area, region)

)

select ckts\_matched\_pct\_num

from ROLLUPs r

where r.area = es.area

and r.region = es.region

)

/\*update xng\_reports.executive\_summary es

set es.ckt\_match\_pct= (select round((sum(ckts\_matched)/sum(ckt\_count) \* 100),2) as ckts\_matched\_pct

from xng\_reports.btp\_xng\_audit\_summary bxas

where es.area= bxas.area and es.region= bxas.region)\*/

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateBTPAuditData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateBTPAuditData(): can''t populate BTPXngAudit numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateBTPAuditData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates data for 'Cross Connect Compliance %' col. in ES. report

\* it calcs area and grand totals and puts them in the exec\_summ table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateCrossConAuditData is

sql\_stmt varchar2(32000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work es

set es.compliant\_pct= (

WITH

ROLLUPS as

(

select nvl(area, ''Grand'') area, nvl(region, ''Totals'') region

, round(sum(matched\_xconn\_count)/sum(total\_xconn\_count)\*100,2) as dacs\_xc\_compliant\_pct

from xng\_reports.xconnect\_summary

group by rollup(area, region)

)

select dacs\_xc\_compliant\_pct

from ROLLUPs r

where r.area = es.area

and r.region = es.region

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateCrossConAuditData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateCrossConAuditData(): can''t populate DACS cross-connect numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateCrossConAuditData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- this populates area and grand totals for only the following metrics

-- overall\_fault\_pct, seg\_with\_path\_pct, equip\_count, equip\_match\_pct,

-- ckt\_match\_pct, compliant\_pct

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure populateMetricTotals is

sqlStmt varchar2(32767);

BEGIN

BEGIN

-- Calc area totals by averaging regional pct in the area

sqlStmt := 'insert into xng\_reports.executive\_summary\_work columns (area, region, seg\_with\_path\_pct

, equip\_count, equip\_match\_pct

--, ckt\_match\_pct, compliant\_pct

, extract\_date)

WITH

AREA\_ROLLUP as

(

select area

,round(avg(SEG\_WITH\_PATH\_PCT), 2) SEG\_WITH\_PATH\_PCT -- seg having paths %

, sum(equip\_count) equip\_count -- vsm device cnt

,round(avg(EQUIP\_MATCH\_PCT), 2) EQUIP\_MATCH\_PCT -- vsm - Xng audit %

--,round(avg(CKT\_MATCH\_PCT), 2) CKT\_MATCH\_PCT -- btp - Xng audit %

--,round(avg(COMPLIANT\_PCT), 2) COMPLIANT\_PCT -- cross-connect audit %

from xng\_reports.executive\_summary\_work

-- "Totals" row will not be there in the table IF the process is run as a unit

-- but if it is being run stand alone, then it might exist.

-- this stmt is here, just to be safe

where Region <> ''Totals''

group by area

)

select nvl(area, ''Grand'') area, ''Totals'' region, seg\_with\_path\_pct

, equip\_count, equip\_match\_pct

--, ckt\_match\_pct, compliant\_pct

, trunc(sysdate)

from AREA\_ROLLUP

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateMetricTotals(): calc area totals for some metrics '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateMetricTotals(): can''t populate area totals for some metrics'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

BEGIN

-- populates area totals for Xng Standards from xngstd\_summ table

sqlStmt:= 'update xng\_reports.executive\_summary\_work es

set es.OVERALL\_FAULT\_PCT = ( select round(xsa.overall\_fault\_pct,2)

from xng\_reports.xngstd\_audit\_summary\_report xsa

where instr(xsa.area, '' '') > 0

and substr(xsa.area, 1, instr(xsa.area, '' '')-1) = es.area

and xsa.region=''Total Compliance''

and to\_char(xsa.creation\_date, ''mm-dd-yyyy'')=

(select to\_char(max(creation\_date),''mm-dd-yyyy'')

from xng\_reports.xngstd\_audit\_summary\_report)

)

where area not Like ''%Grand''

and es.region = ''Totals''

';

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateMetricTotals(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateMetricTotals(): can''t populate XngStdsCompliance area #'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

-- Calc grand totals by averaging regional totals

BEGIN

sqlStmt := 'insert into xng\_reports.executive\_summary\_work columns (area, region, overall\_fault\_pct, seg\_with\_path\_pct

, equip\_count, equip\_match\_pct, ckt\_match\_pct, compliant\_pct, extract\_date)

WITH

GRAND\_ROLLUP as

(

select round(avg(OVERALL\_FAULT\_PCT), 2) OVERALL\_FAULT\_PCT -- Xng Stds Data Audit %

,round(avg(SEG\_WITH\_PATH\_PCT), 2) SEG\_WITH\_PATH\_PCT -- seg having paths %

, sum(equip\_count) equip\_count -- vsm device cnt

,round(avg(EQUIP\_MATCH\_PCT), 2) EQUIP\_MATCH\_PCT -- vsm - Xng audit %

,round(avg(CKT\_MATCH\_PCT), 2) CKT\_MATCH\_PCT -- btp - Xng audit %

,round(avg(COMPLIANT\_PCT), 2) COMPLIANT\_PCT -- cross-connect audit %

from xng\_reports.executive\_summary\_work

-- Area "Totals" row WILL be there in the table

-- Grand total will be calculated based by avg regional pct and not area\_pct

where Region <> ''Totals''

)

select ''Grand'' area, ''Totals'' region, overall\_fault\_pct, seg\_with\_path\_pct

, equip\_count, equip\_match\_pct, ckt\_match\_pct, compliant\_pct, trunc(sysdate)

from GRAND\_ROLLUP

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateMetricTotals(): calc grand totals for some metrics '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateMetricTotals(): can''t populate grand totals totals for some metrics'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END populateMetricTotals;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc populates data for 'N2N Overall Backhaul Complinace' in

\* ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateOBSData is

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work es

set es.obs\_comp\_pct= (select OVERALL\_COMPLIANCE from N2N\_OVERALL\_COMPLIANCE\_SUMM n2n

where decode(n2n.area, ''zEnterprise'', ''Grand'', area) = es.area

and nvl(n2n.region, ''Totals'') = es.region)

where to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateOBSData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateOBSData(): can''t populate OBS #'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateOBSData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates data for 'Overall Compliance' col. in

\* ES report, which is the avg. of the percentages displayed

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateOverallPct is

sqlStmt varchar(32767);

BEGIN

begin

sqlStmt := 'update executive\_summary\_work es

set es.OVERALL\_COM\_PCT = round((

nvl(overall\_fault\_pct,0) +

nvl(seg\_with\_path\_pct,0) +

nvl(equip\_match\_pct,0) +

nvl(ckt\_match\_pct,0) +

nvl(compliant\_pct,0)

)/5,2)

where extract\_date = trunc(sysdate)

and area = ''NNO''

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateOverallPct(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateOverallPct(): can''t calc overall for NNO%'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end;

begin

sqlStmt := 'update executive\_summary\_work es

set es.OVERALL\_COM\_PCT = round((

nvl(overall\_fault\_pct,0) +

nvl(seg\_with\_path\_pct,0) +

nvl(equip\_match\_pct,0) +

nvl(ckt\_match\_pct,0) +

nvl(compliant\_pct,0) +

nvl(obs\_comp\_pct,0)

)/6,2)

where extract\_date = trunc(sysdate)

and area <> ''NNO''

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateOverallPct(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateOverallPct(): can''t calc overall % for non NNO'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

end populateOverallPct;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc get the prev. reported overall Comp. from executive\_summary\_reported

\* table for 'Previous Reported Overall Compliance' col in ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populatePrevGrandTotalsRow is

sql\_stmt varchar2(2000);

begin

-- populate values of last months grand totals

sql\_stmt:= 'insert into xng\_reports.executive\_summary\_work es (area, region, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

)

select ''Previous Grand'', ''Totals''

, trunc(sysdate), esr.overall\_fault\_pct

, esr.seg\_with\_path\_pct, esr.equip\_count, esr.equip\_match\_pct

, esr.ckt\_match\_pct, esr.compliant\_pct, esr.obs\_comp\_pct, esr.overall\_com\_pct

from xng\_reports.executive\_summary\_reported esr

where esr.extract\_date=(select max(extract\_date)

from xng\_reports.executive\_summary\_reported)

and esr.area = ''Grand''

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevGrandTotals: Populating prev grand totals'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populatePrevGrandTotalsRow(): can''t populate prev months metric totals'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populatePrevGrandTotalsRow;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc get the prev. reported overall Comp. from executive\_summary\_reported

\* table for 'Previous Reported Overall Compliance' col in ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateDeltasRow is

sql\_stmt varchar2(2000);

begin

-- populate values of last months grand totals

sql\_stmt:= 'insert into xng\_reports.executive\_summary\_work es (area, region, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

)

select ''Delta'', ''''

, trunc(sysdate)

, (nvl(es.overall\_fault\_pct, 0)-nvl(esr.overall\_fault\_pct, 0))

, (nvl(es.seg\_with\_path\_pct, 0)-nvl(esr.seg\_with\_path\_pct, 0))

, (nvl(es.equip\_count, 0)-nvl(esr.equip\_count, 0))

, (nvl(es.equip\_match\_pct, 0)-nvl(esr.equip\_match\_pct, 0))

, (nvl(es.ckt\_match\_pct, 0)-nvl(esr.ckt\_match\_pct, 0))

, (nvl(es.compliant\_pct, 0)-nvl(esr.compliant\_pct, 0))

, (nvl(es.obs\_comp\_pct, 0)-nvl(esr.obs\_comp\_pct, 0))

, (nvl(es.overall\_com\_pct, 0)-nvl(esr.overall\_com\_pct, 0))

from xng\_reports.executive\_summary\_work es

join xng\_reports.executive\_summary\_reported esr

on es.area=esr.area

and esr.region=es.region

where esr.extract\_date=(select max(extract\_date) from xng\_reports.executive\_summary\_reported)

and to\_char(es.extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')

and esr.area = ''Grand''

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevGrandTotals: Populating prev grand totals'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populatePrevGrandTotalsRow(): can''t populate metric deltas between current months and prev month'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populateDeltasRow;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates

\* the diff. between the current overall comp and prev. reported overall comp.

\* It also gets the prev. reported grand totals for 'Previous Reported Grand Totals'

\* row and changes calculates deltas for the 'Change' row.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Procedure populatePrevAndDeltasCol is

cursor prevAndDelta is

select es.area, es.region, esr.overall\_com\_pct, (nvl(es.overall\_com\_pct, 0)-nvl(esr.overall\_com\_pct, 0)) deltas

from xng\_reports.executive\_summary es

join xng\_reports.executive\_summary\_reported esr

on es.area=esr.area

and esr.region=es.region

where esr.extract\_date=(select max(extract\_date) from xng\_reports.executive\_summary\_reported)

and to\_char(es.extract\_date,'MM-DD-YYYY')=to\_char(sysdate, 'MM-DD-YYYY')

--and es.area <> 'Previous Grand'

;

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work es

set es.delta = :deltas

, es.PREV\_RPT\_OVERALL\_COM\_PCT = :overall\_com\_pct

where es.area = :area

and es.region = :region

';

-- populate col prev and deltas

for rec in prevAndDelta

Loop

BEGIN

--dbms\_output.put\_line(rec.area||','|| rec.region||','|| rec.deltas||','|| rec.overall\_com\_pct);

EXECUTE IMMEDIATE sql\_stmt using rec.deltas, rec.overall\_com\_pct, rec.area, rec.region;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevAndDeltasCol: Populating col prev/delta'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

END;

end loop;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevAndDeltasCol: Populating prev grand totals'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populatePrevAndDeltasCol(): can''t populate area and regional deltas between current month and prev month'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populatePrevAndDeltasCol;

\*/

Procedure populatePrevAndDeltasCol is

cursor prevAndDelta is

select es.area, es.region, esr.overall\_com\_pct, (nvl(es.overall\_com\_pct, 0)-nvl(esr.overall\_com\_pct, 0)) deltas

from xng\_reports.executive\_summary\_work es

join xng\_reports.executive\_summary\_reported esr

on es.area=esr.area

and esr.region=es.region

where esr.extract\_date=(select max(extract\_date) from xng\_reports.executive\_summary\_reported)

--and to\_char(es.extract\_date,'MM-DD-YYYY')=to\_char(sysdate, 'MM-DD-YYYY')

--and es.area <> 'Previous Grand'

;

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work es

set es.delta = :deltas

, es.PREV\_RPT\_OVERALL\_COM\_PCT = :overall\_com\_pct

where es.area = :area

and es.region = :region

';

-- populate col prev and deltas

for rec in prevAndDelta

Loop

BEGIN

--dbms\_output.put\_line(rec.area||','|| rec.region||','|| rec.deltas||','|| rec.overall\_com\_pct);

EXECUTE IMMEDIATE sql\_stmt using rec.deltas, rec.overall\_com\_pct, rec.area, rec.region;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevAndDeltasCol: Populating col prev/delta'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

END;

end loop;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevAndDeltasCol: Populating prev grand totals'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populatePrevAndDeltasCol(): can''t populate area and regional deltas between current month and prev month'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populatePrevAndDeltasCol;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* in\_progress data is copied from WORK to reported.

\* before copying the reported data is copied to a bak table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure copyFromInProgressToReported is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from exec\_summ to exec\_summ\_rptd

BEGIN

sql\_stmt:='truncate table xng\_reports.executive\_summary\_reported\_bak';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

-- do named inserts. select \* causes issues when columns are added

sql\_stmt:='insert into executive\_summary\_reported\_bak columns (

area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

)

select area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

from executive\_summary\_reported';

EXECUTE IMMEDIATE sql\_stmt;

commit;

BEGIN

sql\_stmt:='truncate table xng\_reports.executive\_summary\_reported';

EXECUTE IMMEDIATE sql\_stmt;

commit;

BEGIN

sql\_stmt:='insert into executive\_summary\_reported columns (

area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

)

select area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

from executive\_summary

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting executive\_summary\_rptd '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert in exec\_summ\_rptd '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Truncating executive\_summary '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc exec\_summ table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting executive\_summary\_reported\_bak '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into exec\_summ\_rptd table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating executive\_summary\_reported\_bak '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc exec\_summ\_rptd\_bak table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END copyFromInProgressToReported;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* data is copied from WORK to in\_progress.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure copyFromWorkToInProgress is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from exec\_summ to exec\_summ\_rptd

BEGIN

sql\_stmt:='truncate table xng\_reports.executive\_summary';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

-- do named inserts. select \* causes issues when columns are added

sql\_stmt:='insert into executive\_summary columns (

area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

)

select area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

from executive\_summary\_work';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting executive\_summary '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into exec\_summ table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating executive\_summary '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc exec\_summ table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END copyFromWorkToInProgress;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate a backup of the previous summary data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure createPreviousBackup is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from EXECUTIVE\_SUMMARY\_PREVIOUS to EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK ';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK columns (

area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select area, region

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_previous';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryReported(): Inserting EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInReported(): cant insert into EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryReported(): truncating EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryReported(): cant trunc EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END createPreviousBackup;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* data is copied from Reported to Previous

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure copyFromReportedToPrevious is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from exec\_summ\_rptd to exec\_summ\_previous

BEGIN

-- do named inserts. select \* causes issues when columns are added

sql\_stmt:=' insert into xng\_reports.executive\_summary\_previous columns (

area, region

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, extract\_date

, compliant\_pct

, overall\_com\_pct

, prev\_rpt\_overall\_com\_pct

, delta

, obs\_comp\_pct

)

select area, region

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, extract\_date

, compliant\_pct

, overall\_com\_pct

, prev\_rpt\_overall\_com\_pct

, delta

, obs\_comp\_pct

from executive\_summary\_reported ';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryReported(): Inserting executive\_summary\_reported into executive\_summary\_previous '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryReported(): cannot insert into executive\_summary\_previous table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END copyFromReportedToPrevious;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Delete Old Previous

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure deleteOldPrevious is

sql\_stmt varchar(32767);

BEGIN

-- Previous summaries must be kept for 12 months - this deletes any older than that

BEGIN

-- delete previous summaries over 13 months old

sql\_stmt:=' delete xng\_reports.executive\_summary\_previous

where extract\_date < sysdate - 395';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryReported(): Deleting executive\_summary\_previous over 13 months old '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryReported(): cannot delete from executive\_summary\_previous table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END deleteOldPrevious;

END;

/

--------------------------------------------------------

-- DDL for Package Body EXECUTIVE\_SUMMARY\_PKG\_NEW

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."EXECUTIVE\_SUMMARY\_PKG\_NEW" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate exec\_summ\_rpt

\* first in\_progress will be generated and then moved to rptd

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure executiveSummaryReported is

BEGIN

processName := 'EXECUTIVE\_SUMMARY\_RPTD\_NEW';

BEGIN

-- Tell watchDdog the process has started

watchdog.updateprocessstart (processName);

BEGIN

-- create a snapshot of previous month data

createReportedSnapshot ();

-- copy current reported data to previous

copyFromReportedToPrevious ();

-- previous data over 13 months old

deleteOldPrevious ();

-- copy current previous data to previous bak

createPreviousBackup ();

-- Generate Executive Suumary

genExecutiveSummary();

-- copy generated in\_progress to reported

copyFromInProgressToReported ();

-- mark successful completion of the task

watchdog.updateprocessend (processName, 'STATUS\_SUCCESS', 'Y');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

-- mark failed completion of the task

watchdog.updateprocessend (processName, 'STATUS\_FAILURE', 'N');

END;

END;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate exec\_summ

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure executiveSummaryInProgress is

BEGIN

processName := 'EXECUTIVE\_SUMMARY\_NEW';

-- Tell watchDdog the process has started

BEGIN

watchdog.updateprocessstart (processName);

BEGIN

genExecutiveSummary();

-- tell watchDog process completed successfully

watchdog.updateprocessend (processName, 'STATUS\_SUCCESS', 'Y');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.updateprocessend (processName, 'STATUS\_FAILURE', 'N');

END;

END;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate a snapshot of the previous month data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure createSnapshot is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from EXECUTIVE\_SUMMARY\_NEW to EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW columns (

territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc EXECUTIVE\_SUMMARY\_SNAPSHOT\_NEW table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END createSnapshot;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate a snapshot of the previous month data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure createReportedSnapshot is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from EXECUTIVE\_SUMMARY\_REPORTED to EXECUTIVE\_SUMMARYRPTD\_SNAPSHOT

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMRPTD\_SNAP\_NEW';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMRPTD\_SNAP\_NEW columns (

territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_reported\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting EXECUTIVE\_SUMRPTD\_SNAPSHOT\_NEW '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into EXECUTIVE\_SUMRPTD\_SNAPSHOT\_NEW table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating EXECUTIVE\_SUMRPTD\_SNAPSHOT\_NEW '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc EXECUTIVE\_SUMRPTD\_SNAPSHOT\_NEW table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

-- Copy data from EXECUTIVE\_SUMMARY\_REPORTED\_BAK to EXECUTIVE\_SUMMRPTDBAK\_SNAPSHOT

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMRPTDBAK\_SNAP\_NEW';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMRPTDBAK\_SNAP\_NEW columns (

territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_rptd\_bak\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting EXECUTIVE\_SUMRPTDBAK\_SNAP\_NEW '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into EXECUTIVE\_SUMRPTDBAK\_SNAP\_NEW table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating EXECUTIVE\_SUMRPTDBAK\_SNAP\_NEW '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc EXECUTIVE\_SUMRPTDBAK\_SNAP\_NEW table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

-- Copy data from EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK to EXECUTIVE\_SUMMPREVBAK\_SNAPSHOT

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMPREVBAK\_SNAP\_NEW ';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMPREVBAK\_SNAP\_NEW columns (

territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_prev\_bak\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting EXECUTIVE\_SUMPREVBAK\_SNAP\_NEW '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into EXECUTIVE\_SUMPREVBAK\_SNAP\_NEW table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating EXECUTIVE\_SUMPREVBAK\_SNAP\_NEW '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc EXECUTIVE\_SUMPREVBAK\_SNAP\_NEW table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END createReportedSnapshot;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* procedure that populates exec\_summ table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure genExecutiveSummary is

-- get distinct areas

cursor getTerritory is

select distinct territory

from xng\_reports.domains\_regional\_reporting

order by territory

;

sql\_stmt varchar2(32000);

begin

-- backup executive\_summary table

createSnapshot();

sql\_stmt:='truncate table xng\_reports.executive\_summary\_work\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

--populate \_new from orig and new domains\_regional\_reporting

--after sub reports are changed to new org, add these steps and remove

-- columns from here.

sql\_stmt:='insert into xng\_reports.executive\_summary\_work\_new

(territory, market\_territory, sub\_market, overall\_fault\_pct,

seg\_with\_path\_pct, equip\_count, equip\_match\_pct, ckt\_match\_pct,

extract\_date, compliant\_pct, overall\_com\_pct,

prev\_rpt\_overall\_com\_pct, delta, obs\_comp\_pct)

with new\_dom as (select distinct territory, market\_territory, sub\_market, area, region

from xng\_reports.domains\_regional\_reporting\_new)

select nd.territory, nd.market\_territory, nd.sub\_market, esw.overall\_fault\_pct, esw.seg\_with\_path\_pct,

esw.equip\_count, esw.equip\_match\_pct, esw.ckt\_match\_pct, trunc(sysdate) as extract\_date, esw.compliant\_pct,

esw.overall\_com\_pct, esw.prev\_rpt\_overall\_com\_pct, esw.delta, esw.obs\_comp\_pct

from xng\_reports.executive\_summary esw, new\_dom nd

where esw.area = nd.area

and esw.region = nd.region';

EXECUTE IMMEDIATE sql\_stmt;

commit;

--populate \_new totals

sql\_stmt:='insert into xng\_reports.executive\_summary\_work\_new

(territory, market\_territory, sub\_market, overall\_fault\_pct,

seg\_with\_path\_pct, equip\_count, equip\_match\_pct,

ckt\_match\_pct, compliant\_pct, overall\_com\_pct,

prev\_rpt\_overall\_com\_pct, delta, obs\_comp\_pct, extract\_date)

with sub\_mkt\_rollup as

(

select territory, market\_territory,

round(avg(overall\_fault\_pct), 2) overall\_fault\_pct,

round(avg(seg\_with\_path\_pct), 2) seg\_with\_path\_pct,

sum(equip\_count) equip\_count,

round(avg(equip\_match\_pct), 2) equip\_match\_pct,

round(avg(ckt\_match\_pct), 2) ckt\_match\_pct,

round(avg(compliant\_pct), 2) compliant\_pct,

round(avg(overall\_com\_pct), 2) overall\_com\_pct,

round(avg(prev\_rpt\_overall\_com\_pct), 2) prev\_rpt\_overall\_com\_pct,

round(avg(delta), 2) delta,

round(avg(obs\_comp\_pct), 2) obs\_comp\_pct

from xng\_reports.executive\_summary\_work\_new

where sub\_market <> ''Totals''

group by territory, market\_territory

),

terr\_rollup as

(

select territory,

round(avg(overall\_fault\_pct), 2) overall\_fault\_pct,

round(avg(seg\_with\_path\_pct), 2) seg\_with\_path\_pct,

sum(equip\_count) equip\_count,

round(avg(equip\_match\_pct), 2) equip\_match\_pct,

round(avg(ckt\_match\_pct), 2) ckt\_match\_pct,

round(avg(compliant\_pct), 2) compliant\_pct,

round(avg(overall\_com\_pct), 2) overall\_com\_pct,

round(avg(prev\_rpt\_overall\_com\_pct), 2) prev\_rpt\_overall\_com\_pct,

round(avg(delta), 2) delta,

round(avg(obs\_comp\_pct), 2) obs\_comp\_pct

from xng\_reports.executive\_summary\_work\_new

where sub\_market <> ''Totals''

group by territory

),

grand\_rollup as

(

select ''Grand'' territory, ''TotMkt'' market\_territory, ''Totals'' sub\_market,

round(avg(overall\_fault\_pct), 2) overall\_fault\_pct,

round(avg(seg\_with\_path\_pct), 2) seg\_with\_path\_pct,

sum(equip\_count) equip\_count,

round(avg(equip\_match\_pct), 2) equip\_match\_pct,

round(avg(ckt\_match\_pct), 2) ckt\_match\_pct,

round(avg(compliant\_pct), 2) compliant\_pct,

round(avg(overall\_com\_pct), 2) overall\_com\_pct,

round(avg(prev\_rpt\_overall\_com\_pct), 2) prev\_rpt\_overall\_com\_pct,

round(avg(delta), 2) delta,

round(avg(obs\_comp\_pct), 2) obs\_comp\_pct

from xng\_reports.executive\_summary\_work\_new

where sub\_market <> ''Totals''

),

prev\_rollup as

(select ''Previous Grand'' territory, ''TotMkt'' market\_territory, ''Totals'' sub\_market

, esr.overall\_fault\_pct, esr.seg\_with\_path\_pct, esr.equip\_count, esr.equip\_match\_pct

, esr.ckt\_match\_pct, esr.compliant\_pct, esr.obs\_comp\_pct, esr.overall\_com\_pct

, esr.prev\_rpt\_overall\_com\_pct, esr.delta

from xng\_reports.executive\_summary\_reported\_new esr

where esr.extract\_date=(select max(extract\_date)

from xng\_reports.executive\_summary\_reported\_new)

and esr.territory = ''Grand'')

select territory, market\_territory, ''Totals'' sub\_market, overall\_fault\_pct,

seg\_with\_path\_pct, equip\_count, equip\_match\_pct, ckt\_match\_pct,

compliant\_pct, overall\_com\_pct, prev\_rpt\_overall\_com\_pct,

delta, obs\_comp\_pct, trunc(sysdate)

from sub\_mkt\_rollup

union

select territory, ''TotMkt'' market\_territory, ''Totals'' sub\_market, overall\_fault\_pct,

seg\_with\_path\_pct, equip\_count, equip\_match\_pct, ckt\_match\_pct,

compliant\_pct, overall\_com\_pct, prev\_rpt\_overall\_com\_pct, delta, obs\_comp\_pct, trunc(sysdate)

from terr\_rollup

union

select ''Grand'' territory, ''TotMkt'' market\_territory, ''Totals'' sub\_market, overall\_fault\_pct,

seg\_with\_path\_pct, equip\_count, equip\_match\_pct, ckt\_match\_pct,

compliant\_pct, overall\_com\_pct, prev\_rpt\_overall\_com\_pct, delta, obs\_comp\_pct, trunc(sysdate)

from grand\_rollup';

EXECUTE IMMEDIATE sql\_stmt;

commit;

--after sub reports are changed to new org, add these steps and remove columns

--from above. Code below has been started but not tested

/\*

-- insert all area region rows in the exec summary table.

-- later metrics for area/region will be populated

BEGIN

sql\_Stmt := 'insert into xng\_reports.executive\_summary\_work\_new columns(territory, market, sub\_market, territory\_market\_sub, area, region, extract\_date)

select distinct territory, market, sub\_market, territory\_market\_sub, area, region, to\_date(sysdate)

from xng\_reports.domains\_regional\_reporting

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- populate regional metrics (col) for the following in the exec\_summ table

-- these have to be calculated since these are not persisted anywhere

for terr\_row in getTerritory

loop

-- calc VSM equip data

populateVSMData(terr\_row.territory);

-- calc Segments with Path data

populateSegmentData(terr\_row.territory);

end loop;

-- this only stores regional and area totals and not grand totals

-- get data from xngstds audit,

populateXNGAuditData();

-- create aggregate rollup of all entered metrics so far and store in exec\_summ

populateMetricTotals();

-- get data from btp audit. Calc area and grand totals as well

populateBTPAuditData();

-- get data from cross connect audit

populateCrossConAuditData;

-- processes below already compute and put stats in table with grand totals

-- just populate the columns from these tables

-- add % OBS for the enterprise. This has region/area/enterprise #.

-- so not calculating them again.

populateOBSData();

\*/

-- Now all metric col are fully populated.

-- calc Overall Compliance % for the region/area/enterprise

populateOverallPct();

-- insert prev months metric grand total as a row

populatePrevGrandTotalsRow();

-- for each metric calc deltas between grand total of the metric of current month vs prev month

-- insert these as a row

populateDeltasRow();

-- populate prev month compliance % in a col and

-- delta between current month's total\_compliance and prev month's total compliance

populatePrevAndDeltasCol();

--Now copy from Work to InProgress

copyFromWorkToInProgress ();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end genExecutiveSummary;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates data for '% Switch/DACS Audited in XNG vs VSM' and

\* 'Equip Count' cols. in ES. report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateVSMData(territory in varchar2) is

cursor getMarkets(ARG\_IN\_TERR varchar2) is

select distinct market\_territory, sub\_market from

xng\_reports.domains\_regional\_reporting drr

where drr.territory = ''||ARG\_IN\_TERR||'' ;

cursor getDacsData(ARG\_IN\_TERR varchar2, ARG\_IN\_MKT\_TERR varchar2, ARG\_IN\_SUB\_MKT varchar2) is

with

disc\_nid as

(

select distinct attr\_value node\_id

from vzwnet.EQUIP\_ATTR\_SETTINGS

where VAL\_ATTR\_INST\_ID=(select val\_attr\_inst\_id

from vzwnet.val\_attr\_name

where group\_name='Fault Management'

and attr\_name='Node Id')

) ,

--get rid of this when xng\_reports.vzw\_vsm\_xng\_xref chgd to new dom

data\_new\_dom as

(

select dlr.territory, dlr.market\_territory, dlr.sub\_market, vvxx.\*

from xng\_reports.vzw\_vsm\_xng\_xref vvxx,

xng\_reports.domains\_regional\_reporting dlr

where vvxx.area = dlr.area

and vvxx.region = dlr.region

),

xng\_dacs\_match as

(

select vvxx.territory, vvxx.market\_territory, vvxx.sub\_market, count(\*) dacs\_match

-- use this when xng\_reports.vzw\_vsm\_xng\_xref chgd to new dom

-- from xng\_reports.vzw\_vsm\_xng\_xref vvxx, disc\_nid

from data\_new\_dom vvxx, disc\_nid

where

upper(vvxx.VSM\_DEVICE\_NAME) = upper(NODE\_ID)

and upper(vvxx.VSM\_CLASS) in ('DACS\_ALCATEL','DACSII','DACSII\_LA','ER\_DNX','TELLABS\_532L','TITAN')

group by vvxx.territory, vvxx.market\_territory, vvxx.sub\_market

),

vsm\_dacs as

(

select vvxx.territory, vvxx.market\_territory, vvxx.sub\_market, count(\*) dacs\_count

from data\_new\_dom vvxx

where upper(vvxx.vsm\_class) in ('DACS\_ALCATEL','DACSII','DACSII\_LA','TELLABS\_532L','TITAN','ER\_DNX')

group by vvxx.territory, vvxx.market\_territory, vvxx.sub\_market

)

select vsm\_dacs.territory, vsm\_dacs.market\_territory, vsm\_dacs.sub\_market,

vsm\_dacs.dacs\_count AS equip\_count,

decode(xng\_dacs\_match.dacs\_match, 0, 0

, trunc((xng\_dacs\_match.dacs\_match)/(vsm\_dacs.dacs\_count)\*100, 2)) AS equip\_match,

xng\_dacs\_match.dacs\_match

from vsm\_dacs, xng\_dacs\_match

where vsm\_dacs.territory = xng\_dacs\_match.territory (+)

and vsm\_dacs.market\_territory = xng\_dacs\_match.market\_territory (+)

and vsm\_dacs.sub\_market = xng\_dacs\_match.sub\_market (+)

and vsm\_dacs.territory =''||ARG\_IN\_TERR||''

and vsm\_dacs.market\_territory =''||ARG\_IN\_MKT\_TERR||''

and vsm\_dacs.sub\_market =''||ARG\_IN\_SUB\_MKT||''

;

cursor getSwitchData(ARG\_IN\_TERR varchar2, ARG\_IN\_MKT\_TERR varchar2, ARG\_IN\_SUB\_MKT varchar2) is

with

disc\_nid2 as (

select distinct attr\_value node\_id

from vzwnet.EQUIP\_ATTR\_SETTINGS

where VAL\_ATTR\_INST\_ID=(select val\_attr\_inst\_id

from vzwnet.val\_attr\_name

where group\_name='Fault Management'

and attr\_name='Node Id')

) ,

--get rid of this when xng\_reports.vzw\_vsm\_xng\_xref chgd to new dom

data\_new\_dom as

(

select dlr.territory, dlr.market\_territory, dlr.sub\_market, vvxx.\*

from xng\_reports.vzw\_vsm\_xng\_xref vvxx,

xng\_reports.domains\_regional\_reporting dlr

where vvxx.area = dlr.area

and vvxx.region = dlr.region

),

xng\_sw\_match as

(

select vvxx.territory, vvxx.market\_territory, vvxx.sub\_market, count(\*) switch\_match

--use this when xng\_reports.vzw\_vsm\_xng\_xref chgd to new dom

--from xng\_reports.vzw\_vsm\_xng\_xref vvxx, disc\_nid2

from data\_new\_dom vvxx, disc\_nid2

where

upper(vvxx.VSM\_DEVICE\_NAME) = upper(NODE\_ID)

and upper(vvxx.VSM\_CLASS) in ('5ESS','MOTOROLA\_OMCR','MOTOROLA\_EMX','NTBSM','NTDMSMSC', 'NORTEL\_CNM')

group by vvxx.territory, vvxx.market\_territory, vvxx.sub\_market

) ,

vsm\_switch as

(

select vvxx.territory, vvxx.market\_territory, vvxx.sub\_market, count(\*) switch\_count

from data\_new\_dom vvxx

where upper(vvxx.vsm\_class) in ('5ESS','MOTOROLA\_OMCR','MOTOROLA\_EMX','NTBSM','NTDMSMSC', 'NORTEL\_CNM')

group by vvxx.territory, vvxx.market\_territory, vvxx.sub\_market

)

select vsm\_switch.territory, vsm\_switch.market\_territory, vsm\_switch.sub\_market,

vsm\_switch.switch\_count AS equip\_count,

decode(xng\_sw\_match.switch\_match, 0, 0,

trunc((xng\_sw\_match.switch\_match)/(vsm\_switch.switch\_count)\*100, 2)) AS equip\_match,

xng\_sw\_match.switch\_match

from vsm\_switch, xng\_sw\_match

where vsm\_switch.territory = xng\_sw\_match.territory (+)

and vsm\_switch.market\_territory = xng\_sw\_match.market\_territory (+)

and vsm\_switch.sub\_market = xng\_sw\_match.sub\_market (+)

and vsm\_switch.territory =''||ARG\_IN\_TERR||''

and vsm\_switch.market\_territory =''||ARG\_IN\_MKT\_TERR||''

and vsm\_switch.sub\_market =''||ARG\_IN\_SUB\_MKT||''

;

l\_terr\_equip\_count number:=0;

l\_terr\_equip\_match\_pct number:=0;

l\_terr\_equip\_match number:=0;

l\_sub\_mkt\_count number:=0;

l\_dacs\_count number:=0;

l\_dacs\_match\_pct number:=0;

l\_dacs\_match number:=0;

l\_switch\_count number:=0;

l\_switch\_match\_pct number:=0;

l\_switch\_match number:=0;

l\_sub\_mkt\_equip\_count number:=0;

l\_sub\_mkt\_equip\_match\_pct number:=0;

l\_sub\_mkt\_equip\_match number:=0;

l\_terr varchar2(100);

l\_mkt\_terr varchar2(50);

l\_sub\_mkt varchar2(100);

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work\_new es

set es.equip\_count= :l\_sub\_mkt\_equip\_count

, es.equip\_match\_pct = :l\_sub\_mkt\_equip\_match\_pct

where es.territory = :territory

and es.market\_territory = :market\_territory

and es.sub\_market = :sub\_market

and to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

--dbms\_output.put\_line(sql\_stmt);

for data in getMarkets(territory)

loop

l\_sub\_mkt\_count:=l\_sub\_mkt\_count +1;

--dbms\_output.put\_line(territory||data.region);

open getSwitchData(territory, data.market\_territory, data.sub\_market);

loop

fetch getSwitchData into l\_terr, l\_mkt\_terr, l\_sub\_mkt, l\_switch\_count, l\_switch\_match\_pct, l\_switch\_match;

exit when getSwitchData%notfound;

end loop;

close getSwitchData;

open getDacsData(territory, data.market\_territory, data.sub\_market);

loop

fetch getDacsData into l\_terr, l\_mkt\_terr, l\_sub\_mkt, l\_dacs\_count, l\_dacs\_match\_pct, l\_dacs\_match;

exit when getDacsData%notfound;

end loop;

close getDacsData;

l\_sub\_mkt\_equip\_count:=l\_switch\_count+l\_dacs\_count;

l\_sub\_mkt\_equip\_match:=l\_switch\_match+l\_dacs\_match;

l\_sub\_mkt\_equip\_match\_pct:=round(((l\_sub\_mkt\_equip\_match/l\_sub\_mkt\_equip\_count)\*100),2);

EXECUTE IMMEDIATE sql\_stmt using l\_sub\_mkt\_equip\_count, l\_sub\_mkt\_equip\_match\_pct, territory, data.market\_territory, data.sub\_market;

commit;

end loop;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateVSMData(): Populating Regional Stats'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateVSMData(): can''t populate VSM DACS/Switch numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populateVSMData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates data for '% Segments w/Path' col. in

\* ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateSegmentData(territory in varchar2) is

cursor getMarkets(ARG\_IN\_TERR varchar2) is

select distinct market\_territory, sub\_market from

xng\_reports.domains\_regional\_reporting drr

where drr.territory = ''||ARG\_IN\_TERR||'' ;

l\_terr\_seg\_count number:=0;

l\_terr\_seg\_w\_paths number:=0;

l\_terr\_seg\_w\_paths\_pct number:=0;

l\_sub\_mkt\_count number:=0;

l\_sub\_mkt\_seg\_count number:=0;

l\_sub\_mkt\_seg\_w\_paths number:=0;

l\_sub\_mkt\_seg\_w\_paths\_pct number:=0;

l\_terr varchar2(100);

l\_mkt\_terr varchar2(50);

l\_sub\_mkt varchar2(100);

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work\_new es set es.seg\_with\_path\_pct= :l\_region\_seg\_w\_paths\_pct

where es.territory = :territory

and es.market\_territory = :market\_territory

and es.sub\_market = :sub\_market

and to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

for data in getMarkets(territory)

loop

l\_sub\_mkt\_count:=l\_sub\_mkt\_count +1;

l\_sub\_mkt:=data.sub\_market;

l\_sub\_mkt\_seg\_count:=getSegmentCount(l\_sub\_mkt);

l\_sub\_mkt\_seg\_w\_paths:=getSegmentWithPathCount(l\_sub\_mkt);

l\_sub\_mkt\_seg\_w\_paths\_pct:=round((l\_sub\_mkt\_seg\_w\_paths/l\_sub\_mkt\_seg\_count)\*100,2);

--dbms\_output.put\_line(sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt using l\_sub\_mkt\_seg\_w\_paths\_pct, territory, data.market\_territory, data.sub\_market;

commit;

end loop;

--l\_terr\_seg\_w\_paths\_pct:=round((l\_terr\_seg\_w\_paths/l\_terr\_seg\_count\*100),2);

-- sql\_stmt:= 'update xng\_reports.executive\_summary\_work\_new es set es.seg\_with\_path\_pct='||l\_area\_seg\_w\_paths\_pct||' where es.area='''

-- ||area||''' and es.region=''Totals'''||' and to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

BEGIN

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateSegmentData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateSegmentData(): can''t update ''Xng Segments with Paths'' numbers for '|| territory

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateSegmentData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateSegmentData(): can''t populate ''Xng Segments with Paths'' numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateSegmentData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc is called by populateSegmentData to help in calculating data

\* for '% Segments w/Path' col. in ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION getSegmentCount(p\_sub\_mkt IN varchar2

) RETURN NUMBER

IS

retval NUMBER := 0;

sqlStmt varchar2(32767);

BEGIN

sqlStmt := '

WITH

SEG\_PER\_DOMAIN as

(

select distinct cdm.circ\_inst\_id

from vzwnet.circ\_domain\_map cdm

join xng\_reports.DOMAINS\_REGIONAL\_REPORTING\_NEW drr

on cdm.domain\_inst\_id = drr.domain\_inst\_id

where drr.sub\_market = :p\_sub\_mkt

)

,

SEGS as

(

select distinct ci.circ\_inst\_id

from vzwnet.circ\_inst ci

JOIN SEG\_PER\_DOMAIN ci\_ids

ON ci.circ\_inst\_id=ci\_ids.circ\_inst\_id

where ci.status = ''Live''

and upper(ci.bandwidth) !=''DS0''

)

select count(circ\_inst\_id)

from SEGS

';

execute immediate sqlStmt into retval using p\_sub\_mkt ;

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: getSegmentCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateSegmentData(): can''t get counts of segments for '||p\_sub\_mkt

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

raise;

END getSegmentCount;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc is called by populateSegmentData to help in calculating data

\* for '% Segments w/Path' col. in ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION getSegmentWithPathCount( p\_sub\_mkt IN varchar2

) RETURN NUMBER

IS

retval NUMBER := 0;

sqlStmt varchar2(32767);

BEGIN

sqlStmt := '

WITH

SEG\_PER\_DOMAIN as

(

select distinct cdm.circ\_inst\_id

from vzwnet.circ\_domain\_map cdm

join xng\_reports.DOMAINS\_REGIONAL\_REPORTING\_NEW drr

on cdm.domain\_inst\_id = drr.domain\_inst\_id

where drr.sub\_market = :p\_sub\_mkt

)

,

SEGS as

(

select distinct ci.circ\_inst\_id

from vzwnet.circ\_inst ci

JOIN SEG\_PER\_DOMAIN ci\_ids

ON ci.circ\_inst\_id=ci\_ids.circ\_inst\_id

JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id IN (ci.CIRC\_PATH\_INST\_ID, ci.NEXT\_PATH\_INST\_ID)

where ci.status = ''Live''

and upper(ci.bandwidth) !=''DS0''

)

select count(circ\_inst\_id)

from SEGS

';

execute immediate sqlStmt into retval using p\_sub\_mkt;

RETURN retval;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: getSegmentWithPathCount(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: getSegmentWithPathCount(): can''t get counts of segments with paths for '||p\_sub\_mkt

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END getSegmentWithPathCount;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc populates data for 'Compliance %' col. in ES. report

\* merely populates area/region # in here. Does not calc anything

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateXNGAuditData is

sqlStmt varchar2(32767);

begin

-- populates regional

-- NOT area totals in the exec\_summ table and Grand totals

sqlStmt:= 'update xng\_reports.executive\_summary\_work\_new es

set es.OVERALL\_FAULT\_PCT = ( select round((xsa.overall\_fault\_pct)\*100,2) as overall\_fault\_pct

from xng\_reports.xngstd\_fault\_sum xsa

where to\_char(xsa.creation\_date, ''mm-dd-yyyy'')=(select to\_char(max(creation\_date),''mm-dd-yyyy'')

from xng\_reports.xngstd\_fault\_sum)

and xsa.sub\_mkt = es.sub\_mkt

)

';

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateXNGAuditData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateXNGAuditData(): can''t populate XngStdsCompliance numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateXNGAuditData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc updates '% BTP Circs Matched in XNG' col. in ES. report

\* merely populates regional # in here. Does not calc anything

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateBTPAuditData is

sql\_stmt varchar2(32000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work\_new es

set es.ckt\_match\_pct= (

WITH

ROLLUPS as

(

select nvl(dlr.territory, ''Grand'') territory,

nvl(dlr.market\_territory, ''MktTotals'') market\_territory,

nvl(sub\_market, ''Totals'') sub\_market

, round((sum(ckts\_matched)/sum(ckt\_count) \* 100), 2) as ckts\_matched\_pct\_num

from xng\_reports.btp\_xng\_audit\_summary bxas, xng\_reports.domains\_regional\_reporting\_new dlr

where bxas.area = dlr.area

and bxas.region = dlr.region

group by rollup(territory, market\_territory, sub\_market)

)

select ckts\_matched\_pct\_num

from ROLLUPs r

where r.territory = es.territory

and r.market\_territory = es.market\_territory

and r.sub\_market = es.sub\_market

)

/\*update xng\_reports.executive\_summary es

set es.ckt\_match\_pct= (select round((sum(ckts\_matched)/sum(ckt\_count) \* 100),2) as ckts\_matched\_pct

from xng\_reports.btp\_xng\_audit\_summary bxas

where es.territory = bxas.territory

and es.market\_territory = bxas.market\_territory

and es.sub\_market = bxas.sub\_market

)\*/

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateBTPAuditData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateBTPAuditData(): can''t populate BTPXngAudit numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateBTPAuditData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates data for 'Cross Connect Compliance %' col. in ES. report

\* it calcs area and grand totals and puts them in the exec\_summ table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateCrossConAuditData is

sql\_stmt varchar2(32000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work\_new es

set es.compliant\_pct= (

WITH

ROLLUPS as

(

select nvl(territory, ''Grand'') territory, nvl(market\_territory, ''MktTotals'') market\_territory, nvl(sub\_market, ''Totals'') sub\_market

, round(sum(matched\_xconn\_count)/sum(total\_xconn\_count)\*100,2) as dacs\_xc\_compliant\_pct

from xng\_reports.xconnect\_summary

group by rollup(territory, market\_territory, sub\_market)

)

select dacs\_xc\_compliant\_pct

from ROLLUPs r

where r.territory = es.territory

and r.market\_territory = es.market\_territory

and r.sub\_market = es.sub\_market

)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateCrossConAuditData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateCrossConAuditData(): can''t populate DACS cross-connect numbers'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateCrossConAuditData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- this populates area and grand totals for only the following metrics

-- overall\_fault\_pct, seg\_with\_path\_pct, equip\_count, equip\_match\_pct,

-- ckt\_match\_pct, compliant\_pct

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure populateMetricTotals is

sqlStmt varchar2(32767);

BEGIN

BEGIN

-- Calc territory totals by averaging regional pct in the territory

sqlStmt := 'insert into xng\_reports.executive\_summary\_work\_new columns (territory, market\_territory, sub\_market, seg\_with\_path\_pct

, equip\_count, equip\_match\_pct

--, ckt\_match\_pct, compliant\_pct

, extract\_date)

WITH

TERR\_ROLLUP as

(

select territory

,round(avg(SEG\_WITH\_PATH\_PCT), 2) SEG\_WITH\_PATH\_PCT -- seg having paths %

, sum(equip\_count) equip\_count -- vsm device cnt

,round(avg(EQUIP\_MATCH\_PCT), 2) EQUIP\_MATCH\_PCT -- vsm - Xng audit %

--,round(avg(CKT\_MATCH\_PCT), 2) CKT\_MATCH\_PCT -- btp - Xng audit %

--,round(avg(COMPLIANT\_PCT), 2) COMPLIANT\_PCT -- cross-connect audit %

from xng\_reports.executive\_summary\_work\_new

-- "Totals" row will not be there in the table IF the process is run as a unit

-- but if it is being run stand alone, then it might exist.

-- this stmt is here, just to be safe

where sub\_market <> ''Totals''

group by territory

)

select nvl(territory, ''Grand'') territory, ''MktTotals'' market\_territory, ''Totals'' sub\_market, seg\_with\_path\_pct

, equip\_count, equip\_match\_pct

--, ckt\_match\_pct, compliant\_pct

, trunc(sysdate)

from TERR\_ROLLUP

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateMetricTotals(): calc area totals for some metrics '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateMetricTotals(): can''t populate area totals for some metrics'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

BEGIN

-- populates area totals for Xng Standards from xngstd\_summ table

sqlStmt:= 'update xng\_reports.executive\_summary\_work\_new es

set es.OVERALL\_FAULT\_PCT = ( select round(xsa.overall\_fault\_pct,2)

from xng\_reports.xngstd\_audit\_summary\_report xsa,

(select distinct territory, market\_territory, sub\_market, area, region

from xng\_reports.domains\_regional\_reporting\_new) dlr

where substr(xsa.area, 1, instr(xsa.area, '' '')-1) = dlr.area

and instr(xsa.area, '' '') > 0

and xsa.region=''Total Compliance''

and to\_char(xsa.creation\_date, ''mm-dd-yyyy'')=

(select to\_char(max(creation\_date),''mm-dd-yyyy'')

from xng\_reports.xngstd\_audit\_summary\_report)

and dlr.territory = es.territory

)

where territory not Like ''%Grand''

and es.sub\_market = ''Totals''

';

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateMetricTotals(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateMetricTotals(): can''t populate XngStdsCompliance area #'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

-- Calc grand totals by averaging regional totals

BEGIN

sqlStmt := 'insert into xng\_reports.executive\_summary\_work\_new columns (territory, market\_territory, sub\_market, overall\_fault\_pct, seg\_with\_path\_pct

, equip\_count, equip\_match\_pct, ckt\_match\_pct, compliant\_pct, extract\_date)

WITH

GRAND\_ROLLUP as

(

select round(avg(OVERALL\_FAULT\_PCT), 2) OVERALL\_FAULT\_PCT -- Xng Stds Data Audit %

,round(avg(SEG\_WITH\_PATH\_PCT), 2) SEG\_WITH\_PATH\_PCT -- seg having paths %

, sum(equip\_count) equip\_count -- vsm device cnt

,round(avg(EQUIP\_MATCH\_PCT), 2) EQUIP\_MATCH\_PCT -- vsm - Xng audit %

,round(avg(CKT\_MATCH\_PCT), 2) CKT\_MATCH\_PCT -- btp - Xng audit %

,round(avg(COMPLIANT\_PCT), 2) COMPLIANT\_PCT -- cross-connect audit %

from xng\_reports.executive\_summary\_work\_new

-- Area "Totals" row WILL be there in the table

-- Grand total will be calculated based by avg regional pct and not area\_pct

where sub\_market <> ''Totals''

)

select ''Grand'' territory, ''MktTotals'' market\_territory, ''Totals'' sub\_market, overall\_fault\_pct, seg\_with\_path\_pct

, equip\_count, equip\_match\_pct, ckt\_match\_pct, compliant\_pct, trunc(sysdate)

from GRAND\_ROLLUP

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateMetricTotals(): calc grand totals for some metrics '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateMetricTotals(): can''t populate grand totals totals for some metrics'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END populateMetricTotals;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc populates data for 'N2N Overall Backhaul Complinace' in

\* ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateOBSData is

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work\_new es

set es.obs\_comp\_pct= (select overall\_compliance

from (select dlr.territory, dlr.market\_territory, dlr.sub\_market, decode(n2n.area, ''zEnterprise'', ''Grand'', area) area, nvl(n2n.region, ''Totals'') region, OVERALL\_COMPLIANCE

from N2N\_OVERALL\_COMPLIANCE\_SUMM n2n) oc,

xng\_reports.domains\_regional\_reporting\_new dlr

where oc.area = dlr.area

and oc.region = dlr.region) nd

where nd.territory = es.territory

and nd.market\_territory = es.market\_territory

and nd.sub\_market = es.sub\_market)

where to\_char(extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateOBSData(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateOBSData(): can''t populate OBS #'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end populateOBSData;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates data for 'Overall Compliance' col. in

\* ES report, which is the avg. of the percentages displayed

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateOverallPct is

sqlStmt varchar(32767);

BEGIN

begin

sqlStmt := 'update executive\_summary\_work\_new es

set es.OVERALL\_COM\_PCT = round((

nvl(overall\_fault\_pct,0) +

nvl(seg\_with\_path\_pct,0) +

nvl(equip\_match\_pct,0) +

nvl(ckt\_match\_pct,0) +

nvl(compliant\_pct,0)

)/5,2)

where extract\_date = trunc(sysdate)

and territory = ''NNO''

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateOverallPct(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateOverallPct(): can''t calc overall for NNO%'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

end;

begin

sqlStmt := 'update executive\_summary\_work\_new es

set es.OVERALL\_COM\_PCT = round((

nvl(overall\_fault\_pct,0) +

nvl(seg\_with\_path\_pct,0) +

nvl(equip\_match\_pct,0) +

nvl(ckt\_match\_pct,0) +

nvl(compliant\_pct,0) +

nvl(obs\_comp\_pct,0)

)/6,2)

where extract\_date = trunc(sysdate)

and territory <> ''NNO''

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populateOverallPct(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populateOverallPct(): can''t calc overall % for non NNO'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

end populateOverallPct;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc get the prev. reported overall Comp. from executive\_summary\_reported

\* table for 'Previous Reported Overall Compliance' col in ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populatePrevGrandTotalsRow is

sql\_stmt varchar2(2000);

begin

-- populate values of last months grand totals

sql\_stmt:= 'insert into xng\_reports.executive\_summary\_work\_new es (territory, market\_territory, sub\_market, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

)

select ''Previous Grand'', ''TotMkt'',''Totals''

, trunc(sysdate), esr.overall\_fault\_pct

, esr.seg\_with\_path\_pct, esr.equip\_count, esr.equip\_match\_pct

, esr.ckt\_match\_pct, esr.compliant\_pct, esr.obs\_comp\_pct, esr.overall\_com\_pct

from xng\_reports.executive\_summary\_reported\_new esr

where esr.extract\_date=(select max(extract\_date)

from xng\_reports.executive\_summary\_reported\_new)

and esr.territory = ''Grand''

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevGrandTotals: Populating prev grand totals'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populatePrevGrandTotalsRow(): can''t populate prev months metric totals'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populatePrevGrandTotalsRow;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc get the prev. reported overall Comp. from executive\_summary\_reported

\* table for 'Previous Reported Overall Compliance' col in ES report

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure populateDeltasRow is

sql\_stmt varchar2(2000);

begin

-- populate values of last months grand totals

sql\_stmt:= 'insert into xng\_reports.executive\_summary\_work\_new es (territory, market\_territory, sub\_market, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

)

select ''Delta'', '''', ''''

, trunc(sysdate)

, (nvl(es.overall\_fault\_pct, 0)-nvl(esr.overall\_fault\_pct, 0))

, (nvl(es.seg\_with\_path\_pct, 0)-nvl(esr.seg\_with\_path\_pct, 0))

, (nvl(es.equip\_count, 0)-nvl(esr.equip\_count, 0))

, (nvl(es.equip\_match\_pct, 0)-nvl(esr.equip\_match\_pct, 0))

, (nvl(es.ckt\_match\_pct, 0)-nvl(esr.ckt\_match\_pct, 0))

, (nvl(es.compliant\_pct, 0)-nvl(esr.compliant\_pct, 0))

, (nvl(es.obs\_comp\_pct, 0)-nvl(esr.obs\_comp\_pct, 0))

, (nvl(es.overall\_com\_pct, 0)-nvl(esr.overall\_com\_pct, 0))

from xng\_reports.executive\_summary\_work\_new es

join xng\_reports.executive\_summary\_reported\_new esr

on es.territory = esr.territory

and esr.market\_territory = es.market\_territory

and esr.sub\_market = es.sub\_market

where esr.extract\_date=(select max(extract\_date) from xng\_reports.executive\_summary\_reported\_new)

and to\_char(es.extract\_date,''MM-DD-YYYY'')=to\_char(sysdate, ''MM-DD-YYYY'')

and esr.territory = ''Grand''

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevGrandTotals: Populating prev grand totals'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populatePrevGrandTotalsRow(): can''t populate metric deltas between current months and prev month'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populateDeltasRow;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This proc calculates

\* the diff. between the current overall comp and prev. reported overall comp.

\* It also gets the prev. reported grand totals for 'Previous Reported Grand Totals'

\* row and changes calculates deltas for the 'Change' row.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Procedure populatePrevAndDeltasCol is

cursor prevAndDelta is

select es.area, es.region, esr.overall\_com\_pct, (nvl(es.overall\_com\_pct, 0)-nvl(esr.overall\_com\_pct, 0)) deltas

from xng\_reports.executive\_summary es

join xng\_reports.executive\_summary\_reported esr

on es.area=esr.area

and esr.region=es.region

where esr.extract\_date=(select max(extract\_date) from xng\_reports.executive\_summary\_reported)

and to\_char(es.extract\_date,'MM-DD-YYYY')=to\_char(sysdate, 'MM-DD-YYYY')

--and es.area <> 'Previous Grand'

;

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work\_new es

set es.delta = :deltas

, es.PREV\_RPT\_OVERALL\_COM\_PCT = :overall\_com\_pct

where es.area = :area

and es.region = :region

';

-- populate col prev and deltas

for rec in prevAndDelta

Loop

BEGIN

--dbms\_output.put\_line(rec.area||','|| rec.region||','|| rec.deltas||','|| rec.overall\_com\_pct);

EXECUTE IMMEDIATE sql\_stmt using rec.deltas, rec.overall\_com\_pct, rec.area, rec.region;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevAndDeltasCol: Populating col prev/delta'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

END;

end loop;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevAndDeltasCol: Populating prev grand totals'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populatePrevAndDeltasCol(): can''t populate area and regional deltas between current month and prev month'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populatePrevAndDeltasCol;

\*/

Procedure populatePrevAndDeltasCol is

cursor prevAndDelta is

select es.territory, es.market\_territory, es.sub\_market, esr.overall\_com\_pct, (nvl(es.overall\_com\_pct, 0)-nvl(esr.overall\_com\_pct, 0)) deltas

from xng\_reports.executive\_summary\_work\_new es

join xng\_reports.executive\_summary\_reported\_new esr

on es.territory = esr.territory

and esr.market\_territory = es.market\_territory

and esr.sub\_market = es.sub\_market

where esr.extract\_date=(select max(extract\_date) from xng\_reports.executive\_summary\_reported\_new)

--and to\_char(es.extract\_date,'MM-DD-YYYY')=to\_char(sysdate, 'MM-DD-YYYY')

--and es.area <> 'Previous Grand'

;

sql\_stmt varchar2(2000);

begin

sql\_stmt:= 'update xng\_reports.executive\_summary\_work\_new es

set es.delta = :deltas

, es.PREV\_RPT\_OVERALL\_COM\_PCT = :overall\_com\_pct

where es.territory = :territory

and es.market\_territory = :market\_territory

and es.sub\_market = :sub\_market

';

-- populate col prev and deltas

for rec in prevAndDelta

Loop

BEGIN

--dbms\_output.put\_line(rec.area||','|| rec.region||','|| rec.deltas||','|| rec.overall\_com\_pct);

EXECUTE IMMEDIATE sql\_stmt using rec.deltas, rec.overall\_com\_pct, rec.territory, rec.market\_territory, rec.sub\_market;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevAndDeltasCol: Populating col prev/delta'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

END;

end loop;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: populatePrevAndDeltasCol: Populating prev grand totals'||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: populatePrevAndDeltasCol(): can''t populate area and regional deltas between current month and prev month'

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END populatePrevAndDeltasCol;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* in\_progress data is copied from WORK to reported.

\* before copying the reported data is copied to a bak table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure copyFromInProgressToReported is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from exec\_summ to exec\_summ\_rptd

BEGIN

sql\_stmt:='truncate table xng\_reports.executive\_summary\_rptd\_bak\_new';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

-- do named inserts. select \* causes issues when columns are added

sql\_stmt:='insert into executive\_summary\_rptd\_bak\_new columns (

territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

)

select territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

from executive\_summary\_reported\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

BEGIN

sql\_stmt:='truncate table xng\_reports.executive\_summary\_reported\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

BEGIN

sql\_stmt:='insert into executive\_summary\_reported\_new columns (

territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

)

select territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

from executive\_summary\_new

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting executive\_summary\_rptd '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert in exec\_summ\_rptd '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Truncating executive\_summary '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc exec\_summ table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting executive\_summary\_reported\_bak '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into exec\_summ\_rptd table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating executive\_summary\_reported\_bak '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc exec\_summ\_rptd\_bak table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END copyFromInProgressToReported;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* data is copied from WORK to in\_progress.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure copyFromWorkToInProgress is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from exec\_summ to exec\_summ\_rptd

BEGIN

sql\_stmt:='truncate table xng\_reports.executive\_summary\_new';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

-- do named inserts. select \* causes issues when columns are added

sql\_stmt:='insert into executive\_summary\_new columns (

territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

)

select territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, obs\_comp\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

from executive\_summary\_work\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): Inserting executive\_summary '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant insert into exec\_summ table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryInProgress(): truncating executive\_summary '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInProgress(): cant trunc exec\_summ table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END copyFromWorkToInProgress;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* call this procedure to generate a backup of the previous summary data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure createPreviousBackup is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from EXECUTIVE\_SUMMARY\_PREVIOUS to EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK

BEGIN

sql\_stmt:='truncate table xng\_reports.EXECUTIVE\_SUMMARY\_PREV\_BAK\_NEW ';

EXECUTE IMMEDIATE sql\_stmt;

BEGIN

sql\_stmt:='insert into EXECUTIVE\_SUMMARY\_PREV\_BAK\_NEW columns (

territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

)

select territory, market\_territory, sub\_market

, extract\_date

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, compliant\_pct

, overall\_com\_pct

, PREV\_RPT\_OVERALL\_COM\_PCT

, delta

, OBS\_COMP\_PCT

from executive\_summary\_previous\_new';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryReported(): Inserting EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryInReported(): cant insert into EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryReported(): truncating EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryReported(): cant trunc EXECUTIVE\_SUMMARY\_PREVIOUS\_BAK table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END createPreviousBackup;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* data is copied from Reported to Previous

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure copyFromReportedToPrevious is

sql\_stmt varchar(32767);

BEGIN

-- Copy data from exec\_summ\_rptd to exec\_summ\_previous

BEGIN

-- do named inserts. select \* causes issues when columns are added

sql\_stmt:=' insert into xng\_reports.executive\_summary\_previous\_new columns (

territory, market\_territory, sub\_market

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, extract\_date

, compliant\_pct

, overall\_com\_pct

, prev\_rpt\_overall\_com\_pct

, delta

, obs\_comp\_pct

)

select territory, market\_territory, sub\_market

, overall\_fault\_pct

, seg\_with\_path\_pct

, equip\_count

, equip\_match\_pct

, ckt\_match\_pct

, extract\_date

, compliant\_pct

, overall\_com\_pct

, prev\_rpt\_overall\_com\_pct

, delta

, obs\_comp\_pct

from executive\_summary\_reported\_new ';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryReported(): Inserting executive\_summary\_reported into executive\_summary\_previous '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryReported(): cannot insert into executive\_summary\_previous table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END copyFromReportedToPrevious;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Delete Old Previous

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure deleteOldPrevious is

sql\_stmt varchar(32767);

BEGIN

-- Previous summaries must be kept for 12 months - this deletes any older than that

BEGIN

-- delete previous summaries over 13 months old

sql\_stmt:=' delete xng\_reports.executive\_summary\_previous\_new

where extract\_date < sysdate - 395';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: executiveSummaryReported(): Deleting executive\_summary\_previous over 13 months old '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (processName,

8000,

SUBSTR ( 'Error: executiveSummaryReported(): cannot delete from executive\_summary\_previous table '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END deleteOldPrevious;

END;

/

--------------------------------------------------------

-- DDL for Package Body GEOPLAN\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."GEOPLAN\_PKG" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: GEOPLAN\_PKG

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 5/31/2013 David He 1. Created this package body.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE backup\_geoplan\_discovery IS

sqlStmt varchar2(500);

message varchar2(250);

BEGIN

sqlStmt := 'delete from geoplan\_discovery';

message := 'Error to execute:' || sqlStmt;

EXECUTE IMMEDIATE sqlStmt;

sqlStmt := 'insert into geoplan\_discovery select \* from geoplan\_discovery\_wk';

message := 'Error to execute:' || sqlStmt;

EXECUTE IMMEDIATE sqlStmt;

COMMIT;

dbms\_output.put\_line('Done backup\_geoplan\_discovery');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr(message||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

PROCEDURE backup\_geoplan\_sc\_discovery IS

sqlStmt varchar2(500);

message varchar2(250);

BEGIN

sqlStmt := 'delete from geoplan\_smallcell\_discovery';

message := 'Error to execute:' || sqlStmt;

EXECUTE IMMEDIATE sqlStmt;

sqlStmt := 'insert into geoplan\_smallcell\_discovery select \* from geoplan\_smallcell\_discovery\_wk';

message := 'Error to execute:' || sqlStmt;

EXECUTE IMMEDIATE sqlStmt;

COMMIT;

dbms\_output.put\_line('Done backup\_geoplan\_smallcell\_discovery');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr(message||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

END GEOPLAN\_PKG;

/

--------------------------------------------------------

-- DDL for Package Body HPOV\_NCM\_EH\_GI\_CSR\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."HPOV\_NCM\_EH\_GI\_CSR\_AUDIT"

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NCM - HPOV - eHealth- GI audit

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure load\_ne\_vs\_Xng\_csr\_audit is

sql\_stmt varchar2(32000);

message varchar2(250);

methodName varchar2(30) := 'load\_ne\_vs\_Xng\_csr\_audit';

begin

dbms\_output.put\_line('FYI: In '||methodName);

begin

delete xng\_reports.NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t delete records from table: NCM\_HPOV\_GI\_CI\_CSR\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

BEGIN

-- create a full audit of CSR NCM HPOV vs Xng

sql\_stmt := 'insert into xng\_reports.NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK

columns (csr\_vendor, csr\_device\_name

, ncm\_csr\_hostname, ncm\_csr\_partition, ncm\_csr\_device\_ip

, ncm\_csr\_vlan\_hostname, ncm\_csr\_vlan\_partition, ncm\_csr\_vlan\_device\_ip

, hp\_csr\_hostname, hp\_csr\_ip

, eh\_csr\_hostname, eh\_csr\_ipaddress

, gi\_device\_name, equip\_inst\_id, descr

, Live\_in\_xng

, match\_code

, brix\_csr\_site

)

WITH

NCM\_CSRS as

(

select distinct ncm.hostname, ncm.partition, ncm.device\_ip ncm\_csr\_device\_ip

from ncm\_csr\_wk ncm

)

,

NCM\_CSVLAN as

(

select distinct ncm.hostname, ncm.partition, ncm.device\_ip ncm\_csr\_vlan\_device\_ip

from ncm\_csr\_vlan\_wk ncm

)

,

HPOV as

(

-- new table where all the hpov data should go

-- from here put it in csr\_devices\_wk

select hp.csr\_device\_name, hp.csr\_ip, hp.ne\_status, hp.ne\_source

from hpov\_csr\_devices\_wk hp

--where csr\_device\_name like ''%-CI-%''

),

EHEALTH\_CSR as

(

select ecl.hostname,ecl.ip from xng\_reports.ehealth\_csr\_list ecl where ecl.vendor=''CI''

)

,

Xng as

(

select csr.\*, ei.status ei\_status

from XNG\_CSR\_PARSED\_WK csr

join vzwnet.equip\_inst ei

on ei.EQUIP\_INST\_ID = csr.EQUIP\_INST\_ID

where csr.csr\_device\_name like ''%-CI-%''

)

,

BRIX\_CSR as

(

select brx.cell\_site\_name

from xng\_reports.BRIX\_CSR\_EXTRACT brx

where brx.CELL\_SITE\_NAME like ''%-CI-%''

)

,

NCM\_CSVLAN\_HPOV\_FULL\_JOIN as

(

select case when ncm.hostname is not null then hostname

when hp.csr\_device\_name is not null then hp.csr\_device\_name

end csr\_device\_name

, ncm.hostname ncm\_csr\_vlan\_hostname, ncm.partition ncm\_csr\_vlan\_partition

, ncm.ncm\_csr\_vlan\_device\_ip

, hp.csr\_device\_name hp\_csr\_hostname, hp.csr\_ip hp\_csr\_ip

, case when ncm.hostname is not null

and hp.csr\_device\_name is not null

then ''BOTH''

when ncm.hostname is not null or hp.csr\_device\_name is not null then ''NE Only''

else ''xxNE Only''

end match\_code

from HPOV hp

full outer join NCM\_CSVLAN ncm

on upper(hp.csr\_device\_name) = upper(ncm.hostname)

)

,

NCM\_CSRS\_CSVLAN\_HPOV\_FULL\_JOIN as

(

select case when nc.hostname is not null then nc.hostname

when ncvhfj.csr\_device\_name is not null then ncvhfj.csr\_device\_name

end csr\_device\_name

, ncvhfj.ncm\_csr\_vlan\_hostname, ncvhfj.ncm\_csr\_vlan\_partition, ncvhfj.ncm\_csr\_vlan\_device\_ip

, hp\_csr\_hostname, hp\_csr\_ip --, ne\_status, ne\_source

, nc.hostname ncm\_csr\_hostname, nc.partition ncm\_csr\_partition, nc.ncm\_csr\_device\_ip

, case when nc.hostname is not null

and ncvhfj.csr\_device\_name is not null

and match\_code = ''BOTH''

then ''BOTH''

when nc.hostname is not null then ''NE Only''

when ncvhfj.csr\_device\_name is not null then ''NE Only''

else ''xxNE Only''

end match\_code

from NCM\_CSVLAN\_HPOV\_FULL\_JOIN ncvhfj

full outer join NCM\_CSRS nc

on upper(nc.hostname) = upper(ncvhfj.csr\_device\_name)

) ,

NCM\_CSRS\_HPOV\_EH\_FULL\_JOIN as

(

select case when ecsr.hostname is not null then ecsr.hostname

when nhfj.csr\_device\_name is not null then nhfj.csr\_device\_name

end csr\_device\_name

,nhfj.ncm\_csr\_hostname, nhfj.ncm\_csr\_partition, nhfj.ncm\_csr\_device\_ip

,nhfj.ncm\_csr\_vlan\_hostname, nhfj.ncm\_csr\_vlan\_partition, nhfj.ncm\_csr\_vlan\_device\_ip

,nhfj.hp\_csr\_hostname, nhfj.hp\_csr\_ip

,ecsr.hostname eh\_csr\_hostname,ecsr.ip eh\_csr\_ipaddress

,case when ecsr.hostname is not null and nhfj.csr\_device\_name is not null

and match\_code = ''BOTH''

then ''BOTH''

when ecsr.hostname is not null then ''NE Only''

when nhfj.csr\_device\_name is not null then ''NE Only''

else ''xxNE Only''

end match\_code

from NCM\_CSRS\_CSVLAN\_HPOV\_FULL\_JOIN nhfj

full outer join ehealth\_csr ecsr

on upper(nhfj.csr\_device\_name) = upper(ecsr.hostname)

),

NCM\_CSRS\_HPOV\_EH\_BR\_FULL\_JOIN as

(

select case when br.cell\_site\_name is not null then br.cell\_site\_name

when nhfj.csr\_device\_name is not null then nhfj.csr\_device\_name

end csr\_device\_name

,nhfj.ncm\_csr\_hostname, nhfj.ncm\_csr\_partition, nhfj.ncm\_csr\_device\_ip

,nhfj.ncm\_csr\_vlan\_hostname, nhfj.ncm\_csr\_vlan\_partition, nhfj.ncm\_csr\_vlan\_device\_ip

,nhfj.hp\_csr\_hostname, nhfj.hp\_csr\_ip

,nhfj.eh\_csr\_hostname eh\_csr\_hostname,nhfj.eh\_csr\_ipaddress eh\_csr\_ipaddress

,case when nhfj.csr\_device\_name is not null and br.cell\_site\_name is not null

and match\_code = ''BOTH''

then ''BOTH''

when nhfj.csr\_device\_name is not null then ''NE Only''

when br.cell\_site\_name is not null then ''NE Only''

else ''xxNE Only''

end match\_code

, br.cell\_site\_name

from NCM\_CSRS\_HPOV\_EH\_FULL\_JOIN nhfj

full outer join BRIX\_CSR br

on upper(nhfj.csr\_device\_name) = upper(br.cell\_site\_name)

)

select distinct ''CI'' csr\_vendor,

case when nhefj.csr\_device\_name is not null then nhefj.csr\_device\_name

when xng.csr\_device\_name is not null then xng.csr\_device\_name

end csr\_device\_name

, nhefj.ncm\_csr\_hostname, nhefj.ncm\_csr\_partition, nhefj.ncm\_csr\_device\_ip

, nhefj.ncm\_csr\_vlan\_hostname, nhefj.ncm\_csr\_vlan\_partition, nhefj.ncm\_csr\_vlan\_device\_ip

, nhefj.hp\_csr\_hostname, nhefj.hp\_csr\_ip

, nhefj.eh\_csr\_hostname, nhefj.eh\_csr\_ipaddress

, xng.csr\_device\_name gi\_device\_name, xng.equip\_inst\_id, xng.descr

, case when xng.ei\_status = ''Live'' then ''Y''

when xng.ei\_status is not null then ''N''

end Live\_in\_xng

, case when nhefj.csr\_device\_name is not null

and xng.csr\_device\_name is not null

and match\_code = ''BOTH''

then ''BOTH''

when nhefj.csr\_device\_name is not null then ''NE Only''

when xng.csr\_device\_name is not null then ''Xng Only''

else ''xxNE Only''

end match\_code

, nhefj.cell\_site\_name

from NCM\_CSRS\_HPOV\_EH\_BR\_FULL\_JOIN nhefj

full outer join Xng

on upper(xng.csr\_device\_name) = upper(nhefj.csr\_device\_name)

';

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert CI audit into table: NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

insert\_ci\_csr\_equip\_issues(processName);

end load\_ne\_vs\_Xng\_csr\_audit;

PROCEDURE insert\_ci\_csr\_equip\_issues (processName VARCHAR2)

IS

methodname VARCHAR2 (30) := 'insert\_CI\_csr\_equip\_issues';

MESSAGE VARCHAR2 (500);

sql\_stmt VARCHAR2 (32767);

issue\_table VARCHAR2(50) := 'csr\_device\_audit\_issues\_wk';

insertSqlStart VARCHAR2(100) := 'insert into ' || issue\_table || ' columns (csr\_device\_name, csr\_issue\_id, csr\_vendor ) ';

BEGIN

begin

sql\_stmt := 'delete from ' || issue\_table || ' where csr\_vendor=''CI''';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t delete records from table: ' || issue\_table;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

-- CSR name in NCM CSR Extract not Std Complaint --

insert\_csr\_issues (processname, methodname, insertSqlStart ||

'select distinct hostname, 16, ''CI'' from NCM\_CSR\_WK aud where parse\_status like ''%16%''',

'Can''t insert into ' || issue\_table || ' issue 16');

-- CSR name in eHealth Extract not Std Complaint --

insert\_csr\_issues (processname, methodname, insertSqlStart ||

'select distinct hostname, 25, ''CI'' from ehealth\_csr\_list where error\_code = 25 and vendor = ''CI''',

'Can''t insert into ' || issue\_table || ' issue ehealth 25');

-- CSR name in NCM CSR VLAN Extract not Std Complaint --

insert\_csr\_issues (processname, methodname, insertSqlStart ||

'select distinct hostname, 17, ''CI'' from NCM\_CSR\_VLAN\_WK aud where parse\_status like ''%17%''',

'Can''t insert into ' || issue\_table || ' issue 17');

-- GI name not Std Complaint --

insert\_csr\_issues (processname, methodname, insertSqlStart ||

'select distinct csr\_device\_name, 7, ''CI'' from XNG\_CSR\_PARSED\_WK aud where csr\_device\_name like ''%P-CI%'' and parse\_status like ''%Xng CSR Name does not match Standard%''',

'Can''t insert into ' || issue\_table || ' issue 7');

-- Invalid CSR Name --

insert\_csr\_issues (processname, methodname, insertSqlStart ||

'select distinct csr\_device\_name, 8, ''CI'' from XNG\_CSR\_PARSED\_WK aud where csr\_device\_name like ''%P-CI%'' and parse\_status like ''%Invalid CSR Name%''',

'Can''t insert into ' || issue\_table || ' issue 8');

-- 12-14 issues in the previous are critical ONLY IF match\_code <> 'Xng Only'

-- the scheme needs to change somehow on the csr\_issues-------???????

-- device not found in ncm csr extract

insert\_csr\_issues (processname, methodname, insertSqlStart || 'select distinct csr\_device\_name, 12, ''CI'' from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud where ncm\_csr\_hostname is null',

'Can''t insert into ' || issue\_table || ' issue 12');

-- device not found in eHealth extract

insert\_csr\_issues (processname, methodname, insertSqlStart || 'select distinct csr\_device\_name, 24, ''CI'' from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud where eh\_csr\_hostname is null',

'Can''t insert into ' || issue\_table || ' ehealth-issue 24');

-- device not found in ncm vlan extract

insert\_csr\_issues (processname, methodname, insertSqlStart || 'select distinct csr\_device\_name, 13, ''CI'' from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud where ncm\_csr\_vlan\_hostname is null',

'Can''t insert into ' || issue\_table || ' issue 13');

-- device not found in hpov

insert\_csr\_issues (processname, methodname, insertSqlStart || 'select distinct csr\_device\_name, 14, ''CI'' from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud where hp\_csr\_hostname is null',

'Can''t insert into ' || issue\_table || ' issue 14');

-- device not found in Brix

insert\_csr\_issues (processname, methodname, insertSqlStart || 'select distinct csr\_device\_name, 33, ''CI'' from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud where brix\_csr\_site is null',

'Can''t insert into ' || issue\_table || ' issue 33');

-- device not found in Xng

insert\_csr\_issues (processname, methodname, insertSqlStart ||

'select distinct csr\_device\_name, 15, ''CI'' from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud where equip\_inst\_id is null',

'Can''t insert into ' || issue\_table || ' issue 15');

-- DUP CSR match found in Xng

insert\_csr\_issues (processname, methodname, insertSqlStart ||

'select distinct csr\_device\_name, 1, ''CI'' from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK i where match\_code <> ''Xng Only'' group by i.csr\_device\_name having count(1) > 1',

'Can''t insert into ' || issue\_table || ' issue 1');

-- DUPs found in NE. is this possible ????

-- mark 6-char match-not all- use from before

-- mark Invalid CLLI

-- We might have discovered some CLLI that is not in the list of MTSO CLLI

-- mark those as bad

-- We might have discovered some CLLI where 1st 6-char match MTSO CLLI but all 8 dont

-- mark those as bad

-- some times there are 2 legitimate cllis for the same site

-- in those cases the o/p comes as followd

-- clli\_1\_csr clli\_2\_vsm CLLI Mismatch

-- clli\_2\_csr clli\_1\_vsm CLLI Mismatch

-- so reverse the order and subtract it. That will give only the ones that are

-- truly bad

/\* want to be more specific to NCM/NCM VLAN and HPOV \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--NCM issues 18 and 20

begin

sql\_stmt:= insertSqlStart ||

'WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.ncm\_csr\_hostname,

SUBSTR (ei.ncm\_csr\_hostname, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.ncm\_csr\_hostname, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.ncm\_csr\_hostname, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.ncm\_csr\_hostname

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei

MINUS

SELECT ei.ncm\_csr\_hostname

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.ncm\_csr\_hostname, 1, 8)

MINUS

SELECT ei.ncm\_csr\_hostname

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.ncm\_csr\_hostname, 1, 6)

),

ISSUES AS

(

SELECT cm.ncm\_csr\_hostname, 20 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.ncm\_csr\_hostname, 18 issue\_id

FROM bad\_clli bc

)

SELECT distinct WK.CSR\_DEVICE\_NAME , i.issue\_id, ''CI''

FROM ISSUES i, NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK wk

where i.ncm\_csr\_hostname = Wk.NCM\_CSR\_HOSTNAME

';

insert\_csr\_issues (processname, methodname, sql\_stmt, 'Can''t insert into ' || issue\_table || ' issue 18 and 20');

end;

--eHealth issues 26 and 27

begin

sql\_stmt:= insertSqlStart ||

'WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.eh\_csr\_hostname,

SUBSTR (ei.eh\_csr\_hostname, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.ncm\_csr\_hostname, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.eh\_csr\_hostname, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.eh\_csr\_hostname

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei

MINUS

SELECT ei.eh\_csr\_hostname

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.eh\_csr\_hostname, 1, 8)

MINUS

SELECT ei.eh\_csr\_hostname

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.eh\_csr\_hostname, 1, 6)

),

ISSUES AS

(

SELECT cm.eh\_csr\_hostname, 27 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.eh\_csr\_hostname, 26 issue\_id

FROM bad\_clli bc

)

SELECT distinct WK.CSR\_DEVICE\_NAME , i.issue\_id, ''CI''

FROM ISSUES i, NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK wk

where i.eh\_csr\_hostname = Wk.NCM\_CSR\_HOSTNAME

';

insert\_csr\_issues (processname, methodname, sql\_stmt, 'Can''t insert into ' || issue\_table || ' issue 27 and 26');

end;

--NCM VLAN issues 22 and 23

begin

sql\_stmt:= insertSqlStart ||

'WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT EI.NCM\_CSR\_VLAN\_HOSTNAME ,

SUBSTR (ei.NCM\_CSR\_VLAN\_HOSTNAME, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.NCM\_CSR\_VLAN\_HOSTNAME, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.NCM\_CSR\_VLAN\_HOSTNAME, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.NCM\_CSR\_VLAN\_HOSTNAME

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei

MINUS

SELECT ei.NCM\_CSR\_VLAN\_HOSTNAME

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.NCM\_CSR\_VLAN\_HOSTNAME, 1, 8)

MINUS

SELECT ei.NCM\_CSR\_VLAN\_HOSTNAME

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.NCM\_CSR\_VLAN\_HOSTNAME, 1, 6)

),

ISSUES AS

(

SELECT cm.NCM\_CSR\_VLAN\_HOSTNAME, 22 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.NCM\_CSR\_VLAN\_HOSTNAME, 23 issue\_id

FROM bad\_clli bc

)

SELECT distinct WK.CSR\_DEVICE\_NAME , i.issue\_id, ''CI''

FROM ISSUES i, NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK wk

where i.NCM\_CSR\_VLAN\_HOSTNAME = Wk.NCM\_CSR\_VLAN\_HOSTNAME

';

insert\_csr\_issues (processname, methodname, sql\_stmt, 'Can''t insert into ' || issue\_table || ' issue 22 and 23');

end;

--HPOV issues 19 and 21

begin

sql\_stmt:= insertSqlStart ||

'WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT EI.HP\_CSR\_HOSTNAME ,

SUBSTR (ei.HP\_CSR\_HOSTNAME, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.HP\_CSR\_HOSTNAME, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.HP\_CSR\_HOSTNAME, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.HP\_CSR\_HOSTNAME

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei

MINUS

SELECT ei.HP\_CSR\_HOSTNAME

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.HP\_CSR\_HOSTNAME, 1, 8)

MINUS

SELECT ei.HP\_CSR\_HOSTNAME

FROM NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.HP\_CSR\_HOSTNAME, 1, 6)

),

ISSUES AS

(

SELECT cm.HP\_CSR\_HOSTNAME, 21 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.HP\_CSR\_HOSTNAME, 19 issue\_id

FROM bad\_clli bc

)

SELECT distinct WK.CSR\_DEVICE\_NAME , i.issue\_id, ''CI''

FROM ISSUES i, NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK wk

where i.HP\_CSR\_HOSTNAME = Wk.HP\_CSR\_HOSTNAME

';

insert\_csr\_issues (processname, methodname, sql\_stmt, 'Can''t insert into ' || issue\_table || ' issue 21 and 19');

end;

BEGIN

-- if some ne\_hostname matched > 1 granite ne\_inst\_id then mark it dup

update NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud set match\_code = 'NE Only'

where exists (select 1

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK i

where aud.csr\_device\_name = i.csr\_device\_name

group by i.csr\_device\_name

having count(1) > 1

)

and match\_code <> 'Xng Only'

;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t mark match\_code Duplicate in Granite';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END insert\_ci\_csr\_equip\_issues;

PROCEDURE insert\_csr\_issues (processname VARCHAR2, methodname VARCHAR2, insertSql VARCHAR2, errorMsg VARCHAR2)

IS

MESSAGE VARCHAR2 (500);

BEGIN

BEGIN

EXECUTE IMMEDIATE insertSql;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE := 'Error: ' || methodname|| '(): ' || errorMsg;

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END insert\_csr\_issues;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NCM vs HPOV vs Xng, store summary by CLLI\_6

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure load\_csr\_summary\_by\_clli is

-- get the all CSRs found in NCM CSR Extract

cursor CSRs\_NCM is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

,

NCM as

(

select distinct hostname, substr(hostname, 1, 6) clli\_6

from ncm\_csr\_wk ncm

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, ncm.clli\_6

, count (1) ncm\_csrs

from DOMAINS di

right outer join NCM

on ncm.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, ncm.clli\_6

;

-- get the all CSRs found in eHealth CSR Extract

cursor CSRs\_EHEALTH is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

,

EHEALTH as

(

select distinct hostname, substr(hostname, 1, 6) clli\_6

from ehealth\_csr\_list where vendor = 'CI'

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, eh.clli\_6

, count (1) eh\_csrs

from DOMAINS di

right outer join EHEALTH eh

on eh.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, eh.clli\_6

;

-- get the all CSRs found in NCM CSR-VLAN Extract

cursor CSRs\_VLAN\_NCM is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

,

NCM as

(

select distinct hostname, substr(hostname, 1, 6) clli\_6

from ncm\_csr\_vlan\_wk ncm

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, ncm.clli\_6

, count (1) ncm\_csrs

from DOMAINS di

right outer join NCM

on ncm.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, ncm.clli\_6

;

-- get the all CSRs found in Brix CSR Extract

cursor CSRs\_BRIX is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

,

BRIX as

(

select distinct cell\_site\_name, substr(cell\_site\_name, 1, 6) clli\_6

from BRIX\_CSR\_EXTRACT where cell\_site\_name like '%-CI-%'

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, br.clli\_6

, count (1) brix\_csrs

from DOMAINS di

right outer join BRIX br

on br.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, br.clli\_6

;

--Brix\_csr\_issues

cursor BRIX\_CSRs\_W\_ISSUES is

with

domains as

(

select distinct substr(clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

),

brix\_csr\_w\_issues as (

select distinct cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id in (33)

and CAI.CSR\_VENDOR = 'CI'

),

brix as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6 from brix\_csr\_w\_issues brix

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, brix.clli\_6

, count (1) brix\_csrs\_w\_issues

from DOMAINS di

right outer join brix

on brix.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, brix.clli\_6 order by area, region

;

--LR

--ncm\_csr\_issues - in 12,13 but not in both

cursor NCM\_CSRs\_W\_ISSUES is

with

domains as

(

select distinct substr(clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

),

ncm\_csr\_w\_issues as (

select distinct cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id in (20, 22, 23, 12, 13, 16, 17, 18)

and CAI.CSR\_VENDOR='CI'

minus

(

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =12

and CAI.CSR\_VENDOR='CI'

intersect

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =13

and CAI.CSR\_VENDOR='CI'

)

),

ncm as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6 from ncm\_csr\_w\_issues ncm

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, ncm.clli\_6

, count (1) ncm\_csrs\_w\_issues

from DOMAINS di

right outer join ncm

on ncm.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, ncm.clli\_6 order by area, region

;

--LR

--ncm\_hpov\_issues ncm\_vlan vs hpov

cursor NCM\_HPOV\_ISSUES is

with

domains as

(

select distinct substr(clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

),

ncm\_hpov\_issues as (

select cai.csr\_device\_name

from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id in (13, 14, 17, 19, 21, 22, 23) and CAI.CSR\_VENDOR='CI'

minus

(

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =14 and CAI.CSR\_VENDOR='CI'

intersect

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =13 and CAI.CSR\_VENDOR='CI'

)

),

ncm\_hpov as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6

from ncm\_hpov\_issues ncm

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, ncm.clli\_6

, count (1) ncm\_hpov

from DOMAINS di

right outer join ncm\_hpov ncm

on ncm.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, ncm.clli\_6 order by area, region

;

--LR

--ncm\_hpov\_issues ncm\_vlan vs gi

cursor NCM\_GI\_ISSUES is

with

domains as

(

select distinct substr(clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

),

ncm\_gi\_issues as (

select cai.csr\_device\_name

from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id in (17,1,7,8,13,15,22,23) and CAI.CSR\_VENDOR='CI'

minus

(

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =13 and CAI.CSR\_VENDOR='CI'

intersect

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =15 and CAI.CSR\_VENDOR='CI'

)

),ncm\_gi as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6

from ncm\_gi\_issues ncm

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, ncm.clli\_6

, count (1) ncm\_gi

from DOMAINS di

right outer join ncm\_gi ncm

on ncm.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, ncm.clli\_6 order by area, region

;

--EH\_GI\_issues

cursor EH\_GI\_ISSUES is

with

domains as

(

select distinct substr(clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

),

eh\_gi\_issues as (

select cai.csr\_device\_name

from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id in (1,27,8,26,24,25,15) and CAI.CSR\_VENDOR='CI'

minus

(

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =24 and CAI.CSR\_VENDOR='CI'

intersect

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =15 and CAI.CSR\_VENDOR='CI'

)

),eh\_gi as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6

from eh\_gi\_issues eh

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, eh.clli\_6

, count (1) eh\_gi

from DOMAINS di

right outer join eh\_gi eh

on eh.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, eh.clli\_6 order by area, region

;

--LR

--ncm\_hpov\_issues hpov vs gi

cursor HPOV\_GI\_ISSUES is

with

domains as

(

select distinct substr(clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

),

hpov\_gi\_issues as (

select cai.csr\_device\_name

from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id in (1, 21, 8, 19,14, 15) and CAI.CSR\_VENDOR='CI'

minus

(

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =14 and CAI.CSR\_VENDOR='CI'

intersect

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id =15 and CAI.CSR\_VENDOR='CI'

)

),hpov\_gi as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6

from hpov\_gi\_issues

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, hpovgi.clli\_6

, count (1) hpov\_gi

from DOMAINS di

right outer join hpov\_gi hpovgi

on hpovgi.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, hpovgi.clli\_6 order by area, region

;

cursor BRIX\_GI\_ISSUES is

with

domains as

(

select distinct substr(clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

),

brix\_gi\_issues as (

select cai.csr\_device\_name

from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id in (1, 8, 33, 15) and CAI.CSR\_VENDOR='CI'

minus

(

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id = 33 and CAI.CSR\_VENDOR='CI'

intersect

select cai.csr\_device\_name from csr\_device\_audit\_issues\_wk cai

where cai.csr\_issue\_id = 15 and CAI.CSR\_VENDOR='CI'

)

),

brix\_gi as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6

from brix\_gi\_issues

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, nvl(brixgi.clli\_6, 'un') clli\_6

, count (1) brix\_gi

from DOMAINS di

right outer join brix\_gi brixgi

on brixgi.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, brixgi.clli\_6 order by area, region

;

-- get the all CSRs found in HPOV

cursor CSRs\_HPOV is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

,

HPOV as

(

select distinct substr(csr\_device\_name, 1, 6) clli\_6, csr\_device\_name

from HPOV\_CSR\_DEVICES\_WK hp

where csr\_device\_name like '%-CI-%'

)

select nvl(area, 'unknown') area, nvl(region, 'unknown') region, nvl(market, 'unknown') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, hp.clli\_6

, count (1) hpov\_csrs

from DOMAINS di

right outer join HPOV hp

on hp.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, hp.clli\_6

;

-- get mismatches between ncm-csr-vlan extract and hpov

CURSOR NCM\_HPOV\_MATCH\_CSRs is

WITH

NCM\_HPOV\_MATCH as

(

select csr\_device\_name

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

where aud.NCM\_CSR\_VLAN\_HOSTNAME is not null

and aud.HP\_CSR\_HOSTNAME is not null

)

select case when ma.csr\_device\_name is not null then substr(ma.csr\_device\_name, 1, 6)

end clli\_6

, count(ma.csr\_device\_name) ncm\_hpov\_match

from NCM\_HPOV\_MATCH ma

group by case when ma.csr\_device\_name is not null then substr(ma.csr\_device\_name, 1, 6)

end

;

-- make sure duplicate matches in GI are not considered as good match-------??????

-- DUP means > 1 Live ones in Xng

-- if 1 is live and others are not, then that is NOT a DIP. Shirley's logic

-- this should be identifiable from the audit

-- get mismatches between ncm-csr-vlan extract and gi

CURSOR NCM\_GI\_MATCH\_CSRs is

WITH

NCM\_GI\_MATCH as

(

select csr\_device\_name

, case when live\_in\_xng = 'Y' then 1 else 0 end ncm\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end ncm\_gi\_nl\_match

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

where aud.NCM\_CSR\_VLAN\_HOSTNAME is not null

and aud.GI\_DEVICE\_NAME is not null

minus

-- DUPS are NOT good

select aud.csr\_device\_name

, case when live\_in\_xng = 'Y' then 1 else 0 end ncm\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end ncm\_gi\_nl\_match

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

join csr\_device\_audit\_issues\_wk cdai

on upper(cdai.csr\_device\_name) = upper(aud.csr\_device\_name)

where csr\_issue\_id = 1

and match\_code <> 'Xng Only'

)

select

case when ma.csr\_device\_name is not null then substr(ma.csr\_device\_name, 1, 6)

end clli\_6

, count(ma.csr\_device\_name) ncm\_gi\_match

, sum(ma.ncm\_gi\_l\_match) ncm\_gi\_l\_match

, sum(ma.ncm\_gi\_nl\_match) ncm\_gi\_nl\_match

from NCM\_GI\_MATCH ma

group by

case when ma.csr\_device\_name is not null then substr(ma.csr\_device\_name, 1, 6)

end

;

CURSOR EH\_GI\_MATCH\_CSRs is

WITH

EH\_GI\_MATCH as

(

select csr\_device\_name

, case when live\_in\_xng = 'Y' then 1 else 0 end eh\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end eh\_gi\_nl\_match

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

where aud.eh\_csr\_hostname is not null

and aud.GI\_DEVICE\_NAME is not null

minus

-- DUPS are NOT good

select aud.csr\_device\_name

, case when live\_in\_xng = 'Y' then 1 else 0 end eh\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end eh\_gi\_nl\_match

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

join csr\_device\_audit\_issues\_wk cdai

on upper(cdai.csr\_device\_name) = upper(aud.csr\_device\_name)

where csr\_issue\_id = 1

and match\_code <> 'Xng Only'

)

select

case when ma.csr\_device\_name is not null then substr(ma.csr\_device\_name, 1, 6)

end clli\_6

, count(ma.csr\_device\_name) eh\_gi\_match

, sum(ma.eh\_gi\_l\_match) eh\_gi\_l\_match

, sum(ma.eh\_gi\_nl\_match) eh\_gi\_nl\_match

from eh\_GI\_MATCH ma

group by

case when ma.csr\_device\_name is not null then substr(ma.csr\_device\_name, 1, 6)

end

;

-- make sure duplicate matches in GI are not considered as good match-------??????

-- DUP means > 1 Live ones in Xng

-- if 1 is live and others are not, then that is NOT a DUP. Shirley's logic

-- this should be identifiable from the audit

CURSOR HPOV\_GI\_MATCH\_CSRs is

WITH

HPOV\_GI\_MATCH as

(

select csr\_device\_name --, match\_status, live\_in\_xng

, case when live\_in\_xng = 'Y' then 1 else 0 end hpov\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end hpov\_gi\_nl\_match

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

where aud.HP\_CSR\_HOSTNAME is not null

and aud.GI\_DEVICE\_NAME is not null

minus

-- DUPS are NOT good

select aud.csr\_device\_name

, case when live\_in\_xng = 'Y' then 1 else 0 end ncm\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end ncm\_gi\_nl\_match

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

join csr\_device\_audit\_issues\_wk cdai

on upper(cdai.csr\_device\_name) = upper(aud.csr\_device\_name)

where csr\_issue\_id = 1

and match\_code <> 'Xng Only'

)

select case when ma.csr\_device\_name is not null then substr(ma.csr\_device\_name, 1, 6)

end clli\_6

, count(ma.csr\_device\_name) hpov\_gi\_match

, sum(ma.hpov\_gi\_l\_match) hpov\_gi\_l\_match

, sum(ma.hpov\_gi\_nl\_match) hpov\_gi\_nl\_match

from HPOV\_GI\_MATCH ma

group by case when ma.csr\_device\_name is not null then substr(ma.csr\_device\_name, 1, 6)

end

;

-- make sure duplicate matches in GI are not considered as good match-------??????

-- DUP means > 1 Live ones in Xng

-- if 1 is live and others are not, then that is NOT a DUP. Shirley's logic

-- this should be identifiable from the audit

CURSOR BRIX\_GI\_MATCH\_CSRs is

WITH

BRIX\_GI\_MATCH as

(

select csr\_device\_name --, match\_status, live\_in\_xng

, case when live\_in\_xng = 'Y' then 1 else 0 end brix\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end brix\_gi\_nl\_match

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

where aud.BRIX\_CSR\_SITE is not null

and aud.GI\_DEVICE\_NAME is not null

minus

-- DUPS are NOT good

select aud.csr\_device\_name

, case when live\_in\_xng = 'Y' then 1 else 0 end ncm\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end ncm\_gi\_nl\_match

from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

join csr\_device\_audit\_issues\_wk cdai

on upper(cdai.csr\_device\_name) = upper(aud.csr\_device\_name)

where csr\_issue\_id = 1

and match\_code <> 'Xng Only'

)

select case when br.csr\_device\_name is not null then substr(br.csr\_device\_name, 1, 6)

end clli\_6

, count(br.csr\_device\_name) brix\_gi\_match

, sum(br.brix\_gi\_l\_match) brix\_gi\_l\_match

, sum(br.brix\_gi\_nl\_match) brix\_gi\_nl\_match

from BRIX\_GI\_MATCH br

group by case when br.csr\_device\_name is not null then substr(br.csr\_device\_name, 1, 6)

end

;

methodName varchar2(30) := 'load\_csr\_summary\_by\_clli';

message varchar2(300);

updStmt varchar2(32767);

insStmt varchar2(32767);

sqlStmt varchar2(32767);

l\_cnt number;

BEGIN

dbms\_output.put\_line('FYI: In '||methodName);

delete NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK ;

-- from xng insert all CLLI and xng# in the table

sqlStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, gi\_csrs, gi\_l\_csrs, gi\_nl\_csrs)

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> ''NNO''

and region <> ''OSS''

)

,

Xng as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6

, nvl(case when status = ''Live'' then 1 else 0 end, 0) l

, nvl(case when status <> ''Live'' then 1 else 0 end, 0) nl

from XNG\_CSR\_PARSED\_WK csr

join vzwnet.equip\_domain\_map edm

on edm.EQUIP\_INST\_ID = csr.EQUIP\_INST\_ID

join vzwnet.equip\_inst ei

on ei.EQUIP\_INST\_ID = csr.EQUIP\_INST\_ID

where csr\_device\_name like ''%-CI-%''

)

select nvl(area, ''unknown'') area, nvl(region, ''unknown'') region, nvl(market, ''unknown'') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, xng.clli\_6

, count (xng.csr\_device\_name) gi\_csrs

, sum (l) gi\_l\_csr

, sum (nl) gi\_nl\_csr

from DOMAINS di

right outer join Xng

on xng.clli\_6 = di.clli\_6

group by area, region, market, di.leaf\_domain\_inst\_id

, xng.clli\_6

';

execute immediate sqlStmt;

commit;

-- insert new or update metric in the same table for total\_csrs found in NCM in the csr-extract

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK set ncm\_csrs = :ncm\_csrs where clli\_6 = :clli\_6';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, ncm\_csrs)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :ncm\_csrs)

';

for rec in CSRs\_NCM

loop

BEGIN

execute immediate updStmt using rec.ncm\_csrs, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for NCM CLLI');

dbms\_output.put\_line('FYI: should insert in NCM' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.ncm\_csrs);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market

, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.ncm\_csrs;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' NCM only CSR extract CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for NCM-CSR-extract in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

commit;

-- insert new or update metric in the same table for total\_csrs found in ehealth in the csr-extract

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set eh\_csrs = :eh\_csrs

where clli\_6 = :clli\_6

';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, eh\_csrs)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :eh\_csrs)

';

for rec in CSRs\_EHEALTH

loop

BEGIN

execute immediate updStmt using rec.eh\_csrs, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for ehealth CLLI');

dbms\_output.put\_line('FYI: should insert in EH' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.eh\_csrs);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market

, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.eh\_csrs;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' EH only CSR extract CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' forEH-CSR-extract in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

commit;

-- insert new or update metric in the same table for total\_csrs found in ehealth in the csr-extract

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set brix\_csrs = :brix\_csrs

where clli\_6 = :clli\_6

';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, eh\_csrs)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :brix\_csrs)

';

for rec in CSRs\_BRIX

loop

BEGIN

execute immediate updStmt using rec.brix\_csrs, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for ehealth CLLI');

dbms\_output.put\_line('FYI: should insert in EH' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.brix\_csrs);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market

, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.brix\_csrs;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' BRIX only CSR extract CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for BRIX-CSR-extract in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

commit;

-- insert new or update metric in the same table for ncm total\_csrs

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set ncm\_csrs\_w\_vlan = :ncm\_csrs\_w\_vlan

where clli\_6 = :clli\_6

';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, ncm\_csrs\_w\_vlan)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :ncm\_csrs\_w\_vlan)

';

for rec in CSRs\_VLAN\_NCM

loop

BEGIN

execute immediate updStmt using rec.ncm\_csrs, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for NCM CLLI');

dbms\_output.put\_line('FYI: should insert in NCM' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.ncm\_csrs);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.ncm\_csrs;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' NCM-CSR-VLAN-extract CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for NCM-CSR-VLAN-extract in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

-- insert new or update metric in the same table for ncm csrs w issues

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set ncm\_csrs\_w\_issues = :ncm\_csrs\_w\_issues

where clli\_6 = :clli\_6

';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, ncm\_csrs\_w\_issues)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :ncm\_csrs\_w\_issues)

';

for rec in NCM\_CSRs\_W\_ISSUES

loop

BEGIN

execute immediate updStmt using rec.ncm\_csrs\_w\_issues, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for NCM CLLI');

dbms\_output.put\_line('FYI: should insert in NCM' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.ncm\_csrs\_w\_issues);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.ncm\_csrs\_w\_issues;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' NCM-CSRs-W-ISSUES CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for NCM-CSR-VLAN-extract in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

-- insert new or update metric in the same table for ncm hpov issues

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set ncm\_hpov\_issues = :ncm\_hpov\_issues

where clli\_6 = :clli\_6

';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, ncm\_hpov\_issues)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :ncm\_hpov\_issues)

';

for rec in NCM\_HPOV\_ISSUES

loop

BEGIN

execute immediate updStmt using rec.ncm\_hpov, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for NCM CLLI');

dbms\_output.put\_line('FYI: should insert in NCM' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.ncm\_hpov);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.ncm\_hpov;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' NCM\_HPOV\_ISSUES CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for NCM\_HPOV\_ISSUES in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

-- insert new or update metric in the same table for ncm gi issues

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set ncm\_gi\_issues = :ncm\_gi\_issues

where clli\_6 = :clli\_6

';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, ncm\_gi\_issues)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :ncm\_gi\_issues)

';

for rec in NCM\_GI\_ISSUES

loop

BEGIN

execute immediate updStmt using rec.ncm\_gi, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for NCM CLLI');

dbms\_output.put\_line('FYI: should insert in NCM' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.ncm\_gi);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.ncm\_gi;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' NCM-GI-ISSUES CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for NCM-GI-ISSUES in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

-- Eh GI issues

-- insert new or update metric in the same table for ncm gi issues

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK set eh\_gi\_issues = :eh\_gi\_issues where clli\_6 = :clli\_6';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, eh\_gi\_issues)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :eh\_gi\_issues)

';

for rec in EH\_GI\_ISSUES

loop

BEGIN

execute immediate updStmt using rec.eh\_gi, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for NCM CLLI');

dbms\_output.put\_line('FYI: should insert in NCM' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.eh\_gi);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.eh\_gi;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' EH-GI-ISSUES CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for EH-GI-ISSUES in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

-- BRIX GI issues

-- insert new or update metric in the same table for ncm gi issues

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK set brix\_gi\_issues = :brix\_gi\_issues where clli\_6 = :clli\_6';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, brix\_gi\_issues)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :brix\_gi\_issues)

';

for rec in BRIX\_GI\_ISSUES

loop

BEGIN

execute immediate updStmt using rec.brix\_gi, rec.clli\_6;

l\_cnt := sql%rowcount;

if (l\_cnt = 0) then

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for BRIX CLLI');

dbms\_output.put\_line('FYI: should insert in NCM' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.clli\_6||' '|| rec.brix\_gi);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.brix\_gi;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||'BRIX-GI-ISSUES CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

--RAISE;

END;

end if;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for BRIX-GI-ISSUES in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

-- insert new or update metric in the same table for hpov total\_csrs

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set hpov\_csrs = :hpov\_csrs

where clli\_6 = :clli\_6

';

insStmt := 'insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, hpov\_csrs)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :hpov\_csrs)

';

for rec in CSRs\_HPOV

loop

BEGIN

execute immediate updStmt using rec.hpov\_csrs, rec.clli\_6;

l\_cnt := sql%rowcount;

dbms\_output.put\_line('FYI: NCM updated Rows' || l\_cnt);

if (l\_cnt = 0) then

dbms\_output.put\_line('FYI: '||l\_cnt||' matching rows for HPOV CLLI');

dbms\_output.put\_line('FYI: should insert in HPOV' || rec.area

||' '|| rec.region||' '|| rec.market||' '

|| rec.leaf\_domain\_inst\_id||' '|| rec.hpov\_csrs);

BEGIN

execute immediate insStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.clli\_6, rec.hpov\_csrs;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t insert '|| rec.clli\_6||' HPOV only CLLI in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

-- RAISE;

END;

end if;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '|| rec.clli\_6||' for HPOV in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

commit;

-- calc matches / mismatches metric between ncm and hpov

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set ncm\_hpov\_match = :ncm\_hpov\_match

where clli\_6 = :clli\_6

';

for rec in NCM\_HPOV\_MATCH\_CSRs

loop

BEGIN

execute immediate updStmt using rec.ncm\_hpov\_match, rec.clli\_6;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update NCM-HPOV match in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK ';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

-- RAISE;

END;

END LOOP;

-- calc matches / mismatches metric between ncm and gi

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set ncm\_gi\_match = :ncm\_gi\_match

, ncm\_gi\_l\_match = :ncm\_gi\_l\_match

, ncm\_gi\_nl\_match = :ncm\_gi\_nl\_match

where clli\_6 = :clli\_6

';

for rec in NCM\_GI\_MATCH\_CSRs

loop

BEGIN

execute immediate updStmt using rec.ncm\_gi\_match, rec.ncm\_gi\_l\_match, rec.ncm\_gi\_nl\_match, rec.clli\_6;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update NCM-GI mismatch in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

-- RAISE;

END;

END LOOP;

commit;

-- calc matches / mismatches metric between eh and gi

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set eh\_gi\_match = :eh\_gi\_match

, eh\_gi\_l\_match = :eh\_gi\_l\_match

, eh\_gi\_nl\_match = :eh\_gi\_nl\_match

where clli\_6 = :clli\_6

';

for rec in EH\_GI\_MATCH\_CSRs

loop

BEGIN

execute immediate updStmt using rec.eh\_gi\_match, rec.eh\_gi\_l\_match, rec.eh\_gi\_nl\_match, rec.clli\_6;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update EH-GI mismatch in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

-- RAISE;

END;

END LOOP;

commit;

-- calc matches / mismatches metric between hpov and gi

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set hpov\_gi\_match = :hpov\_gi\_match

, hpov\_gi\_l\_match = :hpov\_gi\_l\_match

, hpov\_gi\_nl\_match = :hpov\_gi\_nl\_match

where clli\_6 = :clli\_6

';

for rec in HPOV\_GI\_MATCH\_CSRs

loop

BEGIN

execute immediate updStmt using rec.hpov\_gi\_match, rec.hpov\_gi\_l\_match, rec.hpov\_gi\_nl\_match, rec.clli\_6;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update HPOV match in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

-- RAISE;

END;

END LOOP;

-- calc matches / mismatches metric between hpov and gi

updStmt := 'update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set brix\_gi\_match = :brix\_gi\_match

, brix\_gi\_l\_match = :brix\_gi\_l\_match

, brix\_gi\_nl\_match = :brix\_gi\_nl\_match

where clli\_6 = :clli\_6

';

for rec in BRIX\_GI\_MATCH\_CSRs

loop

BEGIN

execute immediate updStmt using rec.brix\_gi\_match, rec.brix\_gi\_l\_match, rec.brix\_gi\_nl\_match, rec.clli\_6;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update BRIX match in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

-- RAISE;

END;

END LOOP;

-- calculate the percentages

BEGIN

update NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

set ncm\_csrs\_w\_vlan\_per = round(nvl(ncm\_csrs\_w\_vlan, 0) \* 2 / decode ((nvl(ncm\_csrs,0) + nvl(ncm\_csrs\_w\_vlan,0)), 0, 1, nvl(ncm\_csrs,0) + nvl(ncm\_csrs\_w\_vlan,0)) \* 100, 2)

, hpov\_gi\_per = round(nvl(hpov\_gi\_l\_match, 0) \* 2 / decode ((nvl(hpov\_csrs,0) + nvl(gi\_l\_csrs,0)), 0, 1, nvl(hpov\_csrs,0) + nvl(gi\_l\_csrs,0)) \* 100, 2)

, ncm\_gi\_per = round(nvl(ncm\_gi\_l\_match, 0) \* 2 / decode ((nvl(ncm\_csrs,0) + nvl(gi\_l\_csrs,0)), 0, 1, nvl(ncm\_csrs,0) + nvl(gi\_l\_csrs,0)) \* 100, 2)

, ncm\_hpov\_per = round(nvl(ncm\_hpov\_match, 0) \* 2 / decode ((nvl(ncm\_csrs,0) + nvl(hpov\_csrs,0)), 0, 1, nvl(ncm\_csrs,0) + nvl(hpov\_csrs,0)) \* 100, 2)

, eh\_gi\_per = round(nvl(eh\_gi\_l\_match, 0) \* 2 / decode ((nvl(eh\_csrs,0) + nvl(gi\_l\_csrs,0)), 0, 1, nvl(eh\_csrs,0) + nvl(gi\_l\_csrs,0)) \* 100, 2)

, brix\_gi\_per = round(nvl(brix\_gi\_l\_match, 0) \* 2 / decode ((nvl(brix\_csrs,0) + nvl(gi\_l\_csrs,0)), 0, 1, nvl(brix\_csrs,0) + nvl(gi\_l\_csrs,0)) \* 100, 2)

;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update percentages in table: NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

-- RAISE;

END;

END load\_csr\_summary\_by\_clli;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NCM vs HPOV vs Xng, gen summary by region by rolling up CLLI\_6 summary

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure load\_csr\_summary\_by\_region is

sql\_stmt varchar2(32000);

message varchar2(250);

methodName varchar2(30) := 'load\_csr\_summary\_by\_region';

BEGIN

dbms\_output.put\_line('FYI: In '||methodName);

-- truncate the table

BEGIN

sql\_stmt:='truncate table ncm\_hpov\_eh\_gi\_csr\_regn\_sum\_wk';

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t truncate table: ncm\_hpov\_eh\_gi\_csr\_region\_summ\_wk';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

BEGIN

-- rollup all summary by REGION

-- need to make sure all area/region show up in this table, this query does not do that.

insert into ncm\_hpov\_eh\_gi\_csr\_regn\_sum\_wk columns

(AREA, REGION, ncm\_csrs, ncm\_csrs\_w\_vlan, hpov\_csrs,gi\_csrs, gi\_l\_csrs, gi\_nl\_csrs

, ncm\_csrs\_w\_issues

, ncm\_hpov\_match, ncm\_hpov\_issues

, ncm\_gi\_match, ncm\_gi\_l\_match, ncm\_gi\_nl\_match, ncm\_gi\_issues

, hpov\_gi\_match, hpov\_gi\_l\_match, hpov\_gi\_nl\_match, hpov\_gi\_issues

, eh\_csrs,eh\_gi\_match,eh\_gi\_l\_match,eh\_gi\_nl\_match,eh\_gi\_issues

, brix\_csrs, brix\_gi\_match, brix\_gi\_l\_match, brix\_gi\_nl\_match, brix\_gi\_issues

)

select dlr.AREA, dlr.REGION

, sum(ncm\_csrs) ncm\_csrs, sum(ncm\_csrs\_w\_vlan) ncm\_csrs\_w\_vlan

, sum(hpov\_csrs) hpov\_csrs

, sum(gi\_csrs) gi\_csrs, sum(gi\_l\_csrs) gi\_l\_csrs, sum(gi\_nl\_csrs) i\_nl\_csrs

, sum(ncm\_csrs\_w\_issues) ncm\_csrs\_w\_issues

, sum(ncm\_hpov\_match) ncm\_hpov\_match

, sum(ncm\_hpov\_issues) ncm\_hpov\_issues

, sum(ncm\_gi\_match) ncm\_gi\_match

, sum(ncm\_gi\_l\_match) ncm\_gi\_l\_match

, sum(ncm\_gi\_nl\_match) ncm\_gi\_nl\_match

, sum(ncm\_gi\_issues) ncm\_gi\_issues

, sum(hpov\_gi\_match) hpov\_gi\_match

, sum(hpov\_gi\_l\_match) hpov\_gi\_l\_match

, sum(hpov\_gi\_nl\_match) hpov\_gi\_nl\_match

, sum(hpov\_gi\_issues) hpov\_gi\_issues

, sum(eh\_csrs) eh\_csrs

, sum(eh\_gi\_match) eh\_gi\_match

, sum(eh\_gi\_l\_match) eh\_gi\_l\_match

, sum(eh\_gi\_nl\_match) eh\_gi\_nl\_match

, sum(eh\_gi\_issues) eh\_gi\_issues

, sum(brix\_csrs) brix\_csrs

, sum(brix\_gi\_match) brix\_gi\_match

, sum(brix\_gi\_l\_match) brix\_gi\_l\_match

, sum(brix\_gi\_nl\_match) brix\_gi\_nl\_match

, sum(brix\_gi\_issues) brix\_gi\_issues

from domains\_regional\_reporting dlr

left outer join NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK metric

on metric.region = dlr.region

where dlr.region not in ('NNO', 'OSS')

group by rollup (dlr.area, dlr.region)

order by dlr.area, dlr.region

;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into table: ncm\_hpov\_eh\_gi\_csr\_regn\_sum\_wk';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

-- calculate the percentages

BEGIN

update NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM\_WK

set ncm\_csrs\_w\_vlan\_per = round(nvl(ncm\_csrs\_w\_vlan, 0) \* 2 / decode ((nvl(ncm\_csrs,0) + nvl(ncm\_csrs\_w\_vlan,0)), 0, 1, nvl(ncm\_csrs,0) + nvl(ncm\_csrs\_w\_vlan,0)) \* 100, 2)

, hpov\_gi\_per = round(nvl(hpov\_gi\_l\_match, 0) \* 2 / decode ((nvl(hpov\_csrs,0) + nvl(gi\_l\_csrs,0)), 0, 1, nvl(hpov\_csrs,0) + nvl(gi\_l\_csrs,0)) \* 100, 2)

, ncm\_gi\_per = round(nvl(ncm\_gi\_l\_match, 0) \* 2 / decode ((nvl(ncm\_csrs,0) + nvl(gi\_l\_csrs,0)), 0, 1, nvl(ncm\_csrs,0) + nvl(gi\_l\_csrs,0)) \* 100, 2)

, ncm\_hpov\_per = round(nvl(ncm\_hpov\_match, 0) \* 2 / decode ((nvl(ncm\_csrs,0) + nvl(hpov\_csrs,0)), 0, 1, nvl(ncm\_csrs,0) + nvl(hpov\_csrs,0)) \* 100, 2)

, eh\_gi\_per = round(nvl(eh\_gi\_l\_match, 0) \* 2 / decode ((nvl(eh\_csrs,0) + nvl(gi\_l\_csrs,0)), 0, 1, nvl(eh\_csrs,0) + nvl(gi\_l\_csrs,0)) \* 100, 2)

, brix\_gi\_per = round(nvl(brix\_gi\_l\_match, 0) \* 2 / decode ((nvl(brix\_csrs,0) + nvl(gi\_l\_csrs,0)), 0, 1, nvl(brix\_csrs,0) + nvl(gi\_l\_csrs,0)) \* 100, 2)

;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update percentages in table: NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

-- RAISE;

END;

END;

procedure move\_csr\_data\_from\_prod\_to\_wk is

sql\_stmt varchar2(32000);

begin

sql\_stmt:='delete from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK select \* from NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK');

sql\_stmt:='delete from NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK select \* from NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK');

sql\_stmt:='delete from NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM\_WK select \* from NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NCM\_HPOV\_EH\_GI\_CSR\_REGN\_SUM\_WK');

sql\_stmt:='delete from csr\_device\_audit\_issues\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into csr\_device\_audit\_issues\_WK select \* from csr\_device\_audit\_issues';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating csr\_device\_audit\_issues\_WK');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in move\_csr\_data\_from\_prod\_to\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror('CSR\_AUDIT',4000,SubStr('Error in move\_csr\_data\_from\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

xng\_reports.watchdog.updateprocessend('HPOV\_NCM\_EH\_GI\_CI\_CSR\_AUDIT',0,'N');

RAISE;

end move\_csr\_data\_from\_prod\_to\_wk;

END hpov\_ncm\_eh\_gi\_csr\_audit;

/

--------------------------------------------------------

-- DDL for Package Body JUNIPER\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."JUNIPER\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: juniper\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/10/2014 SME Cloned from NetSmart Audit.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure RunAudit \*

\* \*

\* Purpose: Performs the Juniper Audit \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE RUNAUDIT IS

PROCESSNAME VARCHAR2(100);

BEGIN

BEGIN

processName := 'JUNIPER\_AUDIT';

-- Tell watchDog the process has started

watchdog.updateprocessstart (processName);

BEGIN

-- find the top level juniper equipment (or containers) in Granite Inventory

GETGRANITEEQUIP ();

-- Audit the loaded device file against the juniper devices in Granite

AUDITEQUIP ();

-- Generate the overall summary table

GENSUMMARY ();

-- mark successful completion of the task

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_SUCCESS', PROCESSNAME || ' ran successfully!', 'N');

WATCHDOG.UPDATEPROCESSEND (PROCESSNAME, 'STATUS\_SUCCESS', 'Y');

WATCHDOG.POPULATE\_LAST\_SUCCESS\_DT\_PROC(PROCESSNAME);

DBMS\_OUTPUT.PUT\_LINE('Juniper\_Audit.RunAudit() ran successfully!');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SUBSTR('Error: Juniper\_Audit.runAudit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SUBSTR('Error: Juniper\_Audit.RunAudit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

-- mark failed completion of the task

WATCHDOG.UPDATEPROCESSEND (PROCESSNAME, 'STATUS\_FAILURE', 'N');

END;

END RUNAUDIT;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure GetGraniteEquip \*

\* \*

\* Purpose: Finds all the top level Fujitsu FlashWave Equipment in Granite Inventory \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE GETGRANITEEQUIP IS

BEGIN

-- from granite get all equip that have vendor = JUNIPER%

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_XNG\_EQUIP';

insert into juniper\_xng\_equip ( NE\_INST\_ID, EQUIP\_INST\_ID, SITE\_INST\_ID,

PARENT\_EQ\_INST\_ID, EQ\_CLASS\_TYPE, DESCR, MODEL, TYPE, STATUS, VENDOR, EQ\_CLASS

)

SELECT TO\_NUMBER

(TRIM

(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH

(DECODE (LEVEL,

1, ei.equip\_inst\_id,

''

),

'/'

)

)

) ne\_inst\_id,

ei.equip\_inst\_id, ei.site\_inst\_id, ei.parent\_eq\_inst\_id,

DECODE (LEVEL,

1, 'NE',

DECODE (ei.eq\_class, 'S', 'SHELF', 'N/A')

) eq\_class\_type,

ei.descr, ei.model, ei.TYPE, ei.status,

ei.vendor, ei.eq\_class

FROM VZWNET.EQUIP\_INST EI

WHERE (LEVEL = 1 OR EI.EQ\_CLASS = 'S') AND EI.TYPE IN ('ROUTER','ETHERNET SWITCH') AND (EI.MODEL LIKE '%8216%' or UPPER(EI.MODEL) LIKE '%MX960%')

START WITH UPPER(EI.VENDOR) LIKE 'JUNIPER%'

--AND EI.TYPE IN ('ROUTER','ETHERNET SWITCH')

CONNECT BY PRIOR ei.equip\_inst\_id = ei.parent\_eq\_inst\_id;

COMMIT;

-- since we dont know top-equip from where to start looking for all children equip

-- in the previous start-with-connect-by the same equip might come is as both NE and SHELF.

-- if it does then delete those rows where it is a NE.

DELETE JUNIPER\_XNG\_EQUIP JXE

where jxe.EQ\_CLASS\_TYPE = 'NE'

and exists (

WITH

TO\_BE\_DELETED as

(

select equip\_inst\_id

from juniper\_xng\_equip

where eq\_class\_type = 'SHELF'

intersect

select equip\_inst\_id

from juniper\_xng\_equip

where eq\_class\_type = 'NE'

)

select 1

FROM TO\_BE\_DELETED TBD

where tbd.equip\_inst\_id = jxe.EQUIP\_INST\_ID

);

COMMIT;

-- delete dups if some equip\_inst\_id comes in > 1 as SHELF

/\*select equip\_inst\_id

from xng\_fw\_equip

where eq\_class\_type = 'SHELF'

group by equip\_inst\_id

having count(1) > 1

;

commit;\*/

-- update the Xng NE rows that dont match Granite naming format

--UPDATE JUNIPER\_XNG\_EQUIP SET PARSE\_STATUS = 'Name not Granite Stds compliant'

--WHERE NOT REGEXP\_LIKE(DESCR, '[[:alnum:]]{8}[[:digit:]]{2,2}[0-9WVRE]-[AELRP]-JU-(8216|960X)-[[:digit:]]{2,2}') -- granite std name

-- /\*and eq\_class\_type = 'NE'\*/;

--COMMIT;

UPDATE JUNIPER\_XNG\_EQUIP SET PARSE\_STATUS = 'Granite Name not MTCE or Juniper Stds compliant'

WHERE NOT (REGEXP\_LIKE(DESCR, '[[:alnum:]]{8}((D[[:digit:]]{2,2})|([[:digit:]][0-9A-P][0-9WVRE]))') OR -- granite std name (Loose Match)

REGEXP\_LIKE(DESCR, '[[:alnum:]]{8}((D[[:digit:]]{2,2})|([[:digit:]][0-9A-P][0-9WVRE]))-[AELRP]-JU-(8216|960X)-[[:digit:]]{2,2}')) -- granite std name (Full Match)

AND UPPER(MODEL) LIKE '%MX960%';

COMMIT;

UPDATE JUNIPER\_XNG\_EQUIP SET PARSE\_STATUS = 'Granite Name not MTCE DNS Stds compliant'

WHERE NOT REGEXP\_LIKE(DESCR, '[[:alnum:]]{8}((D[[:digit:]]{2,2})|([[:digit:]][0-9A-P][0-9WVRE]))-[AELRP]-JU-(8216|960X)-[[:digit:]]{2,2}') -- granite std name (Full Match)

AND UPPER(MODEL) LIKE '%8216%';

COMMIT;

END;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure AuditEquip \*

\* \*

\* Purpose: Audit Juniper Equipment against Granite Inventory \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE AUDITEQUIP IS

BEGIN

-- find matches in juniper and Xng (look only at eq\_class\_type = NE)

--DELETE FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP';

-- Insert Duplicate in NE records

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP (LEAF\_DOMAIN\_NAME, HOST\_NAME, DEVICE\_IP, DEVICE\_ID, DEVICE\_VENDOR, DEVICE\_MODEL, DEVICE\_PARTITION,

MATCH\_CODE, MATCH\_STATUS, NE\_INST\_ID, DESCR, MODEL, VENDOR, MATCH8, FULL\_MATCH)

WITH JUNIPER\_HOSTS AS

(

SELECT

CASE

WHEN REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1) IS NOT NULL

THEN REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1)

ELSE JD.HOST\_NAME

END HOST\_NAME,

JD.DEVICE\_IP,

JD.DEVICE\_ID,

JD.DEVICE\_VENDOR,

JD.DEVICE\_MODEL,

JD.DEVICE\_PARTITION

FROM

JUNIPER\_DEVICES JD

)

SELECT DISTINCT M.LEAF\_DOMAIN\_NAME, JH.HOST\_NAME, JH.DEVICE\_IP, JH.DEVICE\_ID, JH.DEVICE\_VENDOR, JH.DEVICE\_MODEL, JH.DEVICE\_PARTITION

, 'NE Only' MATCH\_CODE

, NVL2(JXE.PARSE\_STATUS, 'Duplicate in NE, '||RTRIM(JXE.PARSE\_STATUS), 'Duplicate in NE') MATCH\_STATUS

, JXE.NE\_INST\_ID, JXE.DESCR, JXE.MODEL, JXE.VENDOR, NULL, NULL

FROM

JUNIPER\_HOSTS JH

JOIN

-- JUNIPER\_XNG\_EQUIP JXE ON JXE.DESCR LIKE '%'|| SUBSTR(JD.HOST\_NAME,1,11) ||'%' -- need to make these match

JUNIPER\_XNG\_EQUIP JXE ON JXE.DESCR LIKE '%'|| JH.HOST\_NAME ||'%' -- need to make these match

LEFT OUTER JOIN

CLLI\_DOMAIN\_MAP\_V M ON SUBSTR(JH.HOST\_NAME,1,6) = SUBSTR(M.CLLI,1,6)

WHERE

JXE.EQ\_CLASS\_TYPE = 'NE'

AND (UPPER(JH.DEVICE\_MODEL) LIKE '%MX960%' OR JH.DEVICE\_MODEL LIKE '%8216%')

AND EXISTS (SELECT 1

FROM JUNIPER\_HOSTS I

WHERE JH.HOST\_NAME = I.HOST\_NAME

group by I.HOST\_NAME

HAVING COUNT(1) > 1

);

COMMIT;

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP (LEAF\_DOMAIN\_NAME, HOST\_NAME, DEVICE\_IP, DEVICE\_ID, DEVICE\_VENDOR, DEVICE\_MODEL, DEVICE\_PARTITION,

MATCH\_CODE, MATCH\_STATUS, NE\_INST\_ID, DESCR, MODEL, VENDOR, MATCH8, FULL\_MATCH)

WITH JUNIPER\_HOSTS AS

(

SELECT

CASE

WHEN REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1) IS NOT NULL

THEN REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1)

ELSE JD.HOST\_NAME

END HOST\_NAME,

JD.DEVICE\_IP,

JD.DEVICE\_ID,

JD.DEVICE\_VENDOR,

JD.DEVICE\_MODEL,

JD.DEVICE\_PARTITION

FROM

JUNIPER\_DEVICES JD

)

SELECT DISTINCT M.LEAF\_DOMAIN\_NAME, JH.HOST\_NAME, JH.DEVICE\_IP, JH.DEVICE\_ID, JH.DEVICE\_VENDOR, JH.DEVICE\_MODEL, JH.DEVICE\_PARTITION

, CASE WHEN JXE.PARSE\_STATUS IS NULL THEN 'BOTH' ELSE 'NE Only' END MATCH\_CODE

, JXE.PARSE\_STATUS

, JXE.NE\_INST\_ID, JXE.DESCR, JXE.MODEL, JXE.VENDOR, NULL, NULL

FROM

JUNIPER\_HOSTS JH

JOIN

-- JUNIPER\_XNG\_EQUIP JXE ON JXE.DESCR LIKE '%'|| SUBSTR(JD.HOST\_NAME,1,11) ||'%' -- need to make these match

JUNIPER\_XNG\_EQUIP JXE ON JXE.DESCR LIKE '%'|| JH.HOST\_NAME ||'%' -- need to make these match

LEFT OUTER JOIN

CLLI\_DOMAIN\_MAP\_V M ON SUBSTR(JH.HOST\_NAME,1,6) = SUBSTR(M.CLLI,1,6)

WHERE

JXE.EQ\_CLASS\_TYPE = 'NE'

AND (UPPER(JH.DEVICE\_MODEL) LIKE '%MX960%' OR JH.DEVICE\_MODEL LIKE '%8216%')

AND NOT EXISTS (SELECT 1

FROM JUNIPER\_HOSTS I

WHERE JH.HOST\_NAME = I.HOST\_NAME

group by I.HOST\_NAME

HAVING COUNT(1) > 1

);

COMMIT;

-- if some TID matched > 1 granite ne\_inst\_id then mark it dup

update juniper\_ne\_vs\_xng\_audit\_temp aud set match\_code = 'NE Only'

, match\_status = nvl2(match\_status, 'Duplicate in Granite, '||rtrim(match\_status), 'Duplicate in Granite')

WHERE

(AUD.MATCH\_STATUS NOT LIKE '%Duplicate in NE%' or AUD.MATCH\_STATUS IS NULL)

AND EXISTS (SELECT 1

FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP I

WHERE AUD.HOST\_NAME = I.HOST\_NAME

group by I.HOST\_NAME

having count(1) > 1

);

COMMIT;

-- insert NE Only

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP (LEAF\_DOMAIN\_NAME, HOST\_NAME, DEVICE\_IP, DEVICE\_ID, DEVICE\_VENDOR, DEVICE\_MODEL, DEVICE\_PARTITION, MATCH\_CODE, MATCH\_STATUS)

WITH JUNIPER\_HOSTS AS

(

SELECT

CASE

WHEN REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1) IS NOT NULL

THEN REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1)

ELSE JD.HOST\_NAME

END HOST\_NAME,

JD.DEVICE\_IP,

JD.DEVICE\_ID,

JD.DEVICE\_VENDOR,

JD.DEVICE\_MODEL,

JD.DEVICE\_PARTITION

FROM

JUNIPER\_DEVICES JD

)

SELECT LEAF\_DOMAIN\_NAME, HOST\_NAME, DEVICE\_IP, DEVICE\_ID, DEVICE\_VENDOR, DEVICE\_MODEL, DEVICE\_PARTITION, 'NE Only', 'No match in Granite'

FROM (

SELECT CDM.LEAF\_DOMAIN\_NAME, JH.HOST\_NAME, JH.DEVICE\_IP, JH.DEVICE\_ID, JH.DEVICE\_VENDOR, JH.DEVICE\_MODEL, JH.DEVICE\_PARTITION

FROM JUNIPER\_HOSTS JH

JOIN CLLI\_DOMAIN\_MAP\_V CDM ON SUBSTR(CDM.CLLI,1,8) = SUBSTR(JH.HOST\_NAME,1,8)

WHERE UPPER(JH.DEVICE\_MODEL) LIKE '%MX960%' or JH.DEVICE\_MODEL LIKE '%8216%'

MINUS

SELECT AUD.LEAF\_DOMAIN\_NAME, AUD.HOST\_NAME, AUD.DEVICE\_IP, AUD.DEVICE\_ID, AUD.DEVICE\_VENDOR, AUD.DEVICE\_MODEL, AUD.DEVICE\_PARTITION

FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD

);

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP (HOST\_NAME, DEVICE\_IP, DEVICE\_ID, DEVICE\_VENDOR, DEVICE\_MODEL, DEVICE\_PARTITION, MATCH\_CODE, MATCH\_STATUS)

WITH JUNIPER\_HOSTS AS

(

SELECT

CASE

WHEN REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1) IS NOT NULL

THEN REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1)

ELSE JD.HOST\_NAME

END HOST\_NAME,

JD.DEVICE\_IP,

JD.DEVICE\_ID,

JD.DEVICE\_VENDOR,

JD.DEVICE\_MODEL,

JD.DEVICE\_PARTITION

FROM

JUNIPER\_DEVICES JD

)

SELECT HOST\_NAME, DEVICE\_IP, DEVICE\_ID, DEVICE\_VENDOR, DEVICE\_MODEL, DEVICE\_PARTITION, 'NE Only', 'No match in Granite'

FROM (

SELECT JH.HOST\_NAME, JH.DEVICE\_IP, JH.DEVICE\_ID, JH.DEVICE\_VENDOR, JH.DEVICE\_MODEL, JH.DEVICE\_PARTITION

FROM JUNIPER\_HOSTS JH

LEFT OUTER JOIN CLLI\_DOMAIN\_MAP\_V CDM ON SUBSTR(CDM.CLLI,1,8) = SUBSTR(JH.HOST\_NAME,1,8)

WHERE UPPER(JH.DEVICE\_MODEL) LIKE '%MX960%' OR JH.DEVICE\_MODEL LIKE '%8216%'

AND CDM.LEAF\_DOMAIN\_NAME IS NULL

MINUS

SELECT AUD.HOST\_NAME, AUD.DEVICE\_IP, AUD.DEVICE\_ID, AUD.DEVICE\_VENDOR, AUD.DEVICE\_MODEL, AUD.DEVICE\_PARTITION

FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD

);

COMMIT;

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP JD

SET JD.HOST\_NAME = REGEXP\_SUBSTR(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$',1,1,'i',1)

WHERE

REGEXP\_LIKE(JD.HOST\_NAME, '(.\*)-re[[:digit:]]$', 'i');

COMMIT;

-- -- check the host name matches MTCE DNS std, else update match\_code = ''NE Only'' and match\_status = ''TID not MTCE Stds compliant''

-- UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET MATCH\_CODE = 'NE Only',

-- MATCH\_STATUS = NVL2(MATCH\_STATUS, 'Host Name not MTCE DNS Stds compliant (Loose Match), '||RTRIM(MATCH\_STATUS), 'Host Name not MTCE DNS Stds compliant (Loose Match)')

-- WHERE NOT REGEXP\_LIKE(AUD.HOST\_NAME, '[[:alnum:]]{8}(D[[:digit:]]{2,2}|[[:digit:]]{2,2}[0-9EWVR])');

-- COMMIT;

--

-- -- check the host name matches MTCE DNS std, else update match\_code = ''NE Only'' and match\_status = ''TID not MTCE Stds compliant''

-- UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET

-- MATCH\_STATUS = NVL2(MATCH\_STATUS, 'Host Name not MTCE DNS Stds compliant (Full Match), '||RTRIM(MATCH\_STATUS), 'Host Name not MTCE DNS Stds compliant (Full Match)')

-- WHERE NOT REGEXP\_LIKE(AUD.HOST\_NAME, '[[:alnum:]]{8}(D[[:digit:]]{2,2}|[[:digit:]]{2,2}[0-9EWVR])-[AELRP]-JU-(8216|960X)-[[:digit:]]{2,2}')

-- AND MATCH\_CODE LIKE '%BOTH%';

--

-- COMMIT;

-- check the host name matches MTCE DNS std, else update match\_code = ''NE Only'' and match\_status = ''TID not MTCE Stds compliant''

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET FULL\_MATCH = 'Y'

WHERE REGEXP\_LIKE(AUD.HOST\_NAME, '[[:alnum:]]{8}((D[[:digit:]]{2,2})|([[:digit:]][0-9A-P][0-9WVRE]))-[AELRP]-JU-(8216|960X)-[[:digit:]]{2,2}');

COMMIT;

-- check the host name matches MTCE DNS std, else update match\_code = ''NE Only'' and match\_status = ''TID not MTCE Stds compliant''

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS = NVL2(MATCH\_STATUS, 'Host Name not MTCE DNS or Juniper Stds compliant, '||RTRIM(MATCH\_STATUS), 'Host Name not MTCE DNS or Juniper Stds compliant')

WHERE AUD.FULL\_MATCH != 'Y'

AND AUD.MODEL LIKE '%MX960%'

AND NOT REGEXP\_LIKE(AUD.HOST\_NAME, '[[:alnum:]]{8}((D[[:digit:]]{2,2})|([[:digit:]][0-9A-P][0-9WVRE]))');

COMMIT;

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS = NVL2(MATCH\_STATUS, 'Host Name not MTCE DNS Stds compliant, '||RTRIM(MATCH\_STATUS), 'Host Name not MTCE DNS Stds compliant')

WHERE AUD.FULL\_MATCH != 'Y'

AND AUD.MODEL LIKE '%8216%';

COMMIT;

-- check if TID''s don't match first 6-char match some MTSO CLLI then match\_code = ''NE Only'' and match\_status = ''First six chars of Host Namef do not match any MTSO CLLIs''

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS = NVL2(MATCH\_STATUS, RTRIM(MATCH\_STATUS)||', First six chars of Host Name do not match any MTSO CLLIs', 'First six chars of Host Name do not match any MTSO CLLIs')

WHERE NOT EXISTS (

WITH

CLLI\_DOMAINS AS

(

SELECT DISTINCT SUBSTR(CLLI, 1,6) CLLI\_6

FROM CLLI\_DOMAIN\_MAP\_V

--WHERE UPPER(REGION) != 'NNO'

)

SELECT 1

FROM CLLI\_DOMAINS C

WHERE C.CLLI\_6 = SUBSTR(AUD.HOST\_NAME, 1,6)

);

COMMIT;

-- check if TID''s match first 8-char match some MTSO CLLI then set match\_code = ''BOTH'' temporarily

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET AUD.MATCH8 = '1'

WHERE EXISTS (

WITH

CLLI\_DOMAINS as

(

select distinct clli--, substr(clli, 1, 6) clli\_6

FROM CLLI\_DOMAIN\_MAP\_V

--WHERE UPPER(REGION) != 'NNO'

)

select 1

FROM CLLI\_DOMAINS C

WHERE C.CLLI = SUBSTR(AUD.HOST\_NAME, 1,8)

);

COMMIT;

-- check if TID''s match first 6-char match some MTSO CLLI then match\_code = ''NE Only'' and match\_status = ''Only 1st 6-char of CLLI match MTSO''

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS = NVL2(MATCH\_STATUS, RTRIM(MATCH\_STATUS)||', Only first six chars of Host Name match a MTSO CLLI', 'Only first six chars of Host Name match a MTSO CLLI')

WHERE EXISTS (

WITH

CLLI\_DOMAINS AS

(

SELECT DISTINCT SUBSTR(CLLI, 1, 6) CLLI\_6

FROM CLLI\_DOMAIN\_MAP\_V

--WHERE UPPER(REGION) != 'NNO'

)

SELECT 1

FROM CLLI\_DOMAINS C

WHERE C.CLLI\_6 = SUBSTR(AUD.HOST\_NAME, 1,6) AND AUD.MATCH8 IS NULL

);

COMMIT;

-- Set Domain name for NNO Equipment

/\*UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET LEAF\_DOMAIN\_NAME = 'NNO\_DOMAIN'

WHERE REGEXP\_LIKE(AUD.TID, '[[:alnum:]]{8}(MS|OS)[[:digit:]]-NFW[[:digit:]]{5}');\*/

-- insert Granite Only. These will only be the top-level equip NE\_inst\_id ONLY

-- these will be shown in details, but won't affect the summary

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP (NE\_INST\_ID, DESCR, MODEL, VENDOR, MATCH\_CODE, MATCH\_STATUS)

SELECT NE\_INST\_ID, DESCR, MODEL, VENDOR, 'Granite Only', 'No match in NE'

FROM (

SELECT NE\_INST\_ID, DESCR, MODEL, VENDOR

FROM JUNIPER\_XNG\_EQUIP JXE

MINUS

SELECT NE\_INST\_ID, DESCR, MODEL, VENDOR

FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD

);

COMMIT;

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP SET MATCH\_STATUS = RTRIM(MATCH\_STATUS)||', Granite name not MTCE DNS or Juniper Stds compliant'

WHERE MODEL LIKE '%MX960%'

AND NOT (REGEXP\_LIKE(DESCR, '[[:alnum:]]{8}((D[[:digit:]]{2,2})|([[:digit:]][0-9A-P][0-9WVRE]))-[AELRP]-JU-(8216|960X)-[[:digit:]]{2,2}') OR -- granite std name

REGEXP\_LIKE(DESCR, '[[:alnum:]]{8}((D[[:digit:]]{2,2})|([[:digit:]][0-9A-P][0-9WVRE]))'))

AND MATCH\_CODE = 'Granite Only';

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP SET MATCH\_STATUS = RTRIM(MATCH\_STATUS)||', Granite name not MTCE DNS Stds compliant'

WHERE MODEL LIKE '%8216%'

AND NOT REGEXP\_LIKE(DESCR, '[[:alnum:]]{8}((D[[:digit:]]{2,2})|([[:digit:]][0-9A-P][0-9WVRE]))-[AELRP]-JU-(8216|960X)-[[:digit:]]{2,2}') -- granite std name

AND MATCH\_CODE = 'Granite Only';

COMMIT;

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET MATCH\_STATUS = RTRIM(MATCH\_STATUS);

COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK';

-- Move records into WK table where we already know the domain name.

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK

SELECT LEAF\_DOMAIN\_NAME, HOST\_NAME, DEVICE\_IP, DEVICE\_ID,

DEVICE\_VENDOR, DEVICE\_MODEL, DEVICE\_PARTITION, MATCH\_CODE,

MATCH\_STATUS, NE\_INST\_ID, DESCR, MODEL,

VENDOR

FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE LEAF\_DOMAIN\_NAME IS NOT NULL;

DELETE FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE LEAF\_DOMAIN\_NAME IS NOT NULL;

COMMIT;

-- Move Granite\_Only records with a Granite Mapping.

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

WHERE

NE.HOST\_NAME IS NULL;

DELETE FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE NE\_INST\_ID IN (SELECT DISTINCT

NE.NE\_INST\_ID

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

WHERE

NE.HOST\_NAME IS NULL);

COMMIT;

-- Insert NE\_Only records with Granite Mapping.

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

;

DELETE FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE NE\_INST\_ID IN (SELECT DISTINCT

NE.NE\_INST\_ID

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID));

COMMIT;

-- The only records left are those TIDs that do not match the CLLI DOMAIN MAP and have no match in Granite.

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK

SELECT LEAF\_DOMAIN\_NAME, HOST\_NAME, DEVICE\_IP, DEVICE\_ID,

DEVICE\_VENDOR, DEVICE\_MODEL, DEVICE\_PARTITION, MATCH\_CODE,

MATCH\_STATUS, NE\_INST\_ID, DESCR, MODEL,

VENDOR

FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP NE;

--DELETE FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP';

COMMIT;

-- Try to match unknown records in the work table against the CLLI Domain Map view using 8 character CLLI matching

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK NE

SET NE.LEAF\_DOMAIN\_NAME = (SELECT CDM.LEAF\_DOMAIN\_NAME FROM CLLI\_DOMAIN\_MAP\_V CDM WHERE SUBSTR(NE.HOST\_NAME,1,8) = SUBSTR(CDM.CLLI,1,8))

WHERE

NE.LEAF\_DOMAIN\_NAME IS NULL;

COMMIT;

-- Try to match unknown records in the work table against the CLLI Domain Map view using 6 char CLLI matching

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK NE

SET NE.LEAF\_DOMAIN\_NAME = (SELECT DISTINCT CDM.LEAF\_DOMAIN\_NAME FROM CLLI\_DOMAIN\_MAP\_V CDM WHERE SUBSTR(NE.HOST\_NAME,1,6) = SUBSTR(CDM.CLLI,1,6))

WHERE

NE.LEAF\_DOMAIN\_NAME IS NULL;

COMMIT;

-- Try to match unknown records in the work table against the CLLI Domain Map view using 6 char CLLI matching

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK NE

SET NE.LEAF\_DOMAIN\_NAME = (SELECT DISTINCT CDM.LEAF\_DOMAIN\_NAME FROM CLLI\_DOMAIN\_MAP\_V CDM WHERE REGEXP\_LIKE(NE.DESCR,SUBSTR(CDM.CLLI,1,8), 'i'))

WHERE

NE.LEAF\_DOMAIN\_NAME IS NULL

AND (SELECT DISTINCT CDM.LEAF\_DOMAIN\_NAME FROM CLLI\_DOMAIN\_MAP\_V CDM WHERE REGEXP\_LIKE(NE.DESCR,SUBSTR(CDM.CLLI,1,8), 'i')) IS NOT NULL;

COMMIT;

-- Move 3xW NE to the NNO Domain

UPDATE JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK NE

SET NE.LEAF\_DOMAIN\_NAME = 'NNO\_DOMAIN'

WHERE

REGEXP\_LIKE(NE.HOST\_NAME,'^[[:alnum:]]{8}3[0-9A-P]W', 'i')

OR REGEXP\_LIKE(NE.DESCR,'[[:alnum:]]{8}3[0-9A-P]W', 'i');

COMMIT;

END;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure GenSummary \*

\* \*

\* Purpose: Realize summary report \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE GENSUMMARY IS

BEGIN

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_EQ\_SUMM\_REGION\_WK';

-- generate summary

INSERT INTO JUNIPER\_EQ\_SUMM\_REGION\_WK (AREA, REGION, TOTAL\_NE, DUPLICATE\_NE, MATCHED\_NE, PERCENT\_MATCHED, PASSED\_AUDIT, NAMING\_COMPLIANCE/\*, DISCOVERED\_XCONNECTS, LIVE\_XCONNECTS, OTHER\_XCONNECTS, TOTAL\_COMPLIANCE\*/)

WITH DEVICEMETRICS AS

(

SELECT

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

HOST\_NAME,

(CASE (COUNT(HOST\_NAME) - 1) WHEN 0 THEN 0 ELSE 1 END) HAS\_DUPLICATES,

MAX(DECODE(NE\_INST\_ID, NULL, 0, 1)) NE\_MATCHED,

MAX(DECODE(UPPER(TRIM(MATCH\_CODE)), 'BOTH', 1, 0)) PASSED\_AUDIT

--MAX(NUM\_XCONNECT) NUM\_XCONNECT,

--MAX(LIVE\_XCONNECTS) LIVE\_XCONNECTS,

--MAX(OTHER\_XCONNECTS) OTHER\_XCONNECTS

FROM

TABLE(XNG\_REPORTS.JUNIPER\_AUDIT.DETAIL\_TBL\_WRK(1, 10000))

WHERE

HOST\_NAME IS NOT NULL

GROUP BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

HOST\_NAME

),

DOMAINS AS

(

SELECT

AREA,

REGION

FROM

DOMAINS\_REGIONAL\_REPORTING DRR

WHERE

DRR.AREA <> 'OSS'

UNION

SELECT

'unknown' AREA,

'unknown' REGION

FROM

DUAL

)

SELECT

DRR.AREA

, DRR.REGION

, NVL(COUNT(DM.HOST\_NAME),0) TOTAL\_NE

, NVL(SUM(DM.HAS\_DUPLICATES), 0) DUPLICATE\_NE

, NVL(SUM(DM.NE\_MATCHED), 0) MATCHED\_NE

, (CASE SUM(DECODE(NVL(DM.NE\_MATCHED, 0),0,0,1)) WHEN 0 THEN ' 0.00' ELSE (TO\_CHAR(ROUND((SUM(DECODE(NVL(DM.NE\_MATCHED, 0),0,0,1)))/COUNT(DM.HOST\_NAME)\*100,2),'990.00')) END) PERCENT\_MATCHED

, NVL(SUM(DM.PASSED\_AUDIT), 0) PASSED\_AUDIT

, (CASE NVL(COUNT(DM.HOST\_NAME), 0) WHEN 0 THEN ' 0.00' ELSE (TO\_CHAR(ROUND((SUM(DM.PASSED\_AUDIT)/COUNT(DM.HOST\_NAME)\*100),2),'990.00')) END) NAMING\_COMPLIANCE

--, NVL(SUM(DM.NUM\_XCONNECT), 0) DISCOVERED\_XCONNECTS

--, NVL(SUM(DM.LIVE\_XCONNECTS), 0) LIVE\_XCONNECTS

--, NVL(SUM(DM.OTHER\_XCONNECTS), 0) OTHER\_XCONNECTS

--, CASE

-- WHEN NVL(COUNT(DM.TID),0) + NVL(SUM(DM.NUM\_XCONNECT), 0) > 0

-- THEN

-- TO\_CHAR(ROUND(((NVL(SUM(DM.PASSED\_AUDIT), 0) + NVL(SUM(DM.LIVE\_XCONNECTS), 0) + NVL(SUM(DM.OTHER\_XCONNECTS), 0))/(NVL(COUNT(DM.TID),0) + NVL(SUM(DM.NUM\_XCONNECT), 0)))\*100,2),'990.00')

-- ELSE

-- ' 0.00'

-- END TOTAL\_COMPLIANCE

FROM

DOMAINS DRR

LEFT OUTER JOIN

DEVICEMETRICS DM ON DM.REGION = DRR.REGION

GROUP BY

ROLLUP(DRR.AREA, DRR.REGION)

ORDER BY

AREA,

REGION;

COMMIT;

END GENSUMMARY;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Truncate\_WK \*

\* \*

\* Purpose: Truncate work tables. \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE TRUNCATE\_WK IS

BEGIN

-- changed all truncates to deletes

--delete from juniper\_eq\_summ\_region\_wk;

--

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_EQ\_SUMM\_REGION\_WK';

DBMS\_OUTPUT.PUT\_LINE('Deleted from juniper\_eq\_summ\_region\_wk');

-- changed all truncates to deletes

--delete from juniper\_ne\_vs\_xng\_audit\_wk;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK';

DBMS\_OUTPUT.PUT\_LINE('Deleted from juniper\_ne\_vs\_xng\_audit\_wk ');

END TRUNCATE\_WK;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Move\_Data\_From\_WK \*

\* \*

\* Purpose: Copy work tables to production tables. \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE MOVE\_DATA\_FROM\_WK IS

BEGIN

--delete from juniper\_eq\_summ\_region;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_EQ\_SUMM\_REGION';

insert into juniper\_eq\_summ\_region select \* from juniper\_eq\_summ\_region\_wk;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating juniper\_eq\_summ\_region');

--delete from juniper\_ne\_vs\_xng\_audit;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_NE\_VS\_XNG\_AUDIT';

insert into juniper\_ne\_vs\_xng\_audit select \* from juniper\_ne\_vs\_xng\_audit\_wk;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating juniper\_ne\_vs\_xng\_audit');

--delete from juniper\_xconnect\_audit;

--EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_XCONNECT\_AUDIT';

--

--insert into juniper\_xconnect\_audit select \* from juniper\_xconnect\_audit\_wk;

--

--COMMIT;

--DBMS\_OUTPUT.PUT\_LINE('Done populating juniper\_xconnect\_audit');

END MOVE\_DATA\_FROM\_WK;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Move\_Data\_To\_WK \*

\* \*

\* Purpose: Copy production tables to work tables. \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE MOVE\_DATA\_TO\_WK IS

BEGIN

--DELETE FROM JUNIPER\_EQ\_SUMM\_REGION\_WK;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_EQ\_SUMM\_REGION\_WK';

insert into juniper\_eq\_summ\_region\_wk select \* from juniper\_eq\_summ\_region;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating juniper\_eq\_summ\_region\_wk');

--DELETE FROM JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK';

INSERT INTO JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK SELECT \* FROM JUNIPER\_NE\_VS\_XNG\_AUDIT;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating juniper\_ne\_vs\_xng\_audit\_wk');

--delete from juniper\_xconnect\_audit\_wk;

--EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_XCONNECT\_AUDIT\_WK';

--insert into juniper\_xconnect\_audit\_wk select \* from juniper\_xconnect\_audit;

--COMMIT;

--DBMS\_OUTPUT.PUT\_LINE('Done populating juniper\_xconnect\_audit\_wk');

END MOVE\_DATA\_TO\_WK;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Audit\_XConnects \*

\* \*

\* Purpose: Audit FlashWave cross connects against XNG \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*PROCEDURE AUDIT\_XCONNECTS IS

BEGIN

--DELETE FROM JUNIPER\_FUJITSU\_EQUIPMENT;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_FUJITSU\_EQUIPMENT';

-- INSERT INTO JUNIPER\_FUJITSU\_EQUIPMENT

-- WITH FUJITSU\_EQUIPMENT AS

-- (

-- SELECT

-- NE.TID,

-- NE.TID\_TYPE,

-- XFE.NE\_INST\_ID,

-- EI.EQUIP\_INST\_ID,

-- EI.SITE\_INST\_ID,

-- EI.PARENT\_EQ\_INST\_ID,

-- EI.VENDOR,

-- EI.MODEL,

-- EI.DESCR

-- FROM

-- VZWNET.EQUIP\_INST EI

-- JOIN

-- XNG\_REPORTS.JUNIPER\_SNCNE NE ON REGEXP\_LIKE(EI.DESCR, NE.TID)

-- JOIN

-- XNG\_REPORTS.XNG\_FW\_EQUIP XFE ON XFE.EQUIP\_INST\_ID = EI.EQUIP\_INST\_ID

-- WHERE

-- EI.VENDOR = 'FUJITSU'

-- AND REGEXP\_LIKE(EI.MODEL, '(FLASHWAVE|FW)')

-- )

-- SELECT

-- \*

-- FROM

-- FUJITSU\_EQUIPMENT NE

-- ORDER BY

-- NE.TID;

-- COMMIT;

INSERT INTO JUNIPER\_XCONNECT\_EQUIPMENT

SELECT

LEVEL NE\_LEVEL,

TO\_NUMBER

(TRIM

(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH(

DECODE (LEVEL, 1, EI.EQUIP\_INST\_ID, '' ),

'/'

)

)

) NE\_INST\_ID,

EI.EQUIP\_INST\_ID,

EI.SITE\_INST\_ID,

EI.PARENT\_EQ\_INST\_ID,

DECODE (LEVEL,

1, 'NE',

DECODE (EI.EQ\_CLASS, 'S', 'SHELF', 'N/A')

) EQ\_CLASS\_TYPE,

EI.DESCR,

EI.MODEL,

EI.TYPE,

EI.STATUS,

EI.VENDOR,

EI.EQ\_CLASS

FROM

VZWNET.EQUIP\_INST EI

WHERE

(LEVEL = 1 OR EI.EQ\_CLASS = 'S')

START WITH

EI.VENDOR = 'FUJITSU'

AND REGEXP\_LIKE (MODEL, 'FLASHWAVE|FW')

CONNECT BY

PRIOR EI.EQUIP\_INST\_ID = EI.PARENT\_EQ\_INST\_ID;

COMMIT;

INSERT INTO JUNIPER\_FUJITSU\_EQUIPMENT

SELECT

NE.TID,

NE.TID\_TYPE,

NXE.NE\_LEVEL,

NXE.NE\_INST\_ID,

NXE.EQUIP\_INST\_ID,

NXE.SITE\_INST\_ID,

NXE.PARENT\_EQ\_INST\_ID,

NXE.EQ\_CLASS\_TYPE,

NXE.DESCR,

NXE.MODEL,

NXE.TYPE,

NXE.STATUS,

NXE.VENDOR,

NXE.EQ\_CLASS

FROM

JUNIPER\_XCONNECT\_EQUIPMENT NXE

JOIN

JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK NE ON (NE.NE\_INST\_ID = NXE.NE\_INST\_ID)

WHERE

NE.TID IS NOT NULL;

COMMIT;

UPDATE JUNIPER\_FUJITSU\_EQUIPMENT

SET PARENT\_EQ\_INST\_ID = NULL

WHERE EQUIP\_INST\_ID IN

(SELECT

NE2.EQUIP\_INST\_ID

FROM

JUNIPER\_FUJITSU\_EQUIPMENT NE2

WHERE

NE2.PARENT\_EQ\_INST\_ID IS NOT NULL

AND NOT EXISTS (SELECT

NE.EQUIP\_INST\_ID

FROM

JUNIPER\_FUJITSU\_EQUIPMENT NE

WHERE

NE.EQUIP\_INST\_ID = NE2.PARENT\_EQ\_INST\_ID

)

);

COMMIT;

--DELETE FROM JUNIPER\_XNG\_PORTS;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_XNG\_PORTS';

INSERT INTO JUNIPER\_XNG\_PORTS

SELECT DISTINCT

NE.TID,

NE.TID\_TYPE,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-') SLOT,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '') PORT\_HUM\_ID,

EP.PORT\_ACCESS\_ID,

CPE.CIRC\_PATH\_INST\_ID,

LI.LEG\_INST\_ID,

CPE.MEMBER\_NBR,

CPI.A\_SIDE\_SITE\_ID,

S1.SITE\_HUM\_ID A\_SIDE\_SITE\_HUM\_ID,

CPI.Z\_SIDE\_SITE\_ID,

S2.SITE\_HUM\_ID Z\_SIDE\_SITE\_HUM\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.BANDWIDTH,

CPI.STATUS PATH\_STATUS,

CPI.NEXT\_PATH\_INST\_ID,

CPI.PREV\_PATH\_INST\_ID,

NE.NE\_INST\_ID,

NE.EQUIP\_INST\_ID,

NE.PARENT\_EQ\_INST\_ID,

NE.SITE\_INST\_ID,

NE.VENDOR,

NE.MODEL,

NE.DESCR,

EP.PORT\_INST\_ID,

EP.CARD\_INST\_ID

FROM

JUNIPER\_FUJITSU\_EQUIPMENT NE

JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.EQUIP\_INST\_ID)

JOIN

VZWNET.EPA EP ON (EP.EQUIP\_INST\_ID = EQ.EQUIP\_INST\_ID)

JOIN

VZWNET.CIRC\_PATH\_ELEMENT CPE ON CPE.PORT\_INST\_ID = EP.PORT\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_INST LI ON LI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_MEMBER PLM ON PLM.LEG\_INST\_ID = LI.LEG\_INST\_ID AND PLM.ELEMENT\_INST\_ID = CPE.ELEMENT\_INST\_ID

JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.SITE\_INST S1 ON S1.SITE\_INST\_ID = CPI.A\_SIDE\_SITE\_ID

JOIN

VZWNET.SITE\_INST S2 ON S2.SITE\_INST\_ID = CPI.Z\_SIDE\_SITE\_ID

JOIN

VZWNET.CARD\_INST CRD ON (CRD.CARD\_INST\_ID = EP.CARD\_INST\_ID)

WHERE

NE.TID IS NOT NULL

AND REGEXP\_LIKE(EP.PORT\_HUM\_ID, '-')

AND UPPER(CPI.STATUS) = 'LIVE'

ORDER BY

TID,

PORT\_HUM\_ID;

COMMIT;

---------------------------------------------------------------

-- Insert ports where a "live" port does not already exist --

---------------------------------------------------------------

INSERT INTO JUNIPER\_XNG\_PORTS

SELECT DISTINCT

NE.TID,

NE.TID\_TYPE,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-') SLOT,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '') PORT\_HUM\_ID,

EP.PORT\_ACCESS\_ID,

CPE.CIRC\_PATH\_INST\_ID,

LI.LEG\_INST\_ID,

CPE.MEMBER\_NBR,

CPI.A\_SIDE\_SITE\_ID,

S1.SITE\_HUM\_ID A\_SIDE\_SITE\_HUM\_ID,

CPI.Z\_SIDE\_SITE\_ID,

S2.SITE\_HUM\_ID Z\_SIDE\_SITE\_HUM\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.BANDWIDTH,

CPI.STATUS PATH\_STATUS,

CPI.NEXT\_PATH\_INST\_ID,

CPI.PREV\_PATH\_INST\_ID,

NE.NE\_INST\_ID,

NE.EQUIP\_INST\_ID,

NE.PARENT\_EQ\_INST\_ID,

NE.SITE\_INST\_ID,

NE.VENDOR,

NE.MODEL,

NE.DESCR,

EP.PORT\_INST\_ID,

EP.CARD\_INST\_ID

FROM

JUNIPER\_FUJITSU\_EQUIPMENT NE

JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.EQUIP\_INST\_ID OR EQ.PARENT\_EQ\_INST\_ID = NE.EQUIP\_INST\_ID)

JOIN

VZWNET.EPA EP ON (EP.EQUIP\_INST\_ID = EQ.EQUIP\_INST\_ID)

JOIN

VZWNET.CIRC\_PATH\_ELEMENT CPE ON CPE.PORT\_INST\_ID = EP.PORT\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_INST LI ON LI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_MEMBER PLM ON PLM.LEG\_INST\_ID = LI.LEG\_INST\_ID AND PLM.ELEMENT\_INST\_ID = CPE.ELEMENT\_INST\_ID

JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.SITE\_INST S1 ON S1.SITE\_INST\_ID = CPI.A\_SIDE\_SITE\_ID

JOIN

VZWNET.SITE\_INST S2 ON S2.SITE\_INST\_ID = CPI.Z\_SIDE\_SITE\_ID

JOIN

VZWNET.CARD\_INST CRD ON (CRD.CARD\_INST\_ID = EP.CARD\_INST\_ID)

WHERE

NE.TID IS NOT NULL

AND REGEXP\_LIKE(EP.PORT\_HUM\_ID, '-')

AND UPPER(CPI.STATUS) != 'LIVE'

AND NOT EXISTS (SELECT 1

FROM

XNG\_REPORTS.JUNIPER\_XNG\_PORTS NXP

WHERE

NXP.TID = NE.TID

--AND NXP.SLOT = REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-')

AND NXP.PORT\_HUM\_ID = REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '')

)

ORDER BY

TID,

PORT\_HUM\_ID;

COMMIT;

--DELETE FROM JUNIPER\_XCONNECT\_AUDIT\_WK;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE JUNIPER\_XCONNECT\_AUDIT\_WK';

INSERT INTO JUNIPER\_XCONNECT\_AUDIT\_WK

SELECT

NE.TID,

NE.TID\_TYPE,

NULL MATCH\_CODE,

NULL MATCH\_STATUS,

XC.XCONNECT\_ASIDE,

XP1.NE\_INST\_ID A\_SIDE\_EQUIP\_ID,

XP1.CIRC\_PATH\_INST\_ID A\_SIDE\_PATH\_INST\_ID,

XP1.LEG\_INST\_ID A\_SIDE\_LEG\_INST\_ID,

XP1.MEMBER\_NBR A\_SIDE\_MEMBER\_NBR,

XC.XCONNECT\_ZSIDE,

XP2.NE\_INST\_ID Z\_SIDE\_EQUIP\_ID,

XP1.CIRC\_PATH\_INST\_ID Z\_SIDE\_PATH\_INST\_ID,

XP2.LEG\_INST\_ID A\_SIDE\_LEG\_INST\_ID,

XP2.MEMBER\_NBR Z\_SIDE\_MEMBER\_NBR,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.CIRC\_PATH\_INST\_ID

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.CIRC\_PATH\_INST\_ID

ELSE NULL

END CIRC\_PATH\_INST\_ID,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.CIRC\_PATH\_HUM\_ID

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.CIRC\_PATH\_HUM\_ID

ELSE NULL

END CIRC\_PATH\_HUM\_ID,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.PATH\_STATUS

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.PATH\_STATUS

ELSE NULL

END PATH\_STATUS,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.BANDWIDTH

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.BANDWIDTH

ELSE NULL

END BANDWIDTH

FROM

JUNIPER\_SNCNE NE

JOIN

JUNIPER\_XCONNECT XC ON XC.SNCNE\_ID = NE.INST\_ID

LEFT OUTER JOIN

JUNIPER\_XNG\_PORTS XP1 ON XP1.TID = NE.TID AND (((XP1.PORT\_HUM\_ID = XC.XCONNECT\_ASIDE) OR (XP1.SLOT || '-' || XP1.PORT\_HUM\_ID = XC.XCONNECT\_ASIDE)) OR

(XP1.PORT\_ACCESS\_ID = XC.XCONNECT\_ASIDE))

LEFT OUTER JOIN

JUNIPER\_XNG\_PORTS XP2 ON XP2.TID = NE.TID AND (((XP2.PORT\_HUM\_ID = XC.XCONNECT\_ZSIDE) OR (XP2.SLOT || '-' || XP2.PORT\_HUM\_ID = XC.XCONNECT\_ZSIDE)) OR

(XP2.PORT\_ACCESS\_ID = XC.XCONNECT\_ZSIDE))

-- WHERE

-- XP1.A\_SIDE\_SITE\_ID = XP2.Z\_SIDE\_SITE\_ID

-- REGEXP\_LIKE(NE.TID\_TYPE, '9500')

ORDER BY

NE.TID,

XCONNECT\_ASIDE,

XCONNECT\_ZSIDE;

COMMIT;

-- Mark the audit rows where neither side was found

UPDATE

XNG\_REPORTS.JUNIPER\_XCONNECT\_AUDIT\_WK

SET

MATCH\_CODE = 'DISCREPANCY',

MATCH\_STATUS = 'Neither of the two cross connected ports have an associated path in Granite.'

WHERE

A\_SIDE\_LEG\_INST\_ID IS NULL AND Z\_SIDE\_LEG\_INST\_ID IS NULL;

COMMIT;

-- Mark the audit rows where A side was found but the Z side wasn't

UPDATE

XNG\_REPORTS.JUNIPER\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'One of the cross connected ports is missing from the path in Granite. ' || XC.XCONNECT\_ZSIDE || ' not found in the circuit path. Remove cross connect from the MSPP or correct circuit path elements in Granite.'

WHERE

XC.MATCH\_STATUS IS NULL AND

A\_SIDE\_LEG\_INST\_ID IS NOT NULL AND Z\_SIDE\_LEG\_INST\_ID IS NULL;

COMMIT;

-- Mark the audit rows where Z side was found but the A side wasn't

UPDATE

XNG\_REPORTS.JUNIPER\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'One of the cross connected ports is missing from the path in Granite. ' || XC.XCONNECT\_ASIDE || ' not found in the circuit path. Remove cross connect from the MSPP or correct circuit path elements in Granite.'

WHERE

XC.MATCH\_STATUS IS NULL AND

A\_SIDE\_LEG\_INST\_ID IS NULL AND Z\_SIDE\_LEG\_INST\_ID IS NOT NULL;

COMMIT;

-- Mark the rows where the path\_inst ids do not match

UPDATE

XNG\_REPORTS.JUNIPER\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The A side port and Z side port of the cross connect do not belong to the same Circuit Path.'

WHERE

XC.MATCH\_CODE IS NULL

AND A\_SIDE\_PATH\_INST\_ID != Z\_SIDE\_PATH\_INST\_ID;

COMMIT;

-- Mark the rows where the leg\_inst ids do not match

UPDATE

XNG\_REPORTS.JUNIPER\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The A side port and Z side port are not endpoints of a single circuit leg.'

WHERE

XC.MATCH\_CODE IS NULL

AND A\_SIDE\_LEG\_INST\_ID != Z\_SIDE\_LEG\_INST\_ID;

COMMIT;

-- Mark the rows where the leg\_inst match but the member numbers do not match

UPDATE

XNG\_REPORTS.JUNIPER\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The member number of the A side port does not match the member number of the Z side port.'

WHERE

A\_SIDE\_LEG\_INST\_ID = Z\_SIDE\_LEG\_INST\_ID

AND ((A\_SIDE\_MEMBER\_NBR != Z\_SIDE\_MEMBER\_NBR

OR (A\_SIDE\_MEMBER\_NBR IS NULL AND Z\_SIDE\_MEMBER\_NBR IS NOT NULL)

or (A\_SIDE\_MEMBER\_NBR IS NOT NULL AND Z\_SIDE\_MEMBER\_NBR IS NULL)));

COMMIT;

-- Mark the rest of the Audit records as passed audit

UPDATE

XNG\_REPORTS.JUNIPER\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'MATCH',

XC.MATCH\_STATUS = ''

WHERE

A\_SIDE\_LEG\_INST\_ID = Z\_SIDE\_LEG\_INST\_ID

AND (A\_SIDE\_MEMBER\_NBR = Z\_SIDE\_MEMBER\_NBR or (A\_SIDE\_MEMBER\_NBR IS NULL AND Z\_SIDE\_MEMBER\_NBR IS NULL));

COMMIT;

END AUDIT\_XCONNECTS;\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Function Summary\_Tbl \*

\* \*

\* Purpose: Outputs Juniper Summary Report Table \*

\* \*

\* Inputs: None \*

\* Outputs: Juniper\_Summary\_Tbl \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION SUMMARY\_TBL RETURN JUNIPER\_SUMMARY\_TBL PIPELINED IS

OUT\_REC JUNIPER\_SUMMARY\_REC := JUNIPER\_SUMMARY\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR JUNIPER\_CUR IS

SELECT

AREA,

REGION,

TOTAL\_NE,

DUPLICATE\_NE,

MATCHED\_NE,

PERCENT\_MATCHED,

PASSED\_AUDIT,

NAMING\_COMPLIANCE

FROM

JUNIPER\_EQ\_SUMM\_REGION;

BEGIN

OPEN JUNIPER\_CUR;

LOOP

FETCH JUNIPER\_CUR INTO OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.TOTAL\_NE, OUT\_REC.DUPLICATE\_NE, OUT\_REC.MATCHED\_NE, OUT\_REC.PERCENT\_MATCHED, OUT\_REC.PASSED\_AUDIT,

OUT\_REC.NAMING\_COMPLIANCE;

EXIT WHEN JUNIPER\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE JUNIPER\_CUR;

RETURN;

END SUMMARY\_TBL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function DETAIL\_TBL \*

\* \*

\* Parameters \*

\* REGION\_IN Region Name (from CLLI\_DOMAIN\_MAP) \*

\* use NULL for all regions. \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION DETAIL\_TBL(REGION\_IN IN VARCHAR2, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN JUNIPER\_DETAIL\_TBL PIPELINED IS

OUT\_REC JUNIPER\_DETAIL\_REC := JUNIPER\_DETAIL\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR ALL\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY AREA, REGION, LEAF\_DOMAIN\_NAME, HOST\_NAME, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

HOST\_NAME,

DEVICE\_IP,

DEVICE\_ID,

DEVICE\_VENDOR,

DEVICE\_MODEL,

DEVICE\_PARTITION,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

MODEL,

VENDOR

FROM (SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

S.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.MODEL,

EQ.VENDOR

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

WHERE

CDM.REGION IS NOT NULL

UNION

SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

S.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.MODEL,

EQ.VENDOR

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

WHERE

CDM.REGION IS NULL

)

ORDER BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

HOST\_NAME,

DESCR

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR UNKNOWN\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY LEAF\_DOMAIN\_NAME, HOST\_NAME, DESCR)) ROWNUMBER,

'unknown' AREA,

'unknown' REGION,

'UNKNOWN\_DOMAIN' LEAF\_DOMAIN\_NAME,

HOST\_NAME,

DEVICE\_IP,

DEVICE\_ID,

DEVICE\_VENDOR,

DEVICE\_MODEL,

DEVICE\_PARTITION,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

MODEL,

VENDOR

FROM (SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.MODEL,

EQ.VENDOR

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

WHERE

REGION IS NULL

ORDER BY

LEAF\_DOMAIN\_NAME,

HOST\_NAME,

DESCR

)

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR REGION\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY LEAF\_DOMAIN\_NAME, HOST\_NAME, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

HOST\_NAME,

DEVICE\_IP,

DEVICE\_ID,

DEVICE\_VENDOR,

DEVICE\_MODEL,

DEVICE\_PARTITION,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

MODEL,

VENDOR

FROM (SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.MODEL,

EQ.VENDOR

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

WHERE

REGION = REGION\_IN

ORDER BY

LEAF\_DOMAIN\_NAME,

HOST\_NAME,

DESCR

)

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

IF (REGION\_IN IS NULL) THEN

OPEN ALL\_CUR;

LOOP

FETCH ALL\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.HOST\_NAME, OUT\_REC.DEVICE\_IP, OUT\_REC.DEVICE\_ID, OUT\_REC.DEVICE\_VENDOR,

OUT\_REC.DEVICE\_MODEL, OUT\_REC.DEVICE\_PARTITION, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID, OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID,

OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.MODEL, OUT\_REC.VENDOR ;

EXIT WHEN ALL\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE ALL\_CUR;

ELSIF (UPPER(REGION\_IN) = 'UNKNOWN') THEN

OPEN UNKNOWN\_CUR;

LOOP

FETCH UNKNOWN\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.HOST\_NAME, OUT\_REC.DEVICE\_IP, OUT\_REC.DEVICE\_ID, OUT\_REC.DEVICE\_VENDOR,

OUT\_REC.DEVICE\_MODEL, OUT\_REC.DEVICE\_PARTITION, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID, OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID,

OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.MODEL, OUT\_REC.VENDOR;

EXIT WHEN UNKNOWN\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE UNKNOWN\_CUR;

ELSE

OPEN REGION\_CUR;

LOOP

FETCH REGION\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.HOST\_NAME, OUT\_REC.DEVICE\_IP, OUT\_REC.DEVICE\_ID, OUT\_REC.DEVICE\_VENDOR,

OUT\_REC.DEVICE\_MODEL, OUT\_REC.DEVICE\_PARTITION, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID, OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID,

OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.MODEL, OUT\_REC.VENDOR;

EXIT WHEN REGION\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE REGION\_CUR;

END IF ;

RETURN;

END DETAIL\_TBL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function DETAIL\_TBL\_WRK (Internal) \*

\* \*

\* Parameters \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION DETAIL\_TBL\_WRK(START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN JUNIPER\_DETAIL\_TBL PIPELINED IS

OUT\_REC JUNIPER\_DETAIL\_REC := JUNIPER\_DETAIL\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR ALL\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY AREA, REGION, LEAF\_DOMAIN\_NAME, HOST\_NAME, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

HOST\_NAME,

DEVICE\_IP,

DEVICE\_ID,

DEVICE\_VENDOR,

DEVICE\_MODEL,

DEVICE\_PARTITION,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

MODEL,

VENDOR

FROM (

SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.MODEL,

EQ.VENDOR

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK NE

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

WHERE

NE.LEAF\_DOMAIN\_NAME IS NOT NULL

-- AND UPPER(REGION) <> 'NNO'

GROUP BY

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.MODEL,

EQ.VENDOR

UNION

SELECT

'unknown' AREA,

'unknown' REGION,

NULL LEAF\_DOMAIN\_NAME,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.MODEL,

EQ.VENDOR

FROM

JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK NE

LEFT OUTER JOIN

DOMAINS CDM ON ((CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME) AND (UPPER(CDM.REGION) != 'NNO'))

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

WHERE

NE.LEAF\_DOMAIN\_NAME IS NULL

GROUP BY

'unknown',

'unknown',

NULL,

NE.HOST\_NAME,

NE.DEVICE\_IP,

NE.DEVICE\_ID,

NE.DEVICE\_VENDOR,

NE.DEVICE\_MODEL,

NE.DEVICE\_PARTITION,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.MODEL,

EQ.VENDOR

)

ORDER BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

HOST\_NAME,

DESCR

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

OPEN ALL\_CUR;

LOOP

FETCH ALL\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.HOST\_NAME, OUT\_REC.DEVICE\_IP, OUT\_REC.DEVICE\_ID, OUT\_REC.DEVICE\_VENDOR,

OUT\_REC.DEVICE\_MODEL, OUT\_REC.DEVICE\_PARTITION, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID, OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID,

OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.MODEL, OUT\_REC.VENDOR;

EXIT WHEN ALL\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE ALL\_CUR;

RETURN;

END DETAIL\_TBL\_WRK;

END JUNIPER\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package Body LDAP\_XNG\_USER\_FIX

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."LDAP\_XNG\_USER\_FIX"

IS

PROCEDURE find\_uswin\_ids

IS

v\_xng\_id\_hld xng\_reports.ldap\_xng\_mismatch.user\_inst\_id%type;

cursor detail\_cursor is

select distinct uswin\_user\_id, user\_inst\_id, user\_name, full\_name, department,

telephone, email\_addr, comments, flags, modify\_xng, nvl(match\_type,99) as match\_type

from xng\_reports.ldap\_xng\_match\_wrk

where flags <> '0100'

order by user\_inst\_id, match\_type;

detail\_row detail\_cursor%ROWTYPE;

cursor man\_cursor is

select uswin\_user\_id, xng\_fullname

from xng\_reports.ldap\_users\_manual;

man\_row man\_cursor%ROWTYPE;

BEGIN

XNG\_REPORTS.BTP\_XNG\_AUDIT\_TRUNCATES('ldap\_xng\_match\_wrk');

insert into xng\_reports.ldap\_xng\_match\_wrk

(select uswin\_user\_id, user\_inst\_id, user\_name, full\_name,

department, telephone, email\_addr, comments, flags,

case

when uswin\_user\_id <> user\_name and uswin\_user\_id <> 'nfd' then 'Y'

else 'N'

end modify\_xng, nvl(match\_type,99)

from (select nvl(matches.uswin\_user\_id,'nfd') as uswin\_user\_id, alluser.user\_inst\_id, alluser.user\_name, alluser.full\_name,

alluser.department, alluser.telephone, alluser.email\_addr, alluser.comments, alluser.flags, match\_type

from

(select user\_inst\_id, user\_name, full\_name, department, telephone, email\_addr, comments, flags

from xperadmin.user\_inst) alluser,

(

--exact userid

select uswin\_user\_id, xng\_user\_id, user\_inst\_id, 1 as match\_type

from (select rtrim(ltrim(user\_name)) xng\_user\_id, user\_inst\_id from xperadmin.user\_inst) a,

(select uswin\_user\_id from xng\_reports.ldap\_users) b

where a.xng\_user\_id = b.uswin\_user\_id

union

--exact fullname

select uswin\_user\_id, xng\_user\_id, user\_inst\_id, 2 as match\_type

from (select user\_name xng\_user\_id, lower(rtrim(ltrim(full\_name))) full\_name, user\_inst\_id from xperadmin.user\_inst) a,

(select uswin\_user\_id, lower(rtrim(ltrim(uswin\_fullname))) uswin\_fullname from xng\_reports.ldap\_users) b

where a.full\_name = b.uswin\_fullname

union

--find matches by email

select b.uswin\_user\_id, a.xng\_user\_id, user\_inst\_id, 3 as match\_type

from (select rtrim(ltrim(user\_name)) as xng\_user\_id, user\_inst\_id, lower(email\_addr) as xng\_email

from xperadmin.user\_inst

where email\_addr is not null) a,

(select uswin\_user\_id, lower(uswin\_email) as uswin\_email

from xng\_reports.ldap\_users

where uswin\_email is not null

and uswin\_email <> '<unknown>') b

where a.xng\_email = b.uswin\_email

union

--lowercase userid

select uswin\_user\_id, xng\_user\_id, user\_inst\_id, 4 as match\_type

from (select rtrim(ltrim(user\_name)) xng\_user\_id, user\_inst\_id, lower(user\_name) xng\_user\_id\_low from xperadmin.user\_inst) a,

(select uswin\_user\_id, lower(uswin\_user\_id) uswin\_user\_id\_low from xng\_reports.ldap\_users) b

where a.xng\_user\_id\_low = b.uswin\_user\_id\_low

union

--find matches by parsed names no middle name

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 5 as match\_type

from (select xng\_user\_id, user\_inst\_id, department, firstname, lastname

from xng\_reports.user\_inst\_name\_parse

where middlelen = 0) a,

(select uswin\_user\_id, uswin\_dept, lower(uswin\_firstname) firstname, lower(uswin\_lastname) lastname

from xng\_reports.ldap\_users) b

where a.firstname = b.firstname

and a.lastname = b.lastname

union

--find matches by firstname/lastname/dept

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 6 as match\_type

from (select xng\_user\_id, user\_inst\_id, department, firstname, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse) a,

(select uswin\_user\_id, uswin\_dept, lower(uswin\_firstname) as firstname, lower(uswin\_lastname) as lastname

from xng\_reports.ldap\_users) b

where a.department = b.uswin\_dept

and (a.firstname = b.firstname

or a.middlename = b.firstname)

and (a.middlename = b.lastname

or a.lastname = b.lastname)

union

--find matches by firstname/lastname

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 7 as match\_type

from (select xng\_user\_id, user\_inst\_id, firstname, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse) a,

(select uswin\_user\_id, lower(uswin\_firstname) as firstname, lower(uswin\_lastname) as lastname

from xng\_reports.ldap\_users) b

where (a.firstname = b.firstname

or a.middlename = b.firstname)

and (a.middlename = b.lastname

or a.lastname = b.lastname)

union

--find matches by uswin firstname/lastname and name parsed from xng email

select uswin\_user\_id, xng\_user\_id, user\_inst\_id, 8 as match\_type

from (select user\_inst\_id, user\_name as xng\_user\_id, lower(substr(email\_addr, 1, instr(email\_addr,'.',1,1) - 1)) firstname,

lower(substr(email\_addr, (instr(email\_addr,'.',1,1) + 1), (instr(email\_addr,'@',1,1) - instr(email\_addr,'.',1,1) - 1))) lastname

from xperadmin.user\_inst

where email\_addr is not null) a,

(select uswin\_user\_id, lower(uswin\_firstname) as firstname, lower(uswin\_lastname) as lastname

from xng\_reports.ldap\_users) b

where a.firstname = b.firstname

and a.lastname = b.lastname

union

--find matches by xng parsed name to name parsed from uswin email

select b.uswin\_user\_id, a.xng\_user\_id, user\_inst\_id, 9 as match\_type

from

(select xng\_user\_id, user\_inst\_id, firstname, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse) a,

(select uswin\_user\_id, lower(substr(uswin\_email, 1, instr(uswin\_email,'.',1,1) - 1)) firstname,

lower(substr(uswin\_email, (instr(uswin\_email,'.',1,1) + 1), (instr(uswin\_email,'@',1,1) - instr(uswin\_email,'.',1,1) - 1))) lastname

from xng\_reports.ldap\_users

where uswin\_email is not null and uswin\_email <> '<unknown>') b

where (a.firstname = b.firstname

or a.middlename = b.firstname)

and (a.middlename = b.lastname

or a.lastname = b.lastname)

union

--find matches by uswin parsed name to name parsed from xng email

select b.uswin\_user\_id, a.xng\_user\_id, user\_inst\_id, 10 as match\_type

from

(select user\_name as xng\_user\_id, user\_inst\_id, lower(substr(email\_addr, 1, instr(email\_addr,'.',1,1) - 1)) firstname,

lower(substr(email\_addr, (instr(email\_addr,'.',1,1) + 1), (instr(email\_addr,'@',1,1) - instr(email\_addr,'.',1,1) - 1))) lastname

from xperadmin.user\_inst

where email\_addr is not null) a,

(select uswin\_user\_id, firstname, lastname1, lastname2, lastname3, length(lastname1) len1, length(lastname2) len2, length(lastname3) len3

from (select uswin\_user\_id, lower(substr(uswin\_fullname, 1, instr(uswin\_fullname,' ',1,1) - 1)) firstname,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,1) + 1)) lastname1,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,1) + 1, (instr(uswin\_fullname,' ',1,2) - instr(uswin\_fullname,' ',1,1) - 1))) lastname2,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,2) + 1)) lastname3

from (select uswin\_user\_id, replace(replace(replace(replace(uswin\_fullname,'.',''),',',''),'''',''),'"') as uswin\_fullname

from xng\_reports.ldap\_users))) b

where a.firstname = b.firstname

and (a.lastname = b.lastname1

or a.lastname = b.lastname2

or a.lastname = b.lastname3)

union

--find match by stripped user ids

select b.uswin\_user\_id, a.xng\_user\_id, user\_inst\_id, 11 as match\_type

from (select rtrim(ltrim(user\_name)) as xng\_user\_id, user\_inst\_id, lower(substr(full\_name, 1, instr(full\_name,' ',1,1) - 1)) firstname,

replace(replace(replace(replace(replace(replace(replace(replace(replace(replace(user\_name, '1',''),'2',''),'3',''),'4',''),'5',''),'6',''),'7',''),'8',''),'9',''),'0', '') xng\_strp

from xperadmin.user\_inst) a,

(select uswin\_user\_id, lower(substr(uswin\_fullname, 1, instr(uswin\_fullname,' ',1,1) - 1)) firstname,

replace(replace(replace(replace(replace(replace(replace(replace(replace(replace(uswin\_user\_id, '1',''),'2',''),'3',''),'4',''),'5',''),'6',''),'7',''),'8',''),'9',''),'0', '') uswin\_strp

from xng\_reports.ldap\_users) b

where a.xng\_strp = b.uswin\_strp

and a.firstname = b.firstname

union

--find matches by parsed names/dept middlename

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 12 as match\_type

from (select xng\_user\_id, user\_inst\_id, department, firstname, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse

where middlelen <> 0) a,

(select uswin\_user\_id, uswin\_dept, firstname, lastname1, lastname2, lastname3, length(lastname1) len1, length(lastname2) len2, length(lastname3) len3

from (select uswin\_user\_id, uswin\_dept,lower(substr(uswin\_fullname, 1, instr(uswin\_fullname,' ',1,1) - 1)) firstname,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,1) + 1)) lastname1,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,1) + 1, (instr(uswin\_fullname,' ',1,2) - instr(uswin\_fullname,' ',1,1) - 1))) lastname2,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,2) + 1)) lastname3

from (select uswin\_user\_id, uswin\_dept, replace(replace(replace(replace(uswin\_fullname,'.',''),',',''),'''',''),'"') as uswin\_fullname

from xng\_reports.ldap\_users))) b

where a.department = b.uswin\_dept

and (a.firstname = b.firstname and firstlen > 1

or a.middlename = b.firstname and middlelen > 1

or a.firstname = b.lastname1 and firstlen > 1

or a.middlename = b.lastname1 and middlelen > 1)

and (a.middlename = b.lastname1 and middlelen > 1

or a.lastname = b.lastname1 and lastlen > 1

or a.middlename = b.lastname2 and middlelen > 1

or a.lastname = b.lastname2 and lastlen > 1

or a.middlename = b.lastname3 and middlelen > 1

or a.lastname = b.lastname3 and lastlen > 1)

union

--find matches by parsed names middlename

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 13 as match\_type

from (select xng\_user\_id, user\_inst\_id, department, firstname, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse

where middlelen <> 0) a,

(select uswin\_user\_id, uswin\_dept, firstname, lastname1, lastname2, lastname3, length(lastname1) len1, length(lastname2) len2, length(lastname3) len3

from (select uswin\_user\_id, uswin\_dept,lower(substr(uswin\_fullname, 1, instr(uswin\_fullname,' ',1,1) - 1)) firstname,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,1) + 1)) lastname1,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,1) + 1, (instr(uswin\_fullname,' ',1,2) - instr(uswin\_fullname,' ',1,1) - 1))) lastname2,

lower(substr(uswin\_fullname, instr(uswin\_fullname,' ',1,2) + 1)) lastname3

from (select uswin\_user\_id, uswin\_dept, replace(replace(replace(replace(uswin\_fullname,'.',''),',',''),'''',''),'"') as uswin\_fullname

from xng\_reports.ldap\_users))) b

where (a.firstname = b.firstname and firstlen > 1

or a.middlename = b.firstname and middlelen > 1

or a.firstname = b.lastname1 and firstlen > 1

or a.middlename = b.lastname1 and middlelen > 1)

and (a.middlename = b.lastname1 and middlelen > 1

or a.lastname = b.lastname1 and lastlen > 1

or a.middlename = b.lastname2 and middlelen > 1

or a.lastname = b.lastname2 and lastlen > 1

or a.middlename = b.lastname3 and middlelen > 1

or a.lastname = b.lastname3 and lastlen > 1)

union

--match lastname/first 4 of first name

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 14 as match\_type

from (select xng\_user\_id, user\_inst\_id, substr(firstname,1,4) first4, firstname, substr(middlename,1,4) middle4, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse) a,

(select uswin\_user\_id, uswin\_firstname, uswin\_lastname, lower(substr(uswin\_firstname,1,4)) as firstname4, lower(uswin\_lastname) as lastname

from xng\_reports.ldap\_users) b

where a.first4 = b.firstname4

and (a.lastname = b.lastname

or a.middlename = b.lastname)

union

--match lastname/first 4 of middle name

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 15 as match\_type

from (select xng\_user\_id, user\_inst\_id, substr(firstname,1,4) first4, firstname, substr(middlename,1,4) middle4, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse) a,

(select uswin\_user\_id, uswin\_firstname, uswin\_lastname, lower(substr(uswin\_firstname,1,4)) as firstname4, lower(uswin\_lastname) as lastname

from xng\_reports.ldap\_users) b

where a.middle4 = b.firstname4

and (a.lastname = b.lastname

or a.middlename = b.lastname)

union

--match lastname/first 3 of first name

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 16 as match\_type

from (select xng\_user\_id, user\_inst\_id, substr(firstname,1,3) first3, firstname, substr(middlename,1,3) middle3, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse) a,

(select uswin\_user\_id, uswin\_firstname, uswin\_lastname, lower(substr(uswin\_firstname,1,3)) as firstname3, lower(uswin\_lastname) as lastname

from xng\_reports.ldap\_users) b

where a.first3 = b.firstname3

and (a.lastname = b.lastname

or a.middlename = b.lastname)

union

--match lastname/first 3 of middle name

select b.uswin\_user\_id, xng\_user\_id, user\_inst\_id, 17 as match\_type

from (select xng\_user\_id, user\_inst\_id, substr(firstname,1,3) first3, firstname, substr(middlename,1,3) middle3, middlename, lastname, firstlen, middlelen, lastlen

from xng\_reports.user\_inst\_name\_parse) a,

(select uswin\_user\_id, uswin\_firstname, uswin\_lastname, lower(substr(uswin\_firstname,1,3)) as firstname3, lower(uswin\_lastname) as lastname

from xng\_reports.ldap\_users) b

where a.middle3 = b.firstname3

and (a.lastname = b.lastname

or a.middlename = b.lastname)

) matches

where alluser.user\_inst\_id = matches.user\_inst\_id (+)));

commit work;

--these were not picked up in matching and so were done manually

OPEN man\_cursor;

LOOP

FETCH man\_cursor

INTO man\_row;

EXIT WHEN man\_cursor%NOTFOUND;

update xng\_reports.ldap\_xng\_match\_wrk

set uswin\_user\_id = man\_row.uswin\_user\_id,

match\_type = 50

where full\_name = man\_row.xng\_fullname

and uswin\_user\_id = 'nfd';

commit;

update xng\_reports.ldap\_xng\_match\_wrk

set modify\_xng = 'Y'

where full\_name = man\_row.xng\_fullname

and modify\_xng = 'N'

and uswin\_user\_id <> user\_name;

commit;

END LOOP;

CLOSE man\_cursor;

COMMIT WORK;

XNG\_REPORTS.BTP\_XNG\_AUDIT\_TRUNCATES('ldap\_xng\_mismatch');

v\_xng\_id\_hld := '0';

OPEN detail\_cursor;

LOOP

FETCH detail\_cursor

INTO detail\_row;

EXIT WHEN detail\_cursor%NOTFOUND;

if v\_xng\_id\_hld <> detail\_row.user\_inst\_id then

insert into xng\_reports.ldap\_xng\_mismatch (uswin\_user\_id, user\_inst\_id, user\_name, full\_name, department,

telephone, email\_addr, comments, flags, modify\_xng, upd\_done)

values(detail\_row.uswin\_user\_id, detail\_row.user\_inst\_id, detail\_row.user\_name, detail\_row.full\_name,

detail\_row.department, detail\_row.telephone, detail\_row.email\_addr, detail\_row.comments,

detail\_row.flags, detail\_row.modify\_xng,'N');

v\_xng\_id\_hld := detail\_row.user\_inst\_id;

commit work;

end if;

END LOOP;

CLOSE detail\_cursor;

COMMIT WORK;

--found bad match for this user

update xng\_reports.ldap\_xng\_mismatch

set uswin\_user\_id = 'harrsh7',

modify\_xng = 'Y'

where user\_name = 'harrish';

commit work;

--this is an old dup account that is disabled so changing so real one can set reset correctly

update xng\_reports.ldap\_xng\_mismatch

set uswin\_user\_id = 'jordajon0',

modify\_xng = 'Y'

where user\_name = 'jordajon';

commit work;

--one xng user has the others uswin user id so setting to not auto update - will do upd separate

update xng\_reports.ldap\_xng\_mismatch

set modify\_xng = 'N'

where user\_name = 'davisgle';

commit work;

update xng\_reports.ldap\_xng\_mismatch

set uswin\_user\_id = 'ggdavis',

modify\_xng = 'N'

where user\_name = 'gdavis';

commit work;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END find\_uswin\_ids;

PROCEDURE update\_xng\_tables

IS

v\_uswin\_id xng\_reports.ldap\_users.uswin\_user\_id%type;

v\_xng\_id xperadmin.user\_inst.user\_name%type;

v\_user\_inst\_id xperadmin.user\_inst.user\_inst\_id%type;

v\_dupck xperadmin.user\_inst.user\_inst\_id%type;

v\_timeck char(5);

v\_cntck char(5);

v\_sdate date;

v\_process\_name xng\_reports.all\_processes.process\_name%type;

cursor upd\_cursor is

select uswin\_user\_id, user\_name, user\_inst\_id, dupck

from (select a.uswin\_user\_id, a.user\_name, a.user\_inst\_id, rownum as cntck,

nvl(b.user\_inst\_id, 0) dupck --when xng user\_name changing to uswin\_user\_id that is already in GI use existing user\_inst\_id

from xng\_reports.ldap\_xng\_mismatch a, xperadmin.user\_inst b

where a.uswin\_user\_id = b.user\_name (+)

and a.modify\_xng = 'Y'

and a.upd\_done = 'N'

--and a.user\_name = 'modugsa1'

)

order by user\_name;

BEGIN

v\_sdate := sysdate;

v\_uswin\_id := '0';

v\_xng\_id := '0';

v\_user\_inst\_id := 0;

v\_dupck := 0;

v\_timeck := 'go';

v\_process\_name := 'LDAP\_USER\_CLEANUP';

xng\_reports.WATCHDOG.updateprocessstart(v\_process\_name);

--comment this out when doing second run so you do not lose dups from first run

--XNG\_REPORTS.BTP\_XNG\_AUDIT\_TRUNCATES('ldap\_xng\_mismatch\_dups');

OPEN upd\_cursor;

LOOP

FETCH upd\_cursor

INTO v\_uswin\_id, v\_xng\_id, v\_user\_inst\_id, v\_dupck;

EXIT WHEN upd\_cursor%NOTFOUND;

update vzwnet.wo\_task\_inst

set originator = v\_uswin\_id

where originator = v\_xng\_id;

commit work;

update vzwnet.work\_order\_inst

set originator = v\_uswin\_id

where originator = v\_xng\_id;

commit work;

update vzwnet.inst\_audit

set chg\_by = v\_uswin\_id

where chg\_by = v\_xng\_id;

commit work;

if v\_dupck = 0 then

update xperadmin.user\_inst

set user\_name = v\_uswin\_id

where user\_inst\_id = v\_user\_inst\_id;

commit work;

update ldap\_xng\_mismatch

set upd\_done = 'Y'

where user\_inst\_id = v\_user\_inst\_id;

commit work;

else

insert into xng\_reports.ldap\_xng\_mismatch\_dups

(select uswin\_user\_id, user\_inst\_id, user\_name, full\_name, department,

telephone, email\_addr, comments, flags, modify\_xng

from xng\_reports.ldap\_xng\_mismatch

where user\_inst\_id = v\_user\_inst\_id);

end if;

select case when sysdate < to\_date('101920120630','mmddyyyyhhmi') then 'go' else 'stop' end

into v\_timeck

from dual;

if v\_timeck = 'stop' then

exit;

end if;

END LOOP;

CLOSE upd\_cursor;

COMMIT WORK;

--comment this out for the first run since they should be run when everything is finished

--this user has someone elses uswin id so doing him first to avoid dup error

if v\_timeck = 'go' then

update vzwnet.wo\_task\_inst

set originator = 'ggdavis'

where originator = 'gdavis';

commit work;

update vzwnet.work\_order\_inst

set originator = 'ggdavis'

where originator = 'gdavis';

commit work;

update vzwnet.inst\_audit

set chg\_by = 'ggdavis'

where chg\_by = 'gdavis';

commit work;

update xperadmin.user\_inst

set user\_name = 'ggdavis'

where user\_inst\_id = 24573;

commit work;

update ldap\_xng\_mismatch

set upd\_done = 'Y',

modify\_xng = 'Y'

where user\_inst\_id = 24573;

commit work;

--can update him now

update vzwnet.wo\_task\_inst

set originator = 'gdavis'

where originator = 'davisgle';

commit work;

update vzwnet.work\_order\_inst

set originator = 'gdavis'

where originator = 'davisgle';

commit work;

update vzwnet.inst\_audit

set chg\_by = 'gdavis'

where chg\_by = 'davisgle';

commit work;

update xperadmin.user\_inst

set user\_name = 'gdavis'

where user\_inst\_id = 6356;

commit work;

update ldap\_xng\_mismatch

set upd\_done = 'Y',

modify\_xng = 'Y'

where user\_inst\_id = 6356;

commit work;

end if;

xng\_reports.WATCHDOG.updateprocessend(v\_process\_name,'STATUS\_SUCCESS','Y');

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END update\_xng\_tables;

END LDAP\_XNG\_USER\_FIX;

/

--------------------------------------------------------

-- DDL for Package Body LEASED\_SEG\_DF

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."LEASED\_SEG\_DF"

IS

--

-- To modify this template, edit file PKGBODY.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package body

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter procedure, function bodies as shown below

PROCEDURE load\_leased\_seg\_df\_details

IS

sql\_stmt varchar2(32000);

BEGIN

EXECUTE IMMEDIATE 'truncate table leased\_seg\_df\_ndf\_path\_udas\_wk reuse storage';

-- get capacity reporting udas for non\_dark\_fiber segments

sql\_stmt:= ' insert into leased\_seg\_df\_ndf\_path\_udas\_wk columns(NDF\_PATH\_ID,TRANSPORT\_ACTION\_UDA,COMP\_DATE\_UDA,ACTION\_JUST\_UDA,IS\_EXEMPT)

with

all\_df\_segments as (

select distinct CI.CIRC\_INST\_ID df\_seg\_id, CI.CIRC\_HUM\_ID df\_seg\_name, CI.BANDWIDTH df\_seg\_bw,CI.STATUS df\_seg\_status,

CI.Z\_SITE\_ID df\_site\_id, SI.SITE\_HUM\_ID df\_site\_name,

case when ci.CIRC\_PATH\_INST\_ID is null then CI.NEXT\_PATH\_INST\_ID

else ci.CIRC\_PATH\_INST\_ID

end df\_path\_id

from

vzwnet.site\_inst si,

VZWNET.CIRC\_INST ci

where ( ci.type=''DARK FIBER'' or ci.bandwidth=''DARK FIBER'')

and CI.STATUS not in (''Decommissioned'',''Cancelled'')

and SI.SITE\_INST\_ID = CI.Z\_SITE\_ID

and si.num <> ''MTSO''

)

,all\_ndf\_segments as (

select distinct CI.CIRC\_INST\_ID NDF\_SEG\_ID, CI.CIRC\_HUM\_ID ndf\_seg\_name, CI.BANDWIDTH ndf\_seg\_bw,CI.STATUS ndf\_seg\_status,ci.type ndf\_seg\_type,

ci.Z\_SITE\_ID ndf\_site\_id,

case when ci.CIRC\_PATH\_INST\_ID is null then CI.NEXT\_PATH\_INST\_ID

else ci.CIRC\_PATH\_INST\_ID

end ndf\_path\_id, CPI.circ\_Path\_hum\_id ndf\_path\_name, CPI.BANDWIDTH ndf\_path\_bw,cpi.status ndf\_path\_status,

df.\*

from

vzwnet.site\_inst si,all\_df\_segments df,

VZWNET.CIRC\_INST ci

left outer join vzwnet.circ\_path\_inst cpi on (ci.circ\_path\_inst\_id=CPI.CIRC\_PATH\_INST\_ID or ci.next\_path\_inst\_id=cpi.circ\_path\_inst\_id)

where ( CI.Z\_SITE\_ID = df.df\_site\_id )--or ci.A\_site\_id = df.df\_site\_id)

and ( ci.type<>''DARK FIBER'' and ci.bandwidth<>''DARK FIBER'')

and CI.STATUS not in (''Decommissioned'',''Cancelled'')

and SI.SITE\_INST\_ID = CI.Z\_SITE\_ID

)

,TRANSPORT\_ACTION\_UDA AS

( SELECT PAS1.CIRC\_PATH\_INST\_ID, PAS1.ATTR\_VALUE transport\_action\_uda

FROM vzwnet.val\_attr\_name van ,

vzwnet.circ\_path\_ATTR\_SETTINGS pas1

where van.attr\_name = ''Transport Action''

and van.group\_name = ''Capacity Reporting''

and PAS1.VAL\_ATTR\_INST\_ID = VAN.VAL\_ATTR\_INST\_ID

),COMP\_DATE\_UDA AS

( SELECT CIRC\_PATH\_INST\_ID, --comp\_date\_uda

case

when length(comp\_date\_uda) = 9 then to\_date(comp\_date\_uda,''dd-mon-yy'')

when length(comp\_date\_uda) = 11 then to\_date(comp\_date\_uda,''dd-mon-yyyy'')

else null

end comp\_date\_uda

from

(

SELECT PAS2.CIRC\_PATH\_INST\_ID, substr(PAS2.ATTR\_VALUE,1,11) as comp\_date\_uda

FROM vzwnet.val\_attr\_name van ,

vzwnet.circ\_path\_ATTR\_SETTINGS pas2

where van.attr\_name = ''Estimated Completion Date''

and van.group\_name = ''Capacity Reporting''

and PAS2.VAL\_ATTR\_INST\_ID = VAN.VAL\_ATTR\_INST\_ID

)

),ACTION\_JUSTIFICATION\_UDA AS

( SELECT PAS3.CIRC\_PATH\_INST\_ID, PAS3.ATTR\_VALUE action\_just\_uda

FROM vzwnet.val\_attr\_name van ,

vzwnet.circ\_path\_ATTR\_SETTINGS pas3

where van.attr\_name = ''Action Justification''

and van.group\_name = ''Capacity Reporting''

and PAS3.VAL\_ATTR\_INST\_ID = VAN.VAL\_ATTR\_INST\_ID

),

all\_udas as

(select distinct ap.ndf\_path\_id, tau.transport\_action\_uda, cdu.comp\_date\_uda, aju.action\_just\_uda

from all\_ndf\_segments ap

left outer join TRANSPORT\_ACTION\_UDA tau on tau.circ\_path\_inst\_id = ap.ndf\_path\_id

left outer join COMP\_DATE\_UDA cdu on cdu.circ\_path\_inst\_id = ap.ndf\_path\_id

left outer join ACTION\_JUSTIFICATION\_UDA aju on aju.circ\_path\_inst\_id = ap.ndf\_path\_id

where transport\_action\_uda is not null

)select ndf\_path\_id, transport\_action\_uda, comp\_date\_uda, action\_just\_uda

, case when (transport\_action\_uda = ''To Be Disconnected'' and comp\_date\_uda - sysdate < -1) then ''N''

when (transport\_action\_uda = ''Keep'' and action\_just\_uda is null) then ''N''

else ''Y''

end is\_exempt

from all\_udas

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXECUTE IMMEDIATE 'truncate table leased\_seg\_df\_details\_wk reuse storage';

sql\_stmt:= 'insert into leased\_seg\_df\_details\_wk columns (AREA,REGION,DOMAIN\_NAME,DF\_SITE\_ID,

DF\_SITE\_NAME,NDF\_SEG\_ID,NDF\_SEG\_NAME,NDF\_SEG\_STATUS,NDF\_SEG\_TYPE,NDF\_SEG\_BW,IN\_SERVICE\_DATE,

BTP\_EXTRACT\_DATE,UDA\_DISCONNECT\_DATE,NDF\_PATH\_ID,NDF\_PATH\_NAME,NDF\_PATH\_STATUS,NDF\_PATH\_BW,

TRANSPORT\_ACTION\_UDA,COMP\_DATE\_UDA,ACTION\_JUST\_UDA,IS\_EXEMPT,DF\_SEG\_ID,DF\_SEG\_NAME,

DF\_SEG\_STATUS,DF\_SEG\_TYPE,DF\_SEG\_BW, DF\_SEG\_VENDOR, NDF\_SEG\_VENDOR, NDF\_PATH\_TYPE,

DF\_RECUR\_COST, NDF\_RECUR\_COST, DF\_NON\_RECUR\_COST, NDF\_NON\_RECUR\_COST )

with

LEAF\_DOMAINS AS (

SELECT domain\_inst\_id

FROM vzwnet.domain\_inst di

where domain\_inst\_id <> 1012 --exclude NNO domain

MINUS -- minus the parents

SELECT parent\_domain\_inst\_id

FROM vzwnet.domain\_inst

)

,all\_df\_segments as (

select distinct CI.CIRC\_INST\_ID df\_seg\_id, CI.CIRC\_HUM\_ID df\_seg\_name,ci.type df\_seg\_type, CI.BANDWIDTH df\_seg\_bw,CI.STATUS df\_seg\_status,ci.vendor df\_seg\_vendor,

CI.Z\_SITE\_ID df\_site\_id, SI.SITE\_HUM\_ID df\_site\_name,ci.recur\_costs df\_recur\_costs, ci.non\_recur\_costs df\_nonrecur\_costs,

case when ci.CIRC\_PATH\_INST\_ID is null then CI.NEXT\_PATH\_INST\_ID

else ci.CIRC\_PATH\_INST\_ID

end df\_path\_id, CPI.circ\_Path\_hum\_id df\_path\_name, CPI.BANDWIDTH df\_path\_bw,cpi.status df\_path\_status

from

vzwnet.site\_inst si,

VZWNET.CIRC\_INST ci

left outer join vzwnet.circ\_path\_inst cpi on (ci.circ\_path\_inst\_id=CPI.CIRC\_PATH\_INST\_ID or ci.next\_path\_inst\_id=cpi.circ\_path\_inst\_id)

where ( ci.type=''DARK FIBER'' or ci.bandwidth=''DARK FIBER'')

and CI.STATUS not in (''Decommissioned'',''Cancelled'')

and SI.SITE\_INST\_ID = CI.Z\_SITE\_ID

and si.num <> ''MTSO''

)

,all\_ndf\_segments as (

select distinct CI.CIRC\_INST\_ID NDF\_SEG\_ID, CI.CIRC\_HUM\_ID ndf\_seg\_name, CI.BANDWIDTH ndf\_seg\_bw,CI.STATUS ndf\_seg\_status,ci.type ndf\_seg\_type,ci.vendor ndf\_seg\_vendor,

ci.Z\_SITE\_ID ndf\_site\_id,ci.recur\_costs ndf\_recur\_costs, ci.non\_recur\_costs ndf\_nonrecur\_costs,

case when ci.CIRC\_PATH\_INST\_ID is null then CI.NEXT\_PATH\_INST\_ID

else ci.CIRC\_PATH\_INST\_ID

end ndf\_path\_id, CPI.circ\_Path\_hum\_id ndf\_path\_name, CPI.BANDWIDTH ndf\_path\_bw,cpi.status ndf\_path\_status, cpi.type ndf\_path\_type ,

df.\*

from

vzwnet.site\_inst si,all\_df\_segments df,

VZWNET.CIRC\_INST ci

left outer join vzwnet.circ\_path\_inst cpi on (ci.circ\_path\_inst\_id=CPI.CIRC\_PATH\_INST\_ID or ci.next\_path\_inst\_id=cpi.circ\_path\_inst\_id)

where ( CI.Z\_SITE\_ID = df.df\_site\_id )

and ( ci.type <>''DARK FIBER'' and ci.bandwidth<>''DARK FIBER'')

and CI.STATUS not in (''Decommissioned'',''Cancelled'')

and SI.SITE\_INST\_ID = CI.Z\_SITE\_ID

)

,btp\_ndf\_data as (

select ndf.\*, BXA.IN\_SERVICE\_DATE, BXA.BTP\_EXTRACT\_DATE

from all\_ndf\_segments ndf

left outer join xng\_reports.btp\_xng\_audit bxa

on ndf.NDF\_SEG\_ID = BXA.XNG\_CIRC\_INST\_ID

)

,DISCONN\_DATE\_UDA AS

(SELECT van.val\_attr\_inst\_id

FROM vzwnet.val\_attr\_name van

where van.attr\_name = ''Disconnect Date''

and van.group\_name = ''Disconnect Information''

),

SEG\_DISCON\_DATES as

(select circ\_inst\_id,

case

when length(uda\_disconnect\_date) = 9 then to\_date(uda\_disconnect\_date,''dd-mon-yy'')

when length(uda\_disconnect\_date) = 11 then to\_date(uda\_disconnect\_date,''dd-mon-yyyy'')

else null

end uda\_disconnect\_date

from (select cas.circ\_inst\_id, substr(cas.attr\_value,1,11) as uda\_disconnect\_date

from vzwnet.circ\_attr\_settings cas

join btp\_ndf\_data btp on cas.circ\_inst\_id =btp.NDF\_SEG\_ID

join DISCONN\_DATE\_UDA van on van.val\_attr\_inst\_id = cas.VAL\_ATTR\_INST\_ID)

)

select distinct dlr.area, dlr.region, dlr.domain\_name,

df\_site\_id,df\_site\_name,

ndf\_seg\_id,ndf\_seg\_name,ndf\_seg\_status,ndf\_seg\_type,ndf\_seg\_bw,

in\_service\_date, btp\_extract\_date,dd.uda\_disconnect\_date,

bndf.ndf\_path\_id,ndf\_path\_name,ndf\_path\_status,ndf\_path\_bw, PU.TRANSPORT\_ACTION\_UDA, PU.COMP\_DATE\_UDA, PU.ACTION\_JUST\_UDA, PU.IS\_EXEMPT,

df\_seg\_id,df\_seg\_name,df\_seg\_status,df\_seg\_type,df\_seg\_bw,df\_seg\_vendor, ndf\_seg\_vendor, ndf\_path\_type, df\_recur\_costs, ndf\_recur\_costs, df\_nonrecur\_costs, ndf\_nonrecur\_costs

from vzwnet.circ\_domain\_map cdm

join btp\_ndf\_data bndf on bndf.NDF\_SEG\_ID = cdm.circ\_inst\_id

left outer join SEG\_DISCON\_DATES dd on bndf.NDF\_SEG\_ID = dd.circ\_inst\_id

left outer join xng\_reports.leased\_seg\_df\_ndf\_path\_udas\_wk pu on PU.NDF\_PATH\_ID=bndf.ndf\_path\_id

join LEAF\_DOMAINS li on cdm.domain\_inst\_id = li.domain\_inst\_id

join vzwnet.domain\_inst di on li.domain\_inst\_id = di.domain\_inst\_id

join xng\_reports.domains\_leaf\_reporting dlr on DLR.DOMAIN\_INST\_ID = li.domain\_inst\_id

where (PU.IS\_EXEMPT is null or pu.is\_exempt = ''N'')

order by area, region, domain\_name

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in load\_leased\_seg\_df\_details(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend (PROCESS\_NAME,'STATUS\_FAILURE', 'N');

RAISE;

END;

PROCEDURE load\_leased\_seg\_df\_summary

IS

sql\_stmt varchar2(32000);

BEGIN

EXECUTE IMMEDIATE 'truncate table leased\_seg\_df\_summary\_wk reuse storage';

sql\_stmt := 'insert into leased\_seg\_df\_summary\_wk

with all\_ndf\_seg as (

select distinct area, region, edd.NDF\_SEG\_ID

from leased\_seg\_df\_details\_wk edd

)

select area, region, count(NDF\_SEG\_ID)

from all\_ndf\_seg group by rollup (area, region)

order by area, region

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in load\_leased\_seg\_df\_summary(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend (PROCESS\_NAME,'STATUS\_FAILURE', 'N');

RAISE;

END;

PROCEDURE do\_all

IS

sql\_stmt varchar2(32000);

BEGIN

watchdog.updateProcessStart(PROCESS\_NAME);

load\_leased\_seg\_df\_details();

load\_leased\_seg\_df\_summary();

watchdog.updateProcessEnd(PROCESS\_NAME,'STATUS\_SUCCESS','Y');

EXCEPTION

WHEN OTHERS THEN

watchdog.logerror (PROCESS\_NAME, 4000,

SUBSTR ( 'Error in do\_all(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend (PROCESS\_NAME,'STATUS\_FAILURE', 'N');

END;

-- Enter further code below as specified in the Package spec.

END;

/

--------------------------------------------------------

-- DDL for Package Body LIVE\_OBJECTS\_NONLIVE\_PATHS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."LIVE\_OBJECTS\_NONLIVE\_PATHS" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: LIVE\_OBJECTS\_NONLIVE\_PATHS

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 6/03/2013 Jaya Chilakamarri 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE insert\_live\_object\_data

IS

sqlStmt VARCHAR2(32267);

table\_name VARCHAR2 (30) := 'live\_object\_nl\_paths\_wk';

BEGIN

EXECUTE IMMEDIATE 'truncate table ' || table\_name ||' reuse storage';

sqlStmt:= '

insert into ' || table\_name ||

' WITH all\_paths

AS (SELECT cpi.circ\_path\_inst\_id,CPI.CIRC\_PATH\_HUM\_ID||''(Rev ''|| (CPI.CIRC\_PATH\_REV\_NBR) || '')'' path\_hum\_id,CPI.STATUS path\_status,cpi.type path\_type,DRR.AREA,DRR.REGION,DRR.DOMAIN\_INST\_ID

FROM vzwnet.circ\_path\_inst cpi,

vzwnet.site\_inst si,

VZWNET.SITE\_DOMAIN\_MAP sdm,

XNG\_REPORTS.DOMAINS\_REGIONAL\_REPORTING drr

WHERE CPI.STATUS <> ''Live''

AND CPI.Z\_SIDE\_SITE\_ID = SI.SITE\_INST\_ID

AND SI.SITE\_INST\_ID = SDM.SITE\_INST\_ID

AND SDM.DOMAIN\_INST\_ID = DRR.DOMAIN\_INST\_ID),

all\_live\_segments\_nl\_paths

AS (SELECT cpe.circ\_path\_inst\_id,all\_paths.path\_hum\_id,cpe.sequence,

cpe.element\_type,ci.status segment\_status,ci.circ\_inst\_id elem\_inst\_id,ci.TYPE,ci.circ\_hum\_id hum\_id, ci.RECUR\_COSTS MRC,ci.vendor,path\_status,path\_type,area,region

FROM vzwnet.circ\_path\_element cpe, all\_paths, vzwnet.circ\_inst ci

WHERE cpe.circ\_path\_inst\_id = all\_paths.circ\_path\_inst\_id

AND ci.circ\_inst\_id = cpe.segment\_inst\_id

AND cpe.element\_type = ''S''

AND ci.status = ''Live''),

seg\_on\_next\_paths

AS (SELECT lo.\* -- , CPI.PREV\_PATH\_INST\_ID, CI.NEXT\_PATH\_INST\_ID

FROM all\_live\_segments\_nl\_paths lo

JOIN

vzwnet.circ\_inst ci

ON LO.elem\_inst\_id = CI.CIRC\_INST\_ID

AND LO.circ\_path\_inst\_id = CI.NEXT\_PATH\_INST\_ID

JOIN

vzwnet.circ\_path\_inst cpi

ON LO.circ\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

AND CI.CIRC\_PATH\_INST\_ID = CPI.PREV\_PATH\_INST\_ID),

valid\_segments

AS (SELECT \* FROM all\_live\_segments\_nl\_paths

MINUS

SELECT \* FROM seg\_on\_next\_paths),

all\_live\_cables\_nl\_paths

AS (SELECT cpe.circ\_path\_inst\_id,all\_paths.path\_hum\_id,cpe.sequence,

cpe.element\_type,ci.status cable\_status, ci.cable\_inst\_id elem\_inst\_id,ci.TYPE,ci.cable\_name hum\_id,0 MRC,ci.vendor,path\_status,path\_type,area,region

FROM vzwnet.circ\_path\_element cpe, all\_paths, vzwnet.cable\_inst ci

WHERE cpe.circ\_path\_inst\_id = all\_paths.circ\_path\_inst\_id

AND ci.cable\_inst\_id = cpe.cable\_inst\_id --cpe.segment\_inst\_id

AND cpe.element\_type = ''B''

AND ci.status = ''Live''),

cable\_on\_next\_paths

AS (SELECT lo.\*

FROM all\_live\_cables\_nl\_paths lo

JOIN

vzwnet.cable\_inst ci

ON LO.elem\_inst\_id = CI.CABLE\_INST\_ID

AND LO.circ\_path\_inst\_id = CI.NEXT\_PATH\_INST\_ID

JOIN

vzwnet.circ\_path\_inst cpi

ON LO.circ\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

AND CI.CIRC\_PATH\_INST\_ID = CPI.PREV\_PATH\_INST\_ID),

valid\_cables

AS (SELECT \* FROM all\_live\_cables\_nl\_paths

MINUS

SELECT \* FROM cable\_on\_next\_paths ),

all\_live\_equip\_nl\_paths as(

select cpe.circ\_path\_inst\_id, all\_paths.path\_hum\_id, cpe.sequence, epa.port\_inst\_id

,cpe.element\_type, ei.status equip\_status, ei.type, ei.equip\_inst\_id obj\_inst\_id, ei.descr obj\_hum\_id, 0 MRC, ei.vendor, path\_status,path\_type, area, region

from vzwnet.circ\_path\_element cpe, all\_paths, vzwnet.epa epa, vzwnet.equip\_inst ei

where cpe.circ\_path\_inst\_id=all\_paths.circ\_path\_inst\_id

and epa.port\_inst\_id=cpe.port\_inst\_id

and ei.equip\_inst\_id=epa.equip\_inst\_id

and cpe.element\_type=''E''

and ei.status = ''Live''

and EI.TYPE not in (''AC DIST PANEL'',''DSX-0'', ''DSX-1'', ''DSX-3'', ''FIBER PANEL'', ''PATCH PANEL'',''LGX'')

) ,

eq\_on\_next\_paths

AS (SELECT lo.\* , CPI.PREV\_PATH\_INST\_ID, CPI.NEXT\_PATH\_INST\_ID

FROM all\_live\_equip\_nl\_paths lo

JOIN

vzwnet.epa epa

ON LO.port\_inst\_id = epa.port\_INST\_ID

AND LO.circ\_path\_inst\_id = epa.NEXT\_PATH\_INST\_ID

JOIN

vzwnet.circ\_path\_inst cpi

ON LO.circ\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

AND epa.CIRC\_PATH\_INST\_ID = CPI.PREV\_PATH\_INST\_ID),

valid\_equip AS

(

SELECT circ\_path\_inst\_id, path\_hum\_id, sequence,element\_type, equip\_status,type, obj\_inst\_id, obj\_hum\_id, MRC,vendor,path\_status,path\_type, area, region

FROM all\_live\_equip\_nl\_paths

MINUS

SELECT circ\_path\_inst\_id, path\_hum\_id, sequence,element\_type, equip\_status,type, obj\_inst\_id, obj\_hum\_id, MRC,vendor,path\_status,path\_type, area, region

FROM eq\_on\_next\_paths

)

select seg.area, seg.region, seg.circ\_path\_inst\_id, seg.path\_hum\_id, seg.path\_status, seg.element\_type category, seg.type object\_type, seg.sequence,

seg.elem\_inst\_id obj\_inst\_id, seg.hum\_id obj\_hum\_id, seg.segment\_status obj\_status, vendor, seg.mrc, sysdate ,seg.path\_type

from valid\_segments seg

union

select cable.area, cable.region, cable.circ\_path\_inst\_id, cable.path\_hum\_id, cable.path\_status, cable.element\_type category, cable.type object\_type, cable.sequence,

cable.elem\_inst\_id obj\_inst\_id, cable.hum\_id obj\_hum\_id, cable.cable\_status obj\_status, vendor, cable.mrc, sysdate ,cable.path\_type

from valid\_cables cable

union

select eq.area, eq.region, eq.circ\_path\_inst\_id, eq.path\_hum\_id, eq.path\_status, eq.element\_type category, eq.type object\_type, eq.sequence,

eq.obj\_inst\_id, eq.obj\_hum\_id, eq.equip\_status obj\_status, vendor, eq.mrc, sysdate, eq.path\_type

from valid\_equip eq

'

;

EXECUTE IMMEDIATE sqlStmt;

commit;

sqlStmt:= '

insert into ' || table\_name ||

' with all\_paths as (

select cpi.circ\_path\_inst\_id, CPI.PREV\_PATH\_INST\_ID, CPI.CIRC\_PATH\_HUM\_ID||''(Rev ''|| (CPI.CIRC\_PATH\_REV\_NBR) || '')'' path\_hum\_id, CPI.STATUS path\_status,CPI.type path\_type, DRR.AREA, DRR.REGION, DRR.DOMAIN\_INST\_ID

from vzwnet.circ\_path\_inst cpi, vzwnet.site\_inst si, VZWNET.SITE\_DOMAIN\_MAP sdm, XNG\_REPORTS.DOMAINS\_REGIONAL\_REPORTING drr

where CPI.STATUS <> ''Live'' and CPI.Z\_SIDE\_SITE\_ID = SI.SITE\_INST\_ID and SI.SITE\_INST\_ID = SDM.SITE\_INST\_ID and SDM.DOMAIN\_INST\_ID = DRR.DOMAIN\_INST\_ID --and CPI.CIRC\_PATH\_INST\_ID = 2933693

)

,all\_live\_pathelem\_nl\_paths as(

select cpe.circ\_path\_inst\_id, all\_paths.PREV\_PATH\_INST\_ID, LI.LEG\_INST\_ID, LI.LEG\_NAME, all\_paths.path\_hum\_id, cpe.sequence, cpe.element\_type, ci.status path\_element\_status, ci.circ\_path\_inst\_id obj\_inst\_id, ci.type, ci.circ\_path\_hum\_id obj\_hum\_id, 0 MRC, ''N/A'' vendor, path\_status,path\_type,area, region

from vzwnet.circ\_path\_element cpe, all\_paths, vzwnet.circ\_path\_inst ci, VZWNET.PATH\_LEG\_INST li

where cpe.circ\_path\_inst\_id=all\_paths.circ\_path\_inst\_id

and CPE.CIRC\_PATH\_INST\_ID = LI.CIRC\_PATH\_INST\_ID

and ci.circ\_path\_inst\_id=cpe.path\_inst\_id

and cpe.element\_type=''P''

and ci.status = ''Live''

)

,pathelem\_on\_next\_paths AS (

SELECT lo.\*

FROM all\_live\_pathelem\_nl\_paths lo

JOIN

vzwnet.circ\_path\_element cpe2

ON LO.prev\_path\_inst\_id = CPE2.CIRC\_PATH\_INST\_ID and lo.obj\_inst\_id = CPE2.PATH\_INST\_ID

join VZWNET.PATH\_LEG\_INST li2 on lo.LEG\_NAME = li2.LEG\_NAME and li2.circ\_path\_inst\_id = CPE2.CIRC\_PATH\_INST\_ID

union

SELECT lo.\*

FROM all\_live\_pathelem\_nl\_paths lo

JOIN vzwnet.circ\_path\_element cpe3 ON LO.prev\_path\_inst\_id = CPE3.CIRC\_PATH\_INST\_ID

join VZWNET.PATH\_LEG\_INST li3 on lo.LEG\_NAME = li3.LEG\_NAME and li3.circ\_path\_inst\_id = CPE3.CIRC\_PATH\_INST\_ID

join VZWNET.CIRC\_PATH\_INST cpi3 on lo.obj\_inst\_id = CPI3.NEXT\_PATH\_INST\_ID and CPE3.PATH\_INST\_ID = CPI3.CIRC\_PATH\_INST\_ID

),

valid\_paths

AS (

SELECT \* FROM all\_live\_pathelem\_nl\_paths

MINUS

SELECT \* FROM pathelem\_on\_next\_paths

) select pth.area, pth.region, pth.circ\_path\_inst\_id, pth.path\_hum\_id, pth.path\_status, pth.element\_type category, pth.type object\_type, pth.sequence,

pth.obj\_inst\_id obj\_inst\_id, pth.obj\_hum\_id obj\_hum\_id, pth.path\_element\_status obj\_status, pth.vendor, pth.mrc, sysdate, pth.path\_type

from valid\_paths pth'

;

-- dbms\_output.put\_line(sqlStmt);

EXECUTE IMMEDIATE sqlStmt;

commit;

-- dbms\_output.put\_line('Done populating '|| table\_name);

EXECUTE IMMEDIATE 'truncate table live\_object\_nl\_paths reuse storage';

EXECUTE IMMEDIATE 'insert into live\_object\_nl\_paths select \* from live\_object\_nl\_paths\_wk';

commit;

-- dbms\_output.put\_line('Done populating live\_object\_nl\_paths ');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in insert\_live\_object\_data(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

COMMIT;

-- dbms\_output.put\_line('Done populating '|| table\_name);

END insert\_live\_object\_data;

PROCEDURE insert\_live\_object\_data\_rollup

IS

sqlStmt VARCHAR2(32267);

table\_name VARCHAR2 (30) := 'live\_object\_nl\_paths\_Summary';

BEGIN

EXECUTE IMMEDIATE 'truncate table ' || table\_name ||' reuse storage';

sqlStmt:=

'

insert into ' || table\_name ||' col(area, region, segment\_ebh\_count, cable\_ebh\_count, path\_element\_ebh\_count, equip\_ebh\_count,

segment\_nebh\_count, cable\_nebh\_count, path\_element\_nebh\_count, equip\_nebh\_count,

segment\_count, cable\_count, path\_element\_count, equip\_count, extract\_date)

with allobj\_count\_ebh as (select lnl.area, lnl.region region,

case when LNL.CATEGORY = ''S'' then 1 else 0 end as segment\_ebh\_count,

case when LNL.CATEGORY = ''B'' then 1 else 0 end as cable\_ebh\_count,

case when LNL.CATEGORY = ''P'' then 1 else 0 end as path\_ebh\_count,

case when LNL.CATEGORY = ''E'' then 1 else 0 end as equip\_ebh\_count

from xng\_reports.LIVE\_OBJECT\_NL\_PATHS lnl where LNL.PATH\_TYPE=''EBH''

and

lnl.area not in (''OSS'') ),

ebh\_rollup as (

select area, region,

sum(segment\_ebh\_count) segment\_ebh\_count, sum(cable\_ebh\_count) cable\_ebh\_count, sum(path\_ebh\_count) path\_ebh\_count, sum(equip\_ebh\_count) equip\_ebh\_count

from allobj\_count\_ebh

group by rollup (area, region)

order by area, region

),

allobj\_count\_non\_ebh as (select lnl.area, lnl.region region,

case when LNL.CATEGORY = ''S'' then 1 else 0 end as segment\_nebh\_count,

case when LNL.CATEGORY = ''B'' then 1 else 0 end as cable\_nebh\_count,

case when LNL.CATEGORY = ''P'' then 1 else 0 end as path\_nebh\_count,

case when LNL.CATEGORY = ''E'' then 1 else 0 end as equip\_nebh\_count

from xng\_reports.LIVE\_OBJECT\_NL\_PATHS lnl where LNL.PATH\_TYPE<>''EBH''

and

lnl.area not in (''OSS'')

),

non\_ebh\_rollup as (

select area, region,

sum(segment\_nebh\_count) segment\_nebh\_count, sum(cable\_nebh\_count) cable\_nebh\_count, sum(path\_nebh\_count) path\_nebh\_count, sum(equip\_nebh\_count) equip\_nebh\_count

from allobj\_count\_non\_ebh

group by rollup (area, region)

order by area, region

),

allobj\_count as (select lnl.area, lnl.region region,

case when LNL.CATEGORY = ''S'' then 1 else 0 end as segment\_count,

case when LNL.CATEGORY = ''B'' then 1 else 0 end as cable\_count,

case when LNL.CATEGORY = ''P'' then 1 else 0 end as path\_count,

case when LNL.CATEGORY = ''E'' then 1 else 0 end as equip\_count

from xng\_reports.LIVE\_OBJECT\_NL\_PATHS lnl where lnl.area not in (''OSS'')

),

allobj\_rollup as (

select area, region,

sum(segment\_count) segment\_count, sum(cable\_count) cable\_count, sum(path\_count) path\_count, sum(equip\_count) equip\_count

from allobj\_count

group by rollup (area, region)

order by area, region

)

select er.area, er.region, er.segment\_ebh\_count, er.cable\_ebh\_count, er.path\_ebh\_count,er.equip\_ebh\_count,

ner.segment\_nebh\_count, ner.cable\_nebh\_count, ner.path\_nebh\_count,ner.equip\_nebh\_count,

ar.segment\_count, ar.cable\_count,ar.path\_count,ar.equip\_count, sysdate EXTRACT\_DATE

from ebh\_rollup er, non\_ebh\_rollup ner, allobj\_rollup ar

where er.area = ner.area

and er.region=ner.region

and ar.area = er.area

and ar.region=er.region

union

select er.area, er.region, er.segment\_ebh\_count, er.cable\_ebh\_count, er.path\_ebh\_count,er.equip\_ebh\_count,

ner.segment\_nebh\_count, ner.cable\_nebh\_count, ner.path\_nebh\_count,ner.equip\_nebh\_count,

ar.segment\_count, ar.cable\_count, ar.path\_count,ar.equip\_count, sysdate EXTRACT\_DATE

from ebh\_rollup er, non\_ebh\_rollup ner, allobj\_rollup ar

where er.area = ner.area

and er.region is null

and ner.region is null

and ar.area = er.area

and ar.region is null

'

;

EXECUTE IMMEDIATE sqlStmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in insert\_live\_object\_data\_rollup(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

COMMIT;

dbms\_output.put\_line('Done populating '|| table\_name);

END insert\_live\_object\_data\_rollup;

PROCEDURE do\_all

IS

BEGIN

insert\_live\_object\_data;

insert\_live\_object\_data\_rollup;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line (SUBSTR ( 'Error in LIVE\_OBJECTS\_NONLIVE\_PATHS:do\_all(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END do\_all;

END LIVE\_OBJECTS\_NONLIVE\_PATHS;

/

--------------------------------------------------------

-- DDL for Package Body LIVE\_OBJS\_NL\_SITES

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."LIVE\_OBJS\_NL\_SITES" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: LIVE\_OBJ\_NL\_SITES

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/17/2014 Gautam 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE insert\_objs IS

sqlStmt VARCHAR2(32267);

methodName VARCHAR2(100) := 'insert\_objs';

message VARCHAR2(200);

BEGIN

xng\_reports.watchdog.updateprocessstart(processName);

BEGIN

EXECUTE IMMEDIATE 'truncate table ' || table\_name || ' reuse storage';

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t truncate table: ' || table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

xng\_reports.watchdog.updateprocessend(processName, 'STATUS\_FAILURE', 'N');

RAISE;

END;

sqlStmt := 'insert into ' || table\_name || '

WITH

LEAF\_DOMAINS\_MINUS\_NNO as

( select pdi.\* from vzwnet.domain\_inst pdi, vzwnet.domain\_inst cdi

where pdi.DOMAIN\_INST\_ID = cdi.PARENT\_DOMAIN\_INST\_ID(+)

and cdi.DOMAIN\_INST\_ID is null

and pdi.DOMAIN\_NAME not in (''NNO\_DOMAIN'', ''SLTX\_LAB\_DOMAIN'', ''ORPHAN\_DOMAIN'')

),

all\_equip\_in\_NNO\_w\_prefix as

( select distinct ci.equip\_inst\_id

from vzwnet.equip\_domain\_map pdm, vzwnet.equip\_inst ci, vzwnet.domain\_inst di

where ci.equip\_inst\_id = pdm.equip\_inst\_id

and di.DOMAIN\_INST\_ID = pdm.DOMAIN\_INST\_ID

and di.DOMAIN\_NAME = ''NNO\_DOMAIN''

and ci.descr like ''NNO %''

and ci.status =''Live''

),

all\_equip\_in\_NNO\_w\_no\_prefix as

( select distinct ci.equip\_inst\_id

from vzwnet.equip\_domain\_map pdm, vzwnet.equip\_inst ci, vzwnet.domain\_inst di

where ci.equip\_inst\_id = pdm.equip\_inst\_id

and di.DOMAIN\_INST\_ID = pdm.DOMAIN\_INST\_ID

and di.DOMAIN\_NAME = ''NNO\_DOMAIN''

and ci.descr not like ''NNO %''

and ci.status =''Live''

),

not\_exists\_in\_other\_domains as

( select distinct equip\_inst\_id

from all\_equip\_in\_NNO\_w\_no\_prefix

minus

select distinct pdm.equip\_inst\_id

from vzwnet.equip\_domain\_map pdm, LEAF\_DOMAINS\_MINUS\_NNO di

where pdm.domain\_inst\_id = di.DOMAIN\_INST\_ID

)

, all\_nno\_equip as

( select distinct equip\_inst\_id

from all\_equip\_in\_NNO\_w\_prefix

union

select distinct equip\_inst\_id

from not\_exists\_in\_other\_domains

),

OTHER\_DOMAIN\_EQUIPS as

(

select distinct c1.equip\_inst\_id

from vzwnet.equip\_inst c1

where

c1.status=''Live''

MINUS

select equip\_inst\_id

from all\_nno\_equip

)

select drr.AREA, (select dr.REGION from DOMAINS\_LEAF\_REPORTING dr where dr.MARKET = drr.MARKET ) REGION, drr.MARKET, count(ei.equip\_inst\_id), 0, 0

from OTHER\_DOMAIN\_EQUIPS odm, vzwnet.equip\_inst ei, vzwnet.equip\_domain\_map d, xng\_reports.domains\_leaf\_reporting drr, vzwnet.site\_inst si,xng\_apps.nonlive\_site\_status\_chg\_date nss

where

ei.site\_inst\_id=si.site\_inst\_id

and odm.equip\_inst\_id = ei.equip\_inst\_id

and ei.equip\_inst\_id=d.equip\_inst\_id

and nss.site\_inst\_id=si.site\_inst\_id

and ei.site\_inst\_id=nss.site\_inst\_id

and d.DOMAIN\_INST\_ID = drr.DOMAIN\_INST\_ID

and ei.status=''Live''

and si.status <> ''Live''

and drr.AREA <> ''NNO''

group by (drr.AREA, drr.MARKET)

order by drr.AREA, drr.MARKET';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into table: ' || table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

xng\_reports.watchdog.updateprocessend(processName, 'STATUS\_FAILURE', 'N');

RAISE;

END;

EXECUTE IMMEDIATE 'truncate table ' || table\_name || '\_SEG reuse storage';

commit;

sqlStmt := 'INSERT INTO ' || table\_name || '\_SEG

WITH

LEAF\_DOMAINS\_MINUS\_NNO as

( select pdi.\* from vzwnet.domain\_inst pdi, vzwnet.domain\_inst cdi

where pdi.DOMAIN\_INST\_ID = cdi.PARENT\_DOMAIN\_INST\_ID(+)

and cdi.DOMAIN\_INST\_ID is null

and pdi.DOMAIN\_NAME not in (''NNO\_DOMAIN'', ''SLTX\_LAB\_DOMAIN'', ''ORPHAN\_DOMAIN'')

),

all\_seg\_in\_NNO\_w\_prefix as

( select distinct si.circ\_inst\_id

from vzwnet.circ\_DOMAIN\_MAP pdm, vzwnet.circ\_inst si, vzwnet.domain\_inst di

where si.circ\_inst\_id = pdm.circ\_inst\_id

and di.DOMAIN\_INST\_ID = pdm.DOMAIN\_INST\_ID

and di.DOMAIN\_NAME = ''NNO\_DOMAIN''

and si.circ\_hum\_id like ''NNO %''

and si.status =''Live''

),

all\_seg\_in\_NNO\_w\_no\_prefix as

(

select distinct si.circ\_inst\_id

from vzwnet.circ\_DOMAIN\_MAP pdm, vzwnet.circ\_INST si, vzwnet.domain\_inst di

where si.circ\_inst\_id = pdm.circ\_inst\_id

and di.DOMAIN\_INST\_ID = pdm.DOMAIN\_INST\_ID

and di.DOMAIN\_NAME = ''NNO\_DOMAIN''

and si.circ\_hum\_id not like ''NNO %''

and si.status =''Live''

),

not\_exists\_in\_other\_domains as

( select distinct circ\_inst\_id

from all\_seg\_in\_NNO\_w\_no\_prefix

minus

select distinct pdm.circ\_inst\_id

from vzwnet.circ\_DOMAIN\_MAP pdm, LEAF\_DOMAINS\_MINUS\_NNO di

where pdm.domain\_inst\_id = di.DOMAIN\_INST\_ID

)

, all\_nno\_segments as

( select distinct circ\_inst\_id

from all\_seg\_in\_NNO\_w\_prefix

union

select distinct circ\_inst\_id

from not\_exists\_in\_other\_domains

),

OTHER\_DOMAIN\_SEGMENTS as

(

select distinct c1.circ\_inst\_id

from vzwnet.circ\_inst c1

where c1.status=''Live''

MINUS

select circ\_inst\_id

from all\_nno\_segments

)

select drr.AREA, (select dr.REGION from DOMAINS\_LEAF\_REPORTING dr where dr.MARKET = drr.MARKET ) REGION, drr.MARKET, count( ci.circ\_inst\_id )

from OTHER\_DOMAIN\_SEGMENTS odp,vzwnet.circ\_inst ci, vzwnet.site\_inst siz, vzwnet.site\_inst sia, vzwnet.circ\_domain\_map d, DOMAINS\_LEAF\_REPORTING drr,

xng\_apps.nonlive\_site\_status\_chg\_date nss

where

odp.circ\_inst\_id = ci.circ\_inst\_id

and ci.CIRC\_INST\_ID = d.CIRC\_INST\_ID

and d.DOMAIN\_INST\_ID = drr.DOMAIN\_INST\_ID

and ci.z\_site\_id=siz.site\_inst\_id

and ci.a\_site\_id=sia.site\_inst\_id

and siz.site\_inst\_id=nss.site\_inst\_id

and siz.status not in (''Live'', ''Standby'', ''Commissioning'')

and drr.AREA <> ''NNO''

group by (drr.AREA, drr.MARKET)

order by drr.AREA, drr.MARKET';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into table: ' || table\_name || '\_SEG';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

xng\_reports.watchdog.updateprocessend(processName, 'STATUS\_FAILURE', 'N');

RAISE;

END;

sqlStmt := 'update ' || table\_name || ' l set (l.AREA, l.REGION, l.SEG\_CNT ) = (select l1.AREA, l1.REGION, l1.SEG\_CNT from ' || table\_name || '\_SEG l1 where l1.market = l.market) WHERE EXISTS ( SELECT 1 FROM ' || table\_name || '\_SEG l2 WHERE l2.market = l.market)';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t update table: ' || table\_name || '\_SEG';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

xng\_reports.watchdog.updateprocessend(processName, 'STATUS\_FAILURE', 'N');

RAISE;

END;

EXECUTE IMMEDIATE 'truncate table ' || table\_name || '\_PATH reuse storage';

commit;

sqlStmt := 'insert into ' || table\_name || '\_PATH

WITH

LEAF\_DOMAINS\_MINUS\_NNO as

( select pdi.\* from vzwnet.domain\_inst pdi, vzwnet.domain\_inst cdi

where pdi.DOMAIN\_INST\_ID = cdi.PARENT\_DOMAIN\_INST\_ID (+)

and cdi.DOMAIN\_INST\_ID is null

and pdi.DOMAIN\_NAME not in (''NNO\_DOMAIN'', ''SLTX\_LAB\_DOMAIN'', ''ORPHAN\_DOMAIN'')

),

all\_path\_in\_NNO\_w\_prefix as

( select distinct si.circ\_path\_inst\_id

from vzwnet.path\_DOMAIN\_MAP pdm, vzwnet.circ\_path\_INST si, vzwnet.domain\_inst di

where si.circ\_path\_inst\_id = pdm.circ\_path\_inst\_id

and di.DOMAIN\_INST\_ID = pdm.DOMAIN\_INST\_ID

and di.DOMAIN\_NAME = ''NNO\_DOMAIN''

and si.circ\_path\_hum\_id like ''NNO %''

and si.status =''Live''

),

all\_path\_in\_NNO\_w\_no\_prefix as

(

select distinct si.circ\_path\_inst\_id

from vzwnet.path\_DOMAIN\_MAP pdm, vzwnet.circ\_path\_INST si, vzwnet.domain\_inst di

where si.circ\_path\_inst\_id = pdm.circ\_path\_inst\_id

and di.DOMAIN\_INST\_ID = pdm.DOMAIN\_INST\_ID

and di.DOMAIN\_NAME = ''NNO\_DOMAIN''

and si.circ\_path\_hum\_id not like ''NNO %''

and si.status =''Live''

),

not\_exists\_in\_other\_domains as

( select distinct circ\_path\_inst\_id

from all\_path\_in\_NNO\_w\_no\_prefix

minus

select distinct pdm.circ\_path\_inst\_id

from vzwnet.path\_DOMAIN\_MAP pdm, LEAF\_DOMAINS\_MINUS\_NNO di

where pdm.domain\_inst\_id = di.DOMAIN\_INST\_ID

), all\_nno\_circ\_paths as

( select distinct circ\_path\_inst\_id

from all\_path\_in\_NNO\_w\_prefix

union

select distinct circ\_path\_inst\_id

from not\_exists\_in\_other\_domains

) ,

OTHER\_DOMAIN\_PATHS as

(

select distinct c1.circ\_path\_inst\_id

from vzwnet.circ\_path\_inst c1

where c1.status = ''Live''

MINUS

select circ\_path\_inst\_id

from all\_nno\_circ\_paths

)

select drr.AREA, (select dr.REGION from DOMAINS\_LEAF\_REPORTING dr where dr.MARKET = drr.MARKET ) REGION, drr.MARKET, count(cpi.circ\_path\_inst\_id)

from OTHER\_DOMAIN\_PATHS odp, vzwnet.circ\_path\_inst cpi, vzwnet.path\_domain\_map d, DOMAINS\_LEAF\_REPORTING drr,

vzwnet.site\_inst siz, vzwnet.site\_inst sia,

xng\_apps.nonlive\_site\_status\_chg\_date nss

where cpi.circ\_path\_inst\_id=odp.circ\_path\_inst\_id

and cpi.CIRC\_PATH\_INST\_ID = d.CIRC\_PATH\_INST\_ID

and d.DOMAIN\_INST\_ID = drr.DOMAIN\_INST\_ID

and cpi.z\_side\_site\_id=siz.site\_inst\_id

and cpi.a\_side\_site\_id=sia.site\_inst\_id

and siz.site\_inst\_id=nss.site\_inst\_id

and siz.status not in (''Live'', ''Standby'', ''Commissioning'')

and drr.AREA <> ''NNO''

group by (drr.AREA, drr.MARKET)

order by drr.AREA, drr.MARKET';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into table: ' || table\_name || '\_PATH';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

xng\_reports.watchdog.updateprocessend(processName, 'STATUS\_FAILURE', 'N');

RAISE;

END;

sqlStmt := 'update ' || table\_name || ' l set (l.AREA, l.REGION, l.PATH\_CNT ) = (select l1.AREA, l1.REGION, l1.PATH\_CNT from ' || table\_name || '\_PATH l1 where l1.market = l.market) WHERE EXISTS ( SELECT 1 FROM ' || table\_name || '\_PATH l2 WHERE l2.market = l.market)';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t update table: ' || table\_name || '\_PATH';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

xng\_reports.watchdog.updateprocessend(processName, 'STATUS\_FAILURE', 'N');

RAISE;

END;

sqlStmt := 'insert into ' || table\_name || ' SELECT L.AREA, null, null, SUM(L.EQUIP\_CNT), SUM(L.SEG\_CNT), SUM(L.PATH\_CNT) FROM ' || table\_name || ' L group by rollup(L.area)';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert sum values into table: ' || table\_name;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

xng\_reports.watchdog.updateprocessend(processName, 'STATUS\_FAILURE', 'N');

RAISE;

END;

xng\_reports.watchdog.updateprocessend(processName, 'STATUS\_SUCCESS', 'Y');

xng\_reports.watchdog.populate\_last\_success\_dt\_proc(processName);

END insert\_objs;

END LIVE\_OBJS\_NL\_SITES;

/

--------------------------------------------------------

-- DDL for Package Body LUCENT\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."LUCENT\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: lucent\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 2/4/2011 1. Created this package body.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE auditXngVsProspect

IS

BEGIN

BEGIN

auditLucentBTSTerminations();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in auditLucentBTSTerminations(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

generateLucentSwitchSummary();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in generateLucentSwitchSummary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

generateLucentRegSummary();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in generateLucentRegSummary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

END;

PROCEDURE auditLucentBTSTerminations

IS

sql\_stmt varchar2(32676);

-- One Live and N non Live match

CURSOR t1\_nl\_match IS

SELECT sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num, ne\_ds1\_num

FROM Lucent\_ne\_vs\_xng\_audit\_wrk

WHERE match\_code='BOTH'

AND xng\_path\_status='Live'

AND termination\_type = 'T1';

-- One Live and N non Live match

CURSOR ebh\_nl\_match IS

SELECT sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num

FROM Lucent\_ne\_vs\_xng\_audit\_wrk

WHERE match\_code='BOTH'

AND xng\_path\_status='Live'

AND termination\_type = 'Ethernet';

cursor upd\_ethernet\_both\_multiples is

select sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num, count(1)

from Lucent\_ne\_vs\_xng\_audit\_wrk a

where a.MATCH\_CODE = 'BOTH'

and a.TERMINATION\_TYPE = 'Ethernet'

and xng\_path\_status = 'Live'

group by sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num

having count(1) > 1

;

cursor upd\_t1\_both\_multiples is

select sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num, ne\_ds1\_num, count(1)

from Lucent\_ne\_vs\_xng\_audit\_wrk a

where a.MATCH\_CODE = 'BOTH'

and a.TERMINATION\_TYPE = 'T1'

and xng\_path\_status = 'Live'

group by sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num, ne\_ds1\_num

having count(1) > 1

;

cursor upd\_ethernet\_xngonly\_multiples is

select sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num, count(1)

from Lucent\_ne\_vs\_xng\_audit\_wrk a

where a.MATCH\_CODE = 'Xng Only'

and a.TERMINATION\_TYPE = 'Ethernet'

and xng\_path\_status = 'Live'

group by sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num

having count(1) > 1

;

cursor upd\_t1\_xngonly\_multiples is

select sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num, ne\_ds1\_num, count(1)

from Lucent\_ne\_vs\_xng\_audit\_wrk a

where a.MATCH\_CODE = 'Xng Only'

and a.TERMINATION\_TYPE = 'T1'

and xng\_path\_status = 'Live'

group by sys\_id, ecp\_sid, ne\_cell\_num, ne\_urc\_num, ne\_ds1\_num

having count(1) > 1

;

--JC

cursor update\_lucent\_wrk\_no\_csr\_port is

WITH

LUXA\_PATHS as

(

select lu.CIRC\_PATH\_INST\_ID

from LUCENT\_NE\_VS\_XNG\_AUDIT\_wrk lu

where lu.TERMINATION\_TYPE = 'Ethernet'

AND lu.BANDWIDTH LIKE '%Mbps'

and lu.match\_code = 'BOTH'

and lu.XNG\_PATH\_STATUS = 'Live'

)

,

PATHS\_W\_CSR as

(

select cpi.CIRC\_PATH\_INST\_ID

from LUXA\_PATHS cpi

join VZWNET.circ\_path\_element cpe

on cpi.CIRC\_PATH\_INST\_ID = cpe.CIRC\_PATH\_INST\_ID

join vzwnet.epa p

on CPE.PORT\_INST\_ID = P.PORT\_INST\_ID

join VZWNET.equip\_inst ei

on p.equip\_INST\_ID = ei.equip\_INST\_ID

and ei.type = 'CSR'

)

select LUXA.CIRC\_PATH\_INST\_ID

from LUXA\_PATHS luxa

MINUS

select pc.CIRC\_PATH\_INST\_ID

from PATHS\_W\_CSR pc;

BEGIN

EXECUTE IMMEDIATE 'drop index LNVXA\_WRK\_IDX';

EXECUTE IMMEDIATE 'drop index LNVXA\_WRK\_BTS\_URC\_DS1\_IDX ';

EXECUTE IMMEDIATE 'drop index LNVXA\_WRK\_PATHINST\_STA\_TYP\_IDX';

sql\_stmt:= 'truncate table '||AUDIT\_TBL;

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in auditXngVsProspect(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

commit;

sql\_stmt:='insert into '|| AUDIT\_TBL ||'

WITH

XNG\_SPANS\_PER\_SWITCH as

(

select distinct vcp.switch\_id,

vcp.cell\_number xng\_cell\_num,

vcp.urc\_crc as xng\_urc\_num,

vcp.ds1\_number as xng\_ds1\_num,

vcp.circ\_path\_inst\_id circ\_path\_inst\_id,

vcp.PATH\_NAME,

vcp.xng\_path\_status,

vcp.xng\_path\_type,

vcp.parse\_status

from VZW\_LUCENT\_CELL\_PATHS vcp

)

,

BTS\_ETHER\_TERMINATIONS as

(

select ne.\*, esm.switch\_id, ''Ethernet'' termination\_type

from PROS\_ECP\_CELL\_EBH\_MAP ne

join ECP\_SWITCH\_MAP esm

on esm.SYS\_ID = ne.sys\_id

and esm.ecp\_sid = ne.ecp\_sid

)

,

BTS\_T1\_TERMINATIONS as

(

select ne.\*, esm.switch\_id, ''T1'' termination\_type

from PROS\_ECP\_CELL\_DS1\_MAP ne

join ECP\_SWITCH\_MAP esm

on esm.SYS\_ID = ne.sys\_id

and esm.ecp\_sid = ne.ecp\_sid

)

,

ETHERNET\_AUDIT AS

(

SELECT bet.bts\_name ne\_bts\_name

, bet.cell\_type ne\_cell\_type

, bet.sys\_id, bet.ecp\_sid

, bet.cell\_number ne\_cell\_num

, bet.urc\_num ne\_urc\_num

, null ne\_ds1\_num

, bet.version ne\_version

, bet.switch\_id

, bet.service

, bet.termination\_type

, cpi.xng\_cell\_num

, cpi.xng\_urc\_num

, cpi.xng\_ds1\_num

, cpi.circ\_path\_inst\_id

, cpi.path\_name

, cpi.xng\_path\_status

, cpi.xng\_path\_type

, vcpi.bandwidth

, CASE

WHEN cpi.switch\_id IS NULL

THEN ''NE Only''

WHEN cpi.xng\_ds1\_num IS not NULL and upper(vcpi.bandwidth) like ''%MBPS%''

THEN ''NE Only''

WHEN cpi.xng\_ds1\_num IS not NULL and upper(vcpi.bandwidth) not like ''%MBPS%''

THEN ''NE Only''

WHEN cpi.xng\_ds1\_num IS NULL and upper(vcpi.bandwidth) not like ''%MBPS%''

THEN ''NE Only''

WHEN cpi.xng\_ds1\_num IS NULL and upper(vcpi.bandwidth) like ''%MBPS%''

THEN ''BOTH''

END match\_code

, CASE

WHEN cpi.switch\_id IS NULL

THEN ''NE Only''

WHEN cpi.xng\_ds1\_num IS not NULL and upper(vcpi.bandwidth) like ''%MBPS%''

THEN ''Pathname Invalid Format''

WHEN cpi.xng\_ds1\_num IS not NULL and upper(vcpi.bandwidth) not like ''%MBPS%''

THEN ''Pathname Invalid Format, Path Invalid Bandwidth''

WHEN cpi.xng\_ds1\_num IS NULL and upper(vcpi.bandwidth) not like ''%MBPS%''

THEN ''Path Invalid Bandwidth''

WHEN cpi.xng\_ds1\_num IS NULL and upper(vcpi.bandwidth) like ''%MBPS%''

THEN ''BOTH''

else parse\_status

END match\_status

FROM BTS\_ETHER\_TERMINATIONS bet

LEFT OUTER JOIN XNG\_SPANS\_PER\_SWITCH cpi

ON bet.switch\_id = cpi.switch\_id

AND bet.cell\_number = cpi.xng\_cell\_num

AND bet.urc\_num = cpi.xng\_urc\_num

LEFT OUTER JOIN vzwnet.circ\_path\_inst vcpi

on cpi.CIRC\_PATH\_INST\_ID = vcpi.CIRC\_PATH\_INST\_ID

)

,

T1\_AUDIT AS

(

SELECT bet.bts\_name ne\_bts\_name

, bet.cell\_type ne\_cell\_type

, bet.sys\_id, bet.ecp\_sid

, bet.cell\_number ne\_cell\_num

, bet.urc\_num ne\_urc\_num

, bet.ds1\_num ne\_ds1\_num

, bet.version ne\_version

, bet.switch\_id

, bet.service

, bet.termination\_type

, cpi.xng\_cell\_num

, cpi.xng\_urc\_num

, cpi.xng\_ds1\_num

, cpi.circ\_path\_inst\_id

, cpi.path\_name

, cpi.xng\_path\_status

, cpi.xng\_path\_type

, vcpi.bandwidth

, CASE

WHEN cpi.switch\_id IS NULL

THEN ''NE Only''

WHEN upper(vcpi.bandwidth) not like ''%DS1%''

THEN ''NE Only''

WHEN upper(vcpi.bandwidth) like ''%DS1%''

THEN ''BOTH''

END match\_code

, CASE

WHEN cpi.switch\_id IS NULL

THEN null -- ''NE Only''

WHEN upper(vcpi.bandwidth) not like ''%DS1%''

THEN ''Path Invalid Bandwidth''

WHEN upper(vcpi.bandwidth) like ''%DS1%''

THEN null -- ''BOTH''

else parse\_status

END match\_status

FROM BTS\_T1\_TERMINATIONS bet

LEFT OUTER JOIN XNG\_SPANS\_PER\_SWITCH cpi

ON bet.switch\_id = cpi.switch\_id

AND bet.cell\_number = cpi.xng\_cell\_num

AND bet.urc\_num = cpi.xng\_urc\_num

AND bet.ds1\_num = cpi.xng\_ds1\_num

LEFT OUTER JOIN vzwnet.circ\_path\_inst vcpi

on cpi.CIRC\_PATH\_INST\_ID = vcpi.CIRC\_PATH\_INST\_ID

)

,

NE\_ONLY\_AND\_BOTH AS

(

SELECT \*

FROM ETHERNET\_AUDIT ea

UNION

SELECT \*

FROM T1\_AUDIT t1a

)

,

XNG\_ONLY\_PATHS AS

(

SELECT CIRC\_PATH\_INST\_ID

FROM XNG\_SPANS\_PER\_SWITCH cpi

MINUS

SELECT CIRC\_PATH\_INST\_ID

FROM NE\_ONLY\_AND\_BOTH

)

SELECT a.\*, TRUNC (SYSDATE) AS audit\_date

FROM NE\_ONLY\_AND\_BOTH A

UNION

SELECT null ne\_bts\_name

, null ne\_cell\_type

, null sys\_id

, null ecp\_sid

, null ne\_cell\_num

, null ne\_urc\_num

, null ne\_ds1\_num

, null ne\_version

, cpi.switch\_id

, null service

, null termination\_type

, cpi.xng\_cell\_num

, cpi.xng\_urc\_num

, cpi.xng\_ds1\_num

, cpi.circ\_path\_inst\_id

, cpi.path\_name

, cpi.xng\_path\_status

, cpi.xng\_path\_type

, vcpi.bandwidth

, ''Xng Only'' match\_code

, parse\_status match\_status

, TRUNC (SYSDATE) AS audit\_date

FROM XNG\_ONLY\_PATHS a

join XNG\_SPANS\_PER\_SWITCH cpi

on a.CIRC\_PATH\_INST\_ID = cpi.CIRC\_PATH\_INST\_ID

LEFT OUTER JOIN vzwnet.circ\_path\_inst vcpi

on cpi.CIRC\_PATH\_INST\_ID = vcpi.CIRC\_PATH\_INST\_ID

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='CREATE INDEX LNVXA\_WRK\_IDX ON '|| AUDIT\_TBL ||'

(sys\_id, ecp\_sid, NE\_CELL\_NUM, NE\_URC\_NUM, NE\_DS1\_NUM)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='CREATE INDEX LNVXA\_WRK\_BTS\_URC\_DS1\_IDX ON '|| AUDIT\_TBL ||'

(SWITCH\_ID, XNG\_CELL\_NUM, XNG\_URC\_NUM, XNG\_DS1\_NUM)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='CREATE INDEX LNVXA\_WRK\_PATHINST\_STA\_TYP\_IDX ON '|| AUDIT\_TBL ||'

(CIRC\_PATH\_INST\_ID, XNG\_PATH\_STATUS, XNG\_PATH\_TYPE)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- Xng Only and BW = DS1 then it is T1

sql\_stmt:=' update '|| AUDIT\_TBL ||' lt1 set lt1.TERMINATION\_TYPE=''T1''

where exists (

select lt.circ\_path\_inst\_id

from '|| AUDIT\_TBL ||' lt, vzwnet.circ\_path\_inst cpi

where lt.MATCH\_CODE=''Xng Only''

and cpi.CIRC\_PATH\_INST\_ID = lt.circ\_path\_inst\_id

and cpi.BANDWIDTH=''DS1''

and lt1.circ\_path\_inst\_id = lt.circ\_path\_inst\_id

)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

-- Xng Only and BW <> DS1 then it is Ethernet

sql\_stmt:=' update '|| AUDIT\_TBL ||' lt1 set lt1.TERMINATION\_TYPE=''Ethernet''

where exists (

select lt.circ\_path\_inst\_id

from '|| AUDIT\_TBL ||' lt, vzwnet.circ\_path\_inst cpi

where lt.MATCH\_CODE=''Xng Only''

and cpi.CIRC\_PATH\_INST\_ID = lt.circ\_path\_inst\_id

and cpi.BANDWIDTH<>''DS1''

and lt1.circ\_path\_inst\_id = lt.circ\_path\_inst\_id

)';

EXECUTE IMMEDIATE sql\_stmt;

commit;

FOR cursorRec IN upd\_ethernet\_both\_multiples LOOP

UPDATE Lucent\_ne\_vs\_xng\_audit\_wrk u

set match\_status = match\_status || ', Multiple spans found in Xng for same tokens'

, match\_code='NE Only'

WHERE --rowid = cursorRec.ROWID

u.sys\_id = cursorRec.sys\_id

and u.ecp\_sid = cursorRec.ecp\_sid

and u.ne\_cell\_num = cursorRec.ne\_cell\_num

and u.ne\_urc\_num = cursorRec.ne\_urc\_num

;

END LOOP;

commit;

FOR cursorRec IN upd\_t1\_both\_multiples LOOP

UPDATE Lucent\_ne\_vs\_xng\_audit\_wrk u

set match\_status = match\_status || ', Multiple spans found in Xng for same tokens'

, match\_code='NE Only'

WHERE --rowid = cursorRec.ROWID

u.sys\_id = cursorRec.sys\_id

and u.ecp\_sid = cursorRec.ecp\_sid

and u.ne\_cell\_num = cursorRec.ne\_cell\_num

and u.ne\_urc\_num = cursorRec.ne\_urc\_num

and u.ne\_ds1\_num = cursorRec.ne\_ds1\_num

;

END LOOP;

commit;

FOR cursorRec IN upd\_ethernet\_xngonly\_multiples LOOP

UPDATE Lucent\_ne\_vs\_xng\_audit\_wrk u

set match\_status = ', Multiple spans found in Xng for same tokens'

WHERE --rowid = cursorRec.ROWID

u.sys\_id = cursorRec.sys\_id

and u.ecp\_sid = cursorRec.ecp\_sid

and u.ne\_cell\_num = cursorRec.ne\_cell\_num

and u.ne\_urc\_num = cursorRec.ne\_urc\_num

;

END LOOP;

commit;

FOR cursorRec IN upd\_t1\_xngonly\_multiples LOOP

UPDATE Lucent\_ne\_vs\_xng\_audit\_wrk u

set match\_status = nvl2(match\_status, 'Multiple spans found in Xng for same tokens,'||match\_status, 'Multiple spans found in Xng for same tokens')

WHERE --rowid = cursorRec.ROWID

u.sys\_id = cursorRec.sys\_id

and u.ecp\_sid = cursorRec.ecp\_sid

and u.ne\_cell\_num = cursorRec.ne\_cell\_num

and u.ne\_urc\_num = cursorRec.ne\_urc\_num

and u.ne\_ds1\_num = cursorRec.ne\_ds1\_num

;

END LOOP;

commit;

sql\_stmt:='update Lucent\_ne\_vs\_xng\_audit\_wrk u

set match\_code = ''Xng Only''

where u.XNG\_PATH\_TYPE=''AMPS''

'

;

EXECUTE IMMEDIATE sql\_stmt;

commit;

FOR cursorRec IN t1\_nl\_match LOOP

UPDATE Lucent\_ne\_vs\_xng\_audit\_wrk mnvxa

SET match\_status = nvl2(match\_status, 'Live Path Exists, '||match\_status ,'Live Path Exists')

WHERE mnvxa.sys\_id = cursorRec.sys\_id

AND mnvxa.ecp\_sid = cursorRec.ecp\_sid

AND mnvxa.ne\_cell\_num = cursorRec.ne\_cell\_num

AND mnvxa.ne\_urc\_num = cursorRec.ne\_urc\_num

AND mnvxa.ne\_ds1\_num = cursorRec.ne\_ds1\_num

AND mnvxa.termination\_type='T1'

AND mnvxa.xng\_path\_status <> 'Live'

-- AND mnvxa.match\_code='BOTH'

;

END LOOP;

COMMIT;

FOR cursorRec IN ebh\_nl\_match LOOP

UPDATE Lucent\_ne\_vs\_xng\_audit\_wrk mnvxa

SET match\_status = nvl2(match\_status, 'Live Path Exists, '||match\_status ,'Live Path Exists')

WHERE mnvxa.sys\_id = cursorRec.sys\_id

AND mnvxa.ecp\_sid = cursorRec.ecp\_sid

AND mnvxa.ne\_cell\_num = cursorRec.ne\_cell\_num

AND mnvxa.ne\_urc\_num = cursorRec.ne\_urc\_num

AND mnvxa.termination\_type='Ethernet'

AND mnvxa.xng\_path\_status <> 'Live'

--AND mnvxa.match\_code='BOTH'

;

END LOOP;

COMMIT;

--JC

FOR cursorRec IN update\_lucent\_wrk\_no\_csr\_port LOOP

UPDATE Lucent\_ne\_vs\_xng\_audit\_wrk lu

SET MATCH\_CODE = 'NE Only',

match\_status =

case

when MATCH\_STATUS = 'BOTH'

then 'No CSR port on the e-pipe'

else

nvl2(match\_status, 'No CSR port on the e-pipe, '||match\_status, 'No CSR port on the e-pipe')

end

WHERE

lu.CIRC\_PATH\_INST\_ID = cursorRec.CIRC\_PATH\_INST\_ID

;

END LOOP;

commit;

--JC

dbms\_output.put\_line('Lucent audit completed successfully');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in auditLucentBTSTerminations(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end auditLucentBTSTerminations;

procedure generateLucentRegSummary as

sqlstmt varchar2(32767);

TBL\_NAME varchar2(30) := 'LUCENT\_AUDIT\_REG\_SUMM\_WRK';

BEGIN

sqlstmt := 'truncate table '||TBL\_NAME;

begin

execute immediate sqlstmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlstmt);

dbms\_output.put\_line(SubStr('Error in generateLucentRegSummary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

sqlstmt := ' insert into '||TBL\_NAME ||' columns(area, region,t1\_discovered, t1\_matched\_live, t1\_matched\_nonlive, ebh\_discovered,

ebh\_matched\_live, ebh\_matched\_nonlive, t1\_comp, ebh\_CSR\_matched\_live, ebh\_CSR\_matched\_nonlive, ebh\_csr\_reported,

ebh\_csr\_comp,overall\_csr\_comp,ebh\_comp,overall\_comp)

select drr.area, drr.region, sum(NVL (t1\_discovered, 0)) t1\_discovered, sum(NVL (t1\_matched\_live, 0)) t1\_matched\_live,

sum(NVL (t1\_matched\_nonlive, 0)) t1\_matched\_nonlive, sum(NVL (ebh\_discovered, 0)) ebh\_discovered,

sum(NVL (ebh\_matched\_live, 0)) ebh\_matched\_live, sum(NVL (ebh\_matched\_nonlive, 0)) ebh\_matched\_nonlive,

case when SUM (NVL (t1\_discovered, 0)) - SUM (NVL (t1\_matched\_nonlive, 0)) <> 0

then round (SUM (NVL (t1\_matched\_live, 0)) / (SUM (NVL (t1\_discovered, 0)) - SUM (NVL (t1\_matched\_nonlive, 0)))\*100, 2)

else 0

end t1\_comp,

sum(NVL (ebh\_CSR\_matched\_live, 0)) ebh\_CSR\_matched\_live,

sum(NVL (ebh\_CSR\_matched\_nonlive, 0)) ebh\_CSR\_matched\_nonlive, sum(NVL (ebh\_csr\_reported, 0)) ebh\_csr\_reported,

case when SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_CSR\_matched\_nonlive, 0)) <> 0

then round (SUM (NVL (ebh\_csr\_matched\_live, 0)) / (SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_CSR\_matched\_nonlive, 0)))\*100, 2)

else 0

end ebh\_csr\_comp,

case when SUM(nvl(t1\_discovered,0) + nvl(ebh\_discovered,0)) <> 0

then round (((sum (nvl(t1\_matched\_live,0)+nvl(ebh\_csr\_matched\_live,0)))/(SUM(nvl(t1\_discovered,0) + nvl(ebh\_discovered,0))- SUM(NVL (ebh\_CSR\_matched\_nonlive, 0)+NVL (t1\_matched\_nonlive, 0))))\*100,2)

else 0

end overall\_csr\_comp,

case when SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_matched\_nonlive, 0)) <> 0

then round (SUM (NVL (ebh\_matched\_live, 0)) / (SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_matched\_nonlive, 0)))\*100, 2)

else 0

end ebh\_comp,

case when SUM(nvl(t1\_discovered,0) + nvl(ebh\_discovered,0)) <> 0

then round (((sum (nvl(t1\_matched\_live,0)+nvl(ebh\_matched\_live,0)))/(SUM(nvl(t1\_discovered,0) + nvl(ebh\_discovered,0))- SUM(NVL (ebh\_matched\_nonlive, 0)+NVL (t1\_matched\_nonlive, 0))))\*100,2)

else 0

end overall\_comp

from xng\_reports.LUCENT\_AUDIT\_SWITCH\_SUMM\_WRK mrm

right outer join domains\_regional\_reporting drr on DRR.REGION = mrm.region

where area not in (''NNO'', ''OSS'')

group by rollup (drr.area, drr.region)

order by drr.area, drr.region ';

execute immediate sqlstmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlstmt);

dbms\_output.put\_line(SubStr('Error in auditLucentBTSTerminations(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

procedure generateLucentSwitchSummary as

sqlstmt varchar2(32767);

TBL\_NAME varchar2(30) := 'LUCENT\_AUDIT\_SWITCH\_SUMM\_WRK';

BEGIN

sqlstmt := 'drop table '||TBL\_NAME;

begin

execute immediate sqlstmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlstmt);

dbms\_output.put\_line(SubStr('Error in generateLucentRegSummary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

sqlstmt := ' create table '||TBL\_NAME ||' as

WITH

ECPs as

(

SELECT \*

FROM xng\_reports.ecp\_switch\_map

)

,

ecp\_region\_map AS

(

SELECT DISTINCT cdmp.area, cdmp.region, switch\_id, cdmp.leaf\_domain\_name

FROM xng\_reports.clli\_domain\_map\_v cdmp

JOIN ECPs m

ON SUBSTR (switch\_id, 1, 6) = SUBSTR (cdmp.clli, 1, 6)

WHERE cdmp.area <> ''NNO''

),

t1\_discovered AS

(

SELECT DISTINCT switch\_id, nvxa.ecp\_sid, nvxa.ds1\_num, NVXA.CELL\_NUMBER, NVXA.URC\_NUM

FROM xng\_reports.PROS\_ECP\_CELL\_DS1\_MAP nvxa

join ECPs e

on nvxa.sys\_id = e.sys\_id and nvxa.ecp\_sid = e.ecp\_sid

),

t1\_discovered\_cnt AS

(

SELECT DISTINCT switch\_id, COUNT (1) t1\_discovered

FROM t1\_discovered

group by switch\_id

),

ethernet\_discovered AS

(

SELECT DISTINCT switch\_id, nvxa.ecp\_sid, NVXA.CELL\_NUMBER, NVXA.URC\_NUM

FROM xng\_reports.PROS\_ECP\_CELL\_EBH\_MAP nvxa

join ECPs e

on nvxa.sys\_id = e.sys\_id and nvxa.ecp\_sid = e.ecp\_sid

),

ethernet\_discovered\_cnt AS

(

SELECT DISTINCT switch\_id, count(1) ebh\_discovered

FROM ethernet\_discovered

group by switch\_id

),

t1\_matched\_live as

(

SELECT DISTINCT nvxa.switch\_id, nvxa.NE\_CELL\_NUM, nvxa.NE\_URC\_NUM

, nvxa.NE\_DS1\_NUM

FROM xng\_reports.Lucent\_ne\_vs\_xng\_audit\_wrk nvxa

WHERE nvxa.termination\_type = ''T1''

AND nvxa.match\_code IN (''BOTH'')

and nvxa.XNG\_PATH\_STATUS = ''Live''

),

t1\_matched\_live\_cnt AS

(

SELECT switch\_id, COUNT (1) t1\_matched\_live

FROM t1\_matched\_live

GROUP BY switch\_id

)

,

ethernet\_matched\_live as

(

SELECT DISTINCT nvxa.switch\_id, nvxa.NE\_CELL\_NUM, nvxa.NE\_URC\_NUM

FROM xng\_reports.Lucent\_ne\_vs\_xng\_audit\_wrk nvxa

WHERE nvxa.termination\_type = ''Ethernet''

AND nvxa.match\_code IN (''BOTH'')

and nvxa.XNG\_PATH\_STATUS = ''Live''

)

,

ethernet\_matched\_live\_cnt AS

(

SELECT switch\_id, COUNT (1) ebh\_matched\_live

FROM ethernet\_matched\_live

GROUP BY switch\_id

)

,

t1\_matched\_n\_live\_cnt AS

(

SELECT switch\_id, COUNT (1) t1\_matched\_n\_live

FROM (

-- good matched non live minus all live ones

SELECT nvxa.switch\_id, nvxa.NE\_CELL\_NUM, nvxa.NE\_URC\_NUM

, nvxa.NE\_DS1\_NUM

FROM xng\_reports.Lucent\_ne\_vs\_xng\_audit\_wrk nvxa

WHERE nvxa.termination\_type = ''T1''

AND nvxa.match\_code IN (''BOTH'')

and nvxa.XNG\_PATH\_STATUS <> ''Live''

minus

SELECT nvxa.switch\_id, nvxa.NE\_CELL\_NUM, nvxa.NE\_URC\_NUM

, nvxa.NE\_DS1\_NUM

from t1\_matched\_live nvxa

)

GROUP BY switch\_id

)

,

ethernet\_matched\_n\_live\_cnt AS

(

SELECT switch\_id, COUNT (1) ebh\_matched\_n\_live

FROM (

-- good matched non live minus all live ones

SELECT nvxa.switch\_id, nvxa.NE\_CELL\_NUM, nvxa.NE\_URC\_NUM

FROM xng\_reports.Lucent\_ne\_vs\_xng\_audit\_wrk nvxa

WHERE nvxa.termination\_type = ''Ethernet''

AND nvxa.match\_code IN (''BOTH'')

and nvxa.XNG\_PATH\_STATUS <> ''Live''

minus

SELECT nvxa.switch\_id, nvxa.NE\_CELL\_NUM, nvxa.NE\_URC\_NUM

from ethernet\_matched\_live nvxa

)

GROUP BY switch\_id

)

,

--JC

ethernet\_csr\_matched\_live as

(

select DISTINCT lu.switch\_id, lu.NE\_CELL\_NUM, lu.NE\_URC\_NUM

from LUCENT\_NE\_VS\_XNG\_AUDIT\_WRK lu

join VZWNET.CIRC\_PATH\_INST cpi on LU.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

where lu.TERMINATION\_TYPE = ''Ethernet''

AND lu.BANDWIDTH LIKE ''%Mbps''

and lu.match\_code = ''BOTH''

and lu.match\_status = ''BOTH''

and lu.XNG\_PATH\_STATUS = ''Live''

and EI.TYPE = ''CSR''

GROUP BY lu.switch\_id, lu.NE\_CELL\_NUM, lu.NE\_URC\_NUM

)

,

ethernet\_csr\_matched\_live\_cnt AS

(

SELECT switch\_id, COUNT (1) ebh\_csr\_matched\_live

FROM ethernet\_csr\_matched\_live

GROUP BY switch\_id

)

,

ethernet\_csr\_matched\_non\_live as

(

select DISTINCT lu.switch\_id, lu.NE\_CELL\_NUM, lu.NE\_URC\_NUM

from LUCENT\_NE\_VS\_XNG\_AUDIT\_WRK lu

join VZWNET.CIRC\_PATH\_INST cpi on LU.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

where lu.TERMINATION\_TYPE = ''Ethernet''

AND lu.BANDWIDTH LIKE ''%Mbps''

and lu.match\_code = ''BOTH''

and lu.match\_status = ''BOTH''

and lu.XNG\_PATH\_STATUS <> ''Live''

and EI.TYPE = ''CSR''

GROUP BY lu.switch\_id, lu.NE\_CELL\_NUM, lu.NE\_URC\_NUM

)

,

ethernet\_csr\_matched\_nlv\_cnt AS

(

SELECT switch\_id, COUNT (1) ebh\_csr\_matched\_nonlive

FROM ethernet\_csr\_matched\_non\_live

GROUP BY switch\_id

)

--JC

SELECT mrm.region, mrm.leaf\_domain\_name, mrm.switch\_id,

SUM (NVL (t1\_discovered, 0)) AS t1\_discovered,

SUM (NVL (t1\_matched\_live, 0)) AS t1\_matched\_live,

SUM (NVL (t1\_matched\_n\_live, 0)) AS t1\_matched\_nonlive,

case when SUM (NVL (t1\_discovered, 0)) - SUM (NVL (t1\_matched\_n\_live, 0)) <> 0

then round (SUM (NVL (t1\_matched\_live, 0)) / (SUM (NVL (t1\_discovered, 0)) - SUM (NVL (t1\_matched\_n\_live, 0)))\*100, 2)

else 0

end t1\_comp,

SUM (NVL (ebh\_discovered, 0)) AS ebh\_discovered,

SUM (NVL (ebh\_matched\_live, 0)) AS ebh\_matched\_live,

SUM (NVL (ebh\_matched\_n\_live, 0)) AS ebh\_matched\_nonlive,

SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_matched\_n\_live, 0)) ebh\_reported,

--JC

SUM (NVL (ebh\_csr\_matched\_live, 0)) AS ebh\_CSR\_matched\_live,

SUM (NVL (ebh\_csr\_matched\_nonlive, 0)) AS ebh\_CSR\_matched\_nonlive,

SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_csr\_matched\_nonlive, 0)) ebh\_csr\_reported,

case when SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_CSR\_matched\_nonlive, 0)) <> 0

then round (SUM (NVL (ebh\_csr\_matched\_live, 0)) / (SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_CSR\_matched\_nonlive, 0)))\*100, 2)

else 0

end ebh\_csr\_comp,

case when SUM(nvl(t1\_discovered,0) + nvl(ebh\_discovered,0)) <> 0

then round (((sum (nvl(t1\_matched\_live,0)+nvl(ebh\_csr\_matched\_live,0)))/(SUM(nvl(t1\_discovered,0) + nvl(ebh\_discovered,0))- SUM(NVL (ebh\_CSR\_matched\_nonlive, 0)+NVL (t1\_matched\_n\_live, 0))))\*100,2)

else 0

end overall\_csr\_comp,

--JC

case when SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_matched\_n\_live, 0)) <> 0

then round (SUM (NVL (ebh\_matched\_live, 0)) / (SUM (NVL (ebh\_discovered, 0)) - SUM (NVL (ebh\_matched\_n\_live, 0)))\*100, 2)

else 0

end ebh\_comp,

case when SUM(nvl(t1\_discovered,0) + nvl(ebh\_discovered,0)) <> 0

then round (((sum (nvl(t1\_matched\_live,0)+nvl(ebh\_matched\_live,0)))/(SUM(nvl(t1\_discovered,0) + nvl(ebh\_discovered,0))- SUM(NVL (ebh\_matched\_n\_live, 0)+NVL (t1\_matched\_n\_live, 0))))\*100,2)

else 0

end overall\_comp

FROM ecp\_region\_map mrm

LEFT OUTER JOIN t1\_discovered\_cnt td ON mrm.switch\_id = td.switch\_id

LEFT OUTER JOIN t1\_matched\_live\_cnt t1\_l ON mrm.switch\_id = t1\_l.switch\_id

LEFT OUTER JOIN t1\_matched\_n\_live\_cnt t1\_n\_l ON mrm.switch\_id = t1\_n\_l.switch\_id

LEFT OUTER JOIN ethernet\_discovered\_cnt ed ON mrm.switch\_id = ed.switch\_id

LEFT OUTER JOIN ethernet\_matched\_live\_cnt e\_l ON mrm.switch\_id = e\_l.switch\_id

LEFT OUTER JOIN ethernet\_matched\_n\_live\_cnt e\_n\_l ON mrm.switch\_id = e\_n\_l.switch\_id

--JC

LEFT OUTER JOIN ethernet\_csr\_matched\_live\_cnt e\_c\_l ON mrm.switch\_id = e\_c\_l.switch\_id

LEFT OUTER JOIN ethernet\_csr\_matched\_nlv\_cnt e\_c\_nl ON mrm.switch\_id = e\_c\_nl.switch\_id

--JC

GROUP BY mrm.region, mrm.leaf\_domain\_name, mrm.switch\_id

ORDER BY mrm.region, mrm.leaf\_domain\_name, mrm.switch\_id';

execute immediate sqlstmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlstmt);

dbms\_output.put\_line(SubStr('Error in generateLucentSwitchSummary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

PROCEDURE copy\_lucent\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from lucent\_ne\_vs\_xng\_audit';

execute immediate sqlStmt;

sqlStmt := 'insert into lucent\_ne\_vs\_xng\_audit select \* from lucent\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating lucent\_ne\_vs\_xng\_audit');

sqlStmt := 'truncate table lucent\_audit\_reg\_summ';

execute immediate sqlStmt;

sqlStmt := 'insert into lucent\_audit\_reg\_summ select \* from lucent\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating lucent\_audit\_reg\_summ');

sqlStmt := 'truncate table lucent\_audit\_switch\_summ';

execute immediate sqlStmt;

sqlStmt := 'insert into lucent\_audit\_switch\_summ select \* from lucent\_audit\_switch\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating lucent\_audit\_switch\_summ');

END;

PROCEDURE truncate\_lucent\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from lucent\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

dbms\_output.put\_line('Done deleting lucent\_ne\_vs\_xng\_audit\_wrk');

sqlStmt := 'delete from lucent\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting lucent\_audit\_reg\_summ\_wrk');

sqlStmt := 'delete from lucent\_audit\_switch\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting lucent\_audit\_switch\_summ\_wrk');

END;

PROCEDURE restore\_lucent\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from lucent\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into lucent\_ne\_vs\_xng\_audit\_wrk select \* from lucent\_ne\_vs\_xng\_audit';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating lucent\_ne\_vs\_xng\_audit\_wrk');

sqlStmt := 'delete from lucent\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into lucent\_audit\_reg\_summ\_wrk select \* from lucent\_audit\_reg\_summ';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating lucent\_audit\_reg\_summ\_wrk');

sqlStmt := 'delete from lucent\_audit\_switch\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into lucent\_audit\_switch\_summ\_wrk select \* from lucent\_audit\_switch\_summ';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating lucent\_audit\_switch\_summ\_wrk');

END;

END lucent\_audit;

/

--------------------------------------------------------

-- DDL for Package Body MOTOROLA\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."MOTOROLA\_AUDIT"

IS

-- Purpose: To create motorola audit tables for voice and data spans

FUNCTION SET\_MESSAGE (fullMsg IN VARCHAR2, message IN VARCHAR2)

RETURN VARCHAR2 IS

STATUS VARCHAR2(256);

BEGIN

IF (fullMsg IS NULL) THEN

STATUS := message;

ELSE

STATUS := FULLMSG || ', ' || message;

END IF;

return status;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SUBSTR('Error in SET\_MESSAGE(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end SET\_MESSAGE;

FUNCTION GET\_UBS\_MATCH\_STATUS (DS1\_NUMBER IN NUMBER, BANDWIDTH IN VARCHAR, BTS\_TYPE IN VARCHAR, IST1 IN INTEGER,

isUBS IN INTEGER, parseStats IN VARCHAR) RETURN VARCHAR2 IS

status varchar2(256) := parseStats;

BEGIN

IF IST1 = T1\_TYPE THEN

IF UPPER(bandwidth) NOT LIKE '%DS1%' THEN

STATUS := SET\_MESSAGE(STATUS, 'Path Invalid Bandwidth');

end if;

else

IF DS1\_NUMBER IS NULL THEN

IF UPPER(BANDWIDTH) LIKE '%DS1%' THEN

STATUS := SET\_MESSAGE(STATUS, 'Path Invalid Bandwidth');

end if;

ELSE

STATUS := SET\_MESSAGE(STATUS, 'Path Invalid Format');

IF UPPER(BANDWIDTH) LIKE '%DS1%' THEN

STATUS := SET\_MESSAGE(STATUS, 'Path Invalid Bandwidth');

END IF;

END IF;

END IF;

IF ISUBS = UBS\_TYPE then

IF BTS\_TYPE IS NOT NULL AND BTS\_TYPE NOT LIKE 'UB%' THEN

STATUS := SET\_MESSAGE(STATUS, 'Invalid UBS Pathname');

END IF;

ELSE

IF BTS\_TYPE LIKE 'UB%' THEN

STATUS := SET\_MESSAGE(STATUS, 'Invalid UBS Pathname');

END IF;

END IF;

RETURN STATUS;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SUBSTR('Error in GET\_EVDO\_T1\_MATCH\_STATUS(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

END GET\_UBS\_MATCH\_STATUS;

PROCEDURE moto\_cdma\_audit

IS

-- One Live and N non Live match

CURSOR t1\_nl\_match IS

SELECT NE\_OMCR,NE\_BTS\_NUMBER,NE\_CLUSTER\_NUMBER,NE\_SPAN\_NUMBER

FROM moto\_cdma\_ne\_vs\_xng\_audit\_WRK

WHERE match\_code='BOTH' AND xng\_path\_status='Live'

AND termination\_type = 'T1';

-- One Live and N non Live match

CURSOR ebh\_nl\_match IS

SELECT NE\_OMCR,NE\_BTS\_NUMBER,NE\_CLUSTER\_NUMBER

FROM moto\_cdma\_ne\_vs\_xng\_audit\_WRK

WHERE match\_code='BOTH' AND xng\_path\_status='Live'

AND termination\_type = 'Ethernet';

-- T1 spans matching multiple live paths in xng or matching multiple non live paths in xng in case a live match does not exist are marked NE Only

CURSOR t1\_cdma\_multiples\_xng IS

WITH

dup AS

(

SELECT DISTINCT ne\_omcr,ne\_bts\_number,ne\_cluster\_number,ne\_span\_number,

count(CASE WHEN xng\_path\_status = 'Live' THEN 1 ELSE NULL END) over (Partition by ne\_omcr, ne\_bts\_number, ne\_cluster\_number, ne\_span\_number) as live\_count,

COUNT(CASE WHEN XNG\_PATH\_STATUS != 'Live' THEN 1 ELSE NULL END) OVER (PARTITION BY NE\_OMCR, NE\_BTS\_NUMBER, NE\_CLUSTER\_NUMBER, NE\_SPAN\_NUMBER) AS NON\_LIVE\_COUNT

FROM moto\_cdma\_ne\_vs\_xng\_audit\_WRK src

WHERE src.match\_code='BOTH'

AND SRC.TERMINATION\_TYPE='T1'

AND SRC.NE\_BTS\_STATUS IN ('OOS','INS','INS\_ACTIVE')

)

SELECT TGT.ROWID

FROM moto\_cdma\_ne\_vs\_xng\_audit\_WRK tgt

JOIN dup

ON dup.ne\_omcr = tgt.ne\_omcr

AND dup.ne\_bts\_number = tgt.ne\_bts\_number

AND dup.ne\_cluster\_number = tgt.ne\_cluster\_number

AND dup.ne\_span\_number = tgt.ne\_span\_number

AND (dup.live\_count > 1 OR (dup.live\_count = 0 AND dup.non\_live\_count > 1)

);

-- Ethernet spans matching multiple live paths in xng or matching multiple non live paths in xng incase a live match does not exist are marked NE Only

CURSOR ethernet\_cdma\_multiples\_xng IS

WITH

dup AS

(

SELECT DISTINCT ne\_omcr, ne\_bts\_number, ne\_cluster\_number,

count(CASE WHEN xng\_path\_status = 'Live' THEN 1 ELSE null end) over (Partition by ne\_omcr, ne\_bts\_number, ne\_cluster\_number) as live\_count,

COUNT(CASE WHEN XNG\_PATH\_STATUS != 'Live' THEN 1 ELSE NULL END) OVER (PARTITION BY NE\_OMCR, NE\_BTS\_NUMBER, NE\_CLUSTER\_NUMBER) AS NON\_LIVE\_COUNT

FROM moto\_cdma\_ne\_vs\_xng\_audit\_WRK src

WHERE src.match\_code='BOTH'

AND SRC.TERMINATION\_TYPE='Ethernet'

and src.ne\_bts\_status IN ('OOS','INS','INS\_ACTIVE')

)

SELECT TGT.ROWID

FROM moto\_cdma\_ne\_vs\_xng\_audit\_WRK tgt

JOIN dup

ON dup.ne\_omcr = tgt.ne\_omcr

AND dup.ne\_bts\_number = tgt.ne\_bts\_number

AND dup.ne\_cluster\_number = tgt.ne\_cluster\_number

AND (dup.live\_count > 1 OR (dup.live\_count = 0 AND dup.non\_live\_count > 1)

);

--JC

-- Update those paths that dont have a CSR port

cursor update\_mcdma\_wrk\_no\_csr\_port is

WITH

MXA\_PATHS as

(

select mxa.XNG\_PATH\_INST\_ID

from moto\_cdma\_ne\_vs\_xng\_audit\_wrk mxa

where mxa.TERMINATION\_TYPE = 'Ethernet'

AND mxa.XNG\_BANDWIDTH LIKE '%bps'

and mxa.match\_code = 'BOTH'

and mxa.XNG\_PATH\_STATUS = 'Live'

)

,

PATHS\_W\_CSR as

(

select cpi.xng\_path\_inst\_id

from MXA\_PATHS cpi

join VZWNET.circ\_path\_element cpe

on cpi.xng\_path\_inst\_id = cpe.CIRC\_PATH\_INST\_ID

join vzwnet.epa p

on CPE.PORT\_INST\_ID = P.PORT\_INST\_ID

join VZWNET.equip\_inst ei

on p.equip\_INST\_ID = ei.equip\_INST\_ID

and ei.type = 'CSR'

)

select MXAP.xng\_path\_inst\_id

from MXA\_PATHS mxap

MINUS

select pc.XNG\_PATH\_INST\_ID

FROM PATHS\_W\_CSR PC;

-- mis-matched BTS Type

CURSOR BTS\_TYPE\_MATCH IS

SELECT rowid rid

--NE\_OMCR,NE\_BTS\_NUMBER,NE\_CLUSTER\_NUMBER,ne\_span\_number

FROM moto\_cdma\_ne\_vs\_xng\_audit\_WRK

WHERE MATCH\_CODE='BOTH'

and SUBSTR(NE\_BTS\_TYPE, 1,2) = 'UB';

-- find the duplicated entries and remove the NE Only entry

cursor ne\_ethernet\_multiples is

select ne\_omcr, ne\_bts\_number, ne\_cluster\_number

FROM MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK A

WHERE A.MATCH\_CODE = 'BOTH'

AND A.TERMINATION\_TYPE LIKE 'Eth%'

;

cursor ne\_t1\_multiples is

select ne\_omcr, ne\_bts\_number, ne\_cluster\_number, ne\_span\_number

FROM MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK A

where a.MATCH\_CODE = 'BOTH'

and a.TERMINATION\_TYPE = 'T1'

;

sql\_stmt varchar2(32000);

BEGIN

sql\_stmt:='truncate TABLE moto\_cdma\_ne\_vs\_xng\_audit\_wrk';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in moto\_cdma\_audit();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

end;

commit;

Insert into MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK

(ne\_omcr,ne\_bts\_number,ne\_cluster\_number,ne\_bts\_router\_group,ne\_bandwidth,ne\_span\_number,

NE\_SPAN\_TYPE,ne\_access\_node\_number,ne\_ag\_node\_number,ne\_ag\_span\_number,ne\_t1\_line\_number,

NE\_BTS\_TYPE, ne\_bts\_status,termination\_type, switch\_id,mtx, xng\_path\_name,

xng\_path\_inst\_id,XNG\_IPBSCDO\_NUMBER,xng\_bts\_number,xng\_cluster\_number,xng\_span\_number,

xng\_bandwidth,xng\_path\_status, xng\_path\_type,match\_code,match\_status,extract\_date,audit\_date

)

WITH BTS\_STATUS as

(

select distinct \*

from mt\_omcr\_bts\_status stat

where stat.BTS\_MLPPP\_NUMBER is null

AND STAT.PBTS\_NUMBER IS NULL

),

moto\_ether\_terminations AS

(SELECT inv.vsm\_device\_name\_omcr,inv.bts\_number,inv.cluster\_number,

inv.router\_grp,inv.bandwidth,inv.extract\_date,

SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2) switch\_id,

'Ethernet' termination\_type, bmm.vsm\_device\_name\_mtx,

MP.BTS\_TYPE, stat.tel\_state ne\_bts\_status

FROM moto\_cdma\_ethernet\_inv inv

JOIN bsm\_mtx\_map bmm

ON inv.vsm\_device\_name\_omcr = bmm.vsm\_device\_name\_bsm

and upper(bmm.mtx\_status )='LIVE'

and upper(bmm.bsm\_status)='LIVE'

JOIN MOTO\_BTS\_LOCATION\_MAP MP

on inv.vsm\_device\_name\_omcr = MP.VSM\_DEVICE\_NAME\_OMCR

AND INV.BTS\_NUMBER = MP.BTS\_NUMBER

join BTS\_STATUS stat

ON inv.vsm\_device\_name\_omcr = stat.VSM\_DEVICE\_NAME\_OMCR

and inv.bts\_number = stat.bts\_number

),

moto\_t1\_terminations AS

(SELECT mcs.bts\_number, mcs.cluster\_number,

mcs.bts\_router\_group, mcs.span\_number, mcs.access\_node\_number,

mcs.ag\_node\_number, mcs.ag\_span\_number, mcs.t1\_line\_number,

mcs.extract\_date, mcs.vsm\_device\_name\_omcr,

SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2) switch\_id,

'T1' termination\_type, bmm.vsm\_device\_name\_mtx,

MP.BTS\_TYPE, stat.tel\_state ne\_bts\_status

FROM moto\_cdma\_t1\_inv mcs

JOIN bsm\_mtx\_map bmm

ON mcs.vsm\_device\_name\_omcr = bmm.vsm\_device\_name\_bsm

and upper(bmm.mtx\_status )='LIVE'

and upper(bmm.bsm\_status)='LIVE'

JOIN MOTO\_BTS\_LOCATION\_MAP MP

on mcs.vsm\_device\_name\_omcr = MP.VSM\_DEVICE\_NAME\_OMCR

AND MCS.BTS\_NUMBER = MP.BTS\_NUMBER

join BTS\_STATUS stat

ON mcs.vsm\_device\_name\_omcr = stat.VSM\_DEVICE\_NAME\_OMCR

and mcs.bts\_number = stat.bts\_number

),

moto\_ether\_audit AS

(SELECT met.vsm\_device\_name\_omcr ne\_omcr, met.bts\_number ne\_bts\_number,

met.cluster\_number ne\_cluster\_number,

met.router\_grp ne\_bts\_router\_group, met.bandwidth ne\_bandwidth,

NULL ne\_span\_number, 'VOICE' NE\_SPAN\_TYPE, NULL ne\_access\_node\_number,

NULL ne\_ag\_node\_number, NULL ne\_ag\_span\_number,

NULL ne\_t1\_line\_number, met.BTS\_TYPE ne\_BTS\_TYPE, met.ne\_bts\_status,

met.termination\_type, met.switch\_id,

met.vsm\_device\_name\_mtx mtx, cpi.path\_name xng\_path\_name,

CPI.CIRC\_PATH\_INST\_ID XNG\_PATH\_INST\_ID,

null XNG\_IPBSCDO\_NUMBER, cpi.bts\_number xng\_bts\_number,

cpi.cluster\_number xng\_cluster\_number, cpi.span\_number xng\_span\_number,cpi.bandwidth xng\_bandwidth,

xng\_path\_status, xng\_path\_type,

CASE

WHEN cpi.switch\_id IS NULL

THEN 'NE Only'

ELSE 'BOTH'

END match\_code,

CASE

WHEN CPI.SWITCH\_ID IS NULL

THEN 'Not found in Granite'

ELSE

GET\_UBS\_MATCH\_STATUS (CPI.SPAN\_NUMBER, CPI.BANDWIDTH, met.BTS\_TYPE, EBH\_TYPE, NON\_UBS\_TYPE, CPI.PARSE\_STATUS)

END match\_status, met.extract\_date,

cpi.ROWID cpi\_rowid

FROM moto\_ether\_terminations met

LEFT OUTER JOIN (SELECT \* FROM vzw\_moto\_voice\_cell\_paths cp

WHERE cp.bts\_number IS NOT NULL AND cp.cluster\_number IS NOT NULL)cpi

ON met.switch\_id = cpi.switch\_id

AND met.bts\_number = cpi.bts\_number

AND met.cluster\_number = cpi.cluster\_number

),

moto\_t1\_audit AS

(SELECT mtt.vsm\_device\_name\_omcr ne\_omcr, mtt.bts\_number ne\_bts\_number,

mtt.cluster\_number ne\_cluster\_number,

mtt.bts\_router\_group ne\_bts\_router\_group, NULL ne\_bandwidth,

mtt.span\_number ne\_span\_number, 'VOICE' NE\_SPAN\_TYPE,

mtt.access\_node\_number ne\_access\_node\_number,

mtt.ag\_node\_number ne\_ag\_node\_number,

mtt.ag\_span\_number ne\_ag\_span\_number,

mtt.t1\_line\_number ne\_t1\_line\_number, mtt.BTS\_TYPE ne\_BTS\_TYPE, mtt.ne\_bts\_status,

mtt.termination\_type, mtt.switch\_id, mtt.vsm\_device\_name\_mtx mtx,

cpi.path\_name xng\_path\_name, cpi.circ\_path\_inst\_id xng\_path\_inst\_id,

null XNG\_IPBSCDO\_NUMBER, cpi.bts\_number xng\_bts\_number,

cpi.cluster\_number xng\_cluster\_number,

cpi.span\_number xng\_span\_number,cpi.bandwidth xng\_bandwidth, xng\_path\_status, xng\_path\_type,

CASE

WHEN cpi.switch\_id IS NULL

THEN 'NE Only'

ELSE 'BOTH'

END match\_code,

CASE

WHEN CPI.SWITCH\_ID IS NULL

THEN 'Not found in Granite'

ELSE

GET\_UBS\_MATCH\_STATUS (CPI.SPAN\_NUMBER, CPI.BANDWIDTH, mtt.BTS\_TYPE, T1\_TYPE, NON\_UBS\_TYPE, CPI.PARSE\_STATUS)

END match\_status, mtt.extract\_date,

cpi.ROWID cpi\_rowid

FROM moto\_t1\_terminations mtt

LEFT OUTER JOIN (SELECT \* FROM vzw\_moto\_voice\_cell\_paths cp

WHERE cp.bts\_number IS NOT NULL AND cp.cluster\_number IS NOT NULL and cp.span\_number IS NOT NULL) cpi

ON mtt.switch\_id = cpi.switch\_id

AND mtt.bts\_number = cpi.bts\_number

AND mtt.cluster\_number = cpi.cluster\_number

AND mtt.span\_number = cpi.span\_number

),

moto\_ne\_only\_and\_xng AS

(SELECT \* FROM moto\_ether\_audit ea

UNION

SELECT \* FROM moto\_t1\_audit t1a

),

xng\_only\_path\_id AS

(SELECT ROWID cpi\_rowid FROM vzw\_moto\_voice\_cell\_paths cpi

MINUS

SELECT cpi\_rowid FROM moto\_ne\_only\_and\_xng),

moto\_xng\_only AS

(SELECT NULL ne\_omcr, NULL ne\_bts\_number, NULL ne\_cluster\_number,

NULL ne\_bts\_router\_group, NULL ne\_bandwidth,

NULL ne\_span\_number, NULL NE\_SPAN\_TYPE,

NULL ne\_access\_node\_number, NULL ne\_ag\_node\_number,

NULL ne\_ag\_span\_number, NULL ne\_t1\_line\_number,

NULL ne\_bts\_type, null ne\_bts\_status,

CASE

WHEN cpi.bandwidth = 'DS1'

THEN 'T1'

ELSE 'Ethernet'

END termination\_type,

cpi.switch\_id, bmm.vsm\_device\_name\_mtx mtx,

cpi.path\_name xng\_path\_name, cpi.circ\_path\_inst\_id xng\_path\_inst\_id,

null XNG\_IPBSCDO\_NUMBER, cpi.bts\_number xng\_bts\_number,

cpi.cluster\_number xng\_cluster\_number,

cpi.span\_number xng\_span\_number,cpi.bandwidth xng\_bandwidth, xng\_path\_status, xng\_path\_type,

'Xng Only' match\_code, parse\_status match\_status,

NULL extract\_date, cpi.ROWID cpi\_rowid

FROM vzw\_moto\_voice\_cell\_paths cpi,

xng\_reports.bsm\_mtx\_map bmm,

xng\_only\_path\_id xopi

WHERE cpi.switch\_id =

( SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2)

)

AND upper(bmm.mtx\_status )='LIVE'

AND cpi.ROWID = xopi.cpi\_rowid),

full\_audit AS

(SELECT \* FROM moto\_ne\_only\_and\_xng

UNION

SELECT \* FROM moto\_xng\_only

)

SELECT ne\_omcr, ne\_bts\_number, ne\_cluster\_number,

ne\_bts\_router\_group, ne\_bandwidth, ne\_span\_number,

NE\_SPAN\_TYPE, ne\_access\_node\_number, ne\_ag\_node\_number,

ne\_ag\_span\_number, ne\_t1\_line\_number,

NE\_BTS\_TYPE, ne\_bts\_status,

termination\_type, switch\_id, mtx,

XNG\_PATH\_NAME, XNG\_PATH\_INST\_ID, XNG\_IPBSCDO\_NUMBER,

xng\_bts\_number, xng\_cluster\_number, xng\_span\_number,

xng\_bandwidth, xng\_path\_status, xng\_path\_type,

MATCH\_CODE, MATCH\_STATUS, extract\_date,

TRUNC (SYSDATE) AS audit\_date

FROM full\_audit fa

;

COMMIT;

dbms\_output.put\_line('Motorola CDMA audit table created');

-- perform UBS Type audit

CREATE\_UBS\_AUDIT();

-- remove the NE Only entry if BOTH exists in UBS path

-- due to multiple comparison for UBS

FOR CURSORREC IN NE\_ETHERNET\_MULTIPLES LOOP

DELETE FROM MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK U

WHERE U.MATCH\_CODE LIKE 'NE%'

AND u.TERMINATION\_TYPE LIKE 'Eth%'

and u.ne\_omcr = cursorRec.ne\_omcr

and u.ne\_bts\_number = cursorRec.ne\_bts\_number

and u.ne\_cluster\_number = cursorRec.ne\_cluster\_number

;

END LOOP;

COMMIT;

FOR cursorRec IN ne\_t1\_multiples LOOP

DELETE FROM MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK U

WHERE U.MATCH\_CODE LIKE 'NE%'

AND u.TERMINATION\_TYPE = 'T1'

and u.ne\_omcr = cursorRec.ne\_omcr

AND U.NE\_BTS\_NUMBER = CURSORREC.NE\_BTS\_NUMBER

and u.ne\_span\_number = cursorRec.ne\_span\_number

AND U.NE\_CLUSTER\_NUMBER = CURSORREC.NE\_CLUSTER\_NUMBER

;

END LOOP;

COMMIT;

-- check duplication

FOR CURSORREC IN T1\_CDMA\_MULTIPLES\_XNG LOOP

UPDATE MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK

SET MATCH\_STATUS = NVL2(MATCH\_STATUS,MATCH\_STATUS||',Multiple spans found in Granite for same tokens','Multiple spans found in Granite for same tokens') ,

match\_code='NE Only'

WHERE rowid = cursorRec.ROWID;

END LOOP;

commit;

FOR CURSORREC IN ETHERNET\_CDMA\_MULTIPLES\_XNG LOOP

UPDATE moto\_cdma\_ne\_vs\_xng\_audit\_wrk

SET MATCH\_STATUS = NVL2(MATCH\_STATUS,MATCH\_STATUS||',Multiple spans found in Granite for same tokens','Multiple spans found in Granite for same tokens') ,

match\_code='NE Only'

WHERE rowid = cursorRec.ROWID;

END LOOP;

commit;

UPDATE moto\_cdma\_ne\_vs\_xng\_audit\_wrk

SET match\_code = 'NE Only'

WHERE match\_code = 'BOTH' AND match\_status IS NOT NULL;

COMMIT;

FOR cursorRec IN t1\_nl\_match LOOP

UPDATE MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK MNVXA

SET match\_status = nvl2(match\_status,'Live Path Exists, '||match\_status,'Live Path Exists')

WHERE mnvxa.ne\_omcr=cursorRec.ne\_omcr

AND mnvxa.ne\_bts\_number=cursorRec.ne\_bts\_number

AND mnvxa.ne\_cluster\_number=cursorRec.ne\_cluster\_number

AND mnvxa.ne\_span\_number=cursorRec.ne\_span\_number

AND mnvxa.termination\_type='T1'

AND mnvxa.xng\_path\_status <> 'Live';

END LOOP;

COMMIT;

FOR cursorRec IN ebh\_nl\_match LOOP

UPDATE MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK MNVXA

SET match\_status = nvl2(match\_status,'Live Path Exists, '||match\_status,'Live Path Exists')

WHERE mnvxa.ne\_omcr=cursorRec.ne\_omcr

AND mnvxa.ne\_bts\_number=cursorRec.ne\_bts\_number

AND mnvxa.ne\_cluster\_number=cursorRec.ne\_cluster\_number

AND mnvxa.termination\_type='Ethernet'

AND mnvxa.xng\_path\_status <> 'Live';

END LOOP;

COMMIT;

--JC

FOR cursorRec IN update\_mcdma\_wrk\_no\_csr\_port LOOP

UPDATE moto\_cdma\_ne\_vs\_xng\_audit\_wrk mxa

SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS =

CASE

when MATCH\_STATUS = 'BOTH'

then 'No CSR port on the e-pipe'

else

nvl2(match\_status, 'No CSR port on the e-pipe, '||match\_status, 'No CSR port on the e-pipe')

END

WHERE

MXA.XNG\_PATH\_INST\_ID = cursorRec.XNG\_PATH\_INST\_ID

;

END LOOP;

commit;

--JC

dbms\_output.put\_line('Motorola audit completd successfully');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error processing moto\_cdma\_audit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

PROCEDURE create\_ubs\_audit IS

BEGIN

insert into MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_wrk

(ne\_omcr,ne\_bts\_number,ne\_cluster\_number,ne\_bts\_router\_group,ne\_bandwidth,ne\_span\_number,

NE\_SPAN\_TYPE, ne\_access\_node\_number,ne\_ag\_node\_number,ne\_ag\_span\_number,ne\_t1\_line\_number,

NE\_BTS\_TYPE, ne\_bts\_status,termination\_type, switch\_id,mtx, xng\_path\_name,

xng\_path\_inst\_id,XNG\_IPBSCDO\_NUMBER,xng\_bts\_number,xng\_cluster\_number,xng\_span\_number,

xng\_bandwidth,xng\_path\_status, xng\_path\_type,match\_code,match\_status,extract\_date,audit\_date

)

WITH BTS\_STATUS as

(

select distinct \*

from mt\_omcr\_bts\_status stat

where stat.BTS\_MLPPP\_NUMBER is null

and stat.PBTS\_NUMBER is null

),

moto\_ether\_terminations AS

(SELECT inv.vsm\_device\_name\_omcr,inv.bts\_number,inv.cluster\_number,

inv.router\_grp,inv.bandwidth,inv.extract\_date,

SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2) switch\_id,

'Ethernet' termination\_type, bmm.vsm\_device\_name\_mtx,

MP.BTS\_TYPE, stat.tel\_state ne\_bts\_status

FROM moto\_cdma\_ethernet\_inv inv

JOIN bsm\_mtx\_map bmm

ON inv.vsm\_device\_name\_omcr = bmm.vsm\_device\_name\_bsm

and upper(bmm.mtx\_status )='LIVE'

AND UPPER(BMM.BSM\_STATUS)='LIVE'

JOIN MOTO\_BTS\_LOCATION\_MAP MP

on inv.vsm\_device\_name\_omcr = MP.VSM\_DEVICE\_NAME\_OMCR

AND INV.BTS\_NUMBER = MP.BTS\_NUMBER

join BTS\_STATUS stat

ON inv.vsm\_device\_name\_omcr = stat.VSM\_DEVICE\_NAME\_OMCR

and inv.bts\_number = stat.bts\_number

),

moto\_t1\_terminations AS

(SELECT mcs.bts\_number, mcs.cluster\_number,

mcs.bts\_router\_group, mcs.span\_number, mcs.access\_node\_number,

mcs.ag\_node\_number, mcs.ag\_span\_number, mcs.t1\_line\_number,

mcs.extract\_date, mcs.vsm\_device\_name\_omcr,

SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2) switch\_id,

'T1' termination\_type, bmm.vsm\_device\_name\_mtx,

MP.BTS\_TYPE, stat.tel\_state ne\_bts\_status

FROM moto\_cdma\_t1\_inv mcs

JOIN bsm\_mtx\_map bmm

ON mcs.vsm\_device\_name\_omcr = bmm.vsm\_device\_name\_bsm

and upper(bmm.mtx\_status )='LIVE'

AND UPPER(BMM.BSM\_STATUS)='LIVE'

JOIN MOTO\_BTS\_LOCATION\_MAP MP

on mcs.vsm\_device\_name\_omcr = MP.VSM\_DEVICE\_NAME\_OMCR

AND MCS.BTS\_NUMBER = MP.BTS\_NUMBER

join BTS\_STATUS stat

ON mcs.vsm\_device\_name\_omcr = stat.VSM\_DEVICE\_NAME\_OMCR

and mcs.bts\_number = stat.bts\_number

),

evdo\_t1 AS

(SELECT SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2) switch\_id,

MTT.vsm\_device\_name\_omcr,

MTT.BTS\_NUMBER NE\_BTS\_NUMBER,

NVL(MTT.BTS\_CON\_NUMBER, MTT.PBTS\_SPAN\_NUMBER) ne\_SPAN\_NUMBER,

mtt.access\_node\_number ne\_access\_node\_number,

mtt.ag\_node\_number ne\_ag\_node\_number,

mtt.ag\_span\_number ne\_ag\_span\_number,

MTT.T1\_LINE\_NUMBER NE\_T1\_LINE\_NUMBER,

bmm.vsm\_device\_name\_mtx, mtt.extract\_date,

MP.BTS\_TYPE, stat.tel\_state bts\_status

FROM MT\_OMCR\_BTS\_CONCONF MTT

JOIN BTS\_STATUS stat

ON mtt.vsm\_device\_name\_omcr = stat.VSM\_DEVICE\_NAME\_OMCR

and mtt.bts\_number = stat.bts\_number

JOIN bsm\_mtx\_map bmm

ON mtt.vsm\_device\_name\_omcr = bmm.vsm\_device\_name\_bsm

and upper(bmm.mtx\_status )='LIVE'

AND UPPER(BMM.BSM\_STATUS)='LIVE'

JOIN MOTO\_BTS\_LOCATION\_MAP MP

on mtt.VSM\_DEVICE\_NAME\_OMCR = MP.VSM\_DEVICE\_NAME\_OMCR

AND MTT.BTS\_NUMBER = MP.BTS\_NUMBER

),

evdo\_t1\_audit AS

(SELECT MTT.VSM\_DEVICE\_NAME\_OMCR NE\_OMCR, MTT.NE\_BTS\_NUMBER,

null ne\_cluster\_number, null ne\_bts\_router\_group, NULL ne\_bandwidth,

mtt.ne\_SPAN\_NUMBER, 'DATA' NE\_SPAN\_TYPE,

mtt.ne\_access\_node\_number,

mtt.ne\_ag\_node\_number,

mtt.ne\_ag\_span\_number,

MtT.NE\_T1\_LINE\_NUMBER,

mtt.BTS\_TYPE ne\_BTS\_TYPE, mtt.bts\_status ne\_bts\_status,

'T1' termination\_type, mtt.switch\_id, mtt.vsm\_device\_name\_mtx mtx,

cpi.path\_name xng\_path\_name, cpi.circ\_path\_inst\_id xng\_path\_inst\_id,

cpi.ipbscdo\_number xng\_ipbscdo\_number, cpi.bts\_number xng\_bts\_number, null xng\_cluster\_number,

cpi.span\_number xng\_span\_number,cpi.bandwidth xng\_bandwidth, xng\_path\_status, xng\_path\_type,

CASE

WHEN cpi.switch\_id IS NULL or (MTT.SWITCH\_ID <> CPI.SWITCH\_ID)

THEN 'NE Only'

ELSE 'BOTH'

END match\_code,

CASE

WHEN CPI.SWITCH\_ID IS NULL

THEN 'Not found in Granite'

ELSE

GET\_UBS\_MATCH\_STATUS (CPI.SPAN\_NUMBER, CPI.BANDWIDTH, MTT.BTS\_TYPE, T1\_TYPE, UBS\_TYPE, CPI.PARSE\_STATUS)

END match\_status, mtt.extract\_date,

cpi.ROWID cpi\_rowid

FROM evdo\_t1 mtt

LEFT OUTER JOIN VZW\_MOTO\_UBS\_CELL\_PATHS cpi

ON mtt.ne\_bts\_number = cpi.bts\_number

AND MTT.NE\_SPAN\_NUMBER = CPI.SPAN\_NUMBER

AND MTT.SWITCH\_ID = CPI.SWITCH\_ID

),

ubs\_t1\_audit AS

(SELECT mtt.vsm\_device\_name\_omcr ne\_omcr, mtt.bts\_number ne\_bts\_number,

MTT.CLUSTER\_NUMBER NE\_CLUSTER\_NUMBER,

mtt.bts\_router\_group ne\_bts\_router\_group, NULL ne\_bandwidth,

mtt.span\_number ne\_span\_number, 'VOICE' NE\_SPAN\_TYPE,

mtt.access\_node\_number ne\_access\_node\_number,

mtt.ag\_node\_number ne\_ag\_node\_number,

mtt.ag\_span\_number ne\_ag\_span\_number,

mtt.t1\_line\_number ne\_t1\_line\_number,

mtt.BTS\_TYPE ne\_BTS\_TYPE, ne\_bts\_status, mtt.termination\_type,

mtt.switch\_id, mtt.vsm\_device\_name\_mtx mtx,

cpi.path\_name xng\_path\_name, cpi.circ\_path\_inst\_id xng\_path\_inst\_id,

cpi.ipbscdo\_number xng\_ipbscdo\_number, cpi.bts\_number xng\_bts\_number, null xng\_cluster\_number,

cpi.span\_number xng\_span\_number,cpi.bandwidth xng\_bandwidth, xng\_path\_status, xng\_path\_type,

CASE

WHEN cpi.switch\_id IS NULL

THEN 'NE Only'

ELSE 'BOTH'

END match\_code,

CASE

WHEN CPI.SWITCH\_ID IS NULL

THEN 'Not found in Granite'

ELSE

GET\_UBS\_MATCH\_STATUS (CPI.SPAN\_NUMBER, CPI.BANDWIDTH, MTT.BTS\_TYPE, T1\_TYPE, UBS\_TYPE, CPI.PARSE\_STATUS)

END MATCH\_STATUS,

mtt.extract\_date, cpi.ROWID cpi\_rowid

FROM moto\_t1\_terminations mtt

JOIN VZW\_MOTO\_UBS\_CELL\_PATHS cpi

ON mtt.switch\_id = cpi.switch\_id

AND mtt.bts\_number = cpi.bts\_number

AND mtt.span\_number = cpi.span\_number

),

ubs\_ether\_audit AS

(SELECT met.vsm\_device\_name\_omcr ne\_omcr, met.bts\_number ne\_bts\_number,

met.cluster\_number ne\_cluster\_number,

MET.ROUTER\_GRP NE\_BTS\_ROUTER\_GROUP, MET.BANDWIDTH NE\_BANDWIDTH,

NULL ne\_span\_number, null NE\_SPAN\_TYPE, NULL ne\_access\_node\_number,

NULL ne\_ag\_node\_number, NULL ne\_ag\_span\_number,

NULL ne\_t1\_line\_number, met.BTS\_TYPE ne\_BTS\_TYPE, met.ne\_bts\_status,

met.termination\_type, met.switch\_id,

MET.VSM\_DEVICE\_NAME\_MTX MTX, CPI.PATH\_NAME XNG\_PATH\_NAME,

CPI.CIRC\_PATH\_INST\_ID XNG\_PATH\_INST\_ID, CPI.IPBSCDO\_NUMBER XNG\_IPBSCDO\_NUMBER,

cpi.bts\_number xng\_bts\_number, null xng\_cluster\_number,

cpi.span\_number xng\_span\_number,cpi.bandwidth xng\_bandwidth,

xng\_path\_status, xng\_path\_type,

CASE

WHEN cpi.switch\_id IS NULL

THEN 'NE Only'

WHEN MET.BTS\_TYPE IS NULL OR SUBSTR(MET.BTS\_TYPE, 1,2) <> 'UB'

then 'NE Only'

ELSE 'BOTH'

END match\_code,

CASE

WHEN CPI.SWITCH\_ID IS NULL

THEN 'Not found in Granite'

ELSE

GET\_UBS\_MATCH\_STATUS (CPI.SPAN\_NUMBER, CPI.BANDWIDTH, MET.BTS\_TYPE, EBH\_TYPE, UBS\_TYPE, CPI.PARSE\_STATUS)

END MATCH\_STATUS,

met.extract\_date,

cpi.ROWID cpi\_rowid

FROM MOTO\_ETHER\_TERMINATIONS MET

JOIN VZW\_MOTO\_UBS\_CELL\_PATHS cpi

ON met.switch\_id = cpi.switch\_id

AND met.bts\_number = cpi.bts\_number

),

ubs\_ne\_only\_and\_both as

(

SELECT \* FROM ubs\_ether\_audit

UNION

select \* from ubs\_t1\_audit

UNION

select \* from evdo\_t1\_audit

),

ubs\_xng\_only\_path\_id AS

(SELECT ROWID cpi\_rowid FROM VZW\_MOTO\_UBS\_CELL\_PATHS cpi

MINUS

SELECT cpi\_rowid FROM ubs\_ne\_only\_and\_both),

ubs\_xng\_only AS

(SELECT NULL ne\_omcr, NULL ne\_bts\_number, NULL ne\_cluster\_number,

null ne\_bts\_router\_group, NULL ne\_bandwidth, NULL ne\_span\_number,

null NE\_SPAN\_TYPE, NULL ne\_access\_node\_number, NULL ne\_ag\_node\_number,

NULL ne\_ag\_span\_number, NULL ne\_t1\_line\_number,

NULL ne\_bts\_type, null ne\_bts\_status,

CASE

WHEN cpi.bandwidth = 'DS1'

THEN 'T1'

ELSE 'Ethernet'

END termination\_type,

cpi.switch\_id, bmm.vsm\_device\_name\_mtx mtx,

cpi.path\_name xng\_path\_name, cpi.circ\_path\_inst\_id xng\_path\_inst\_id,

CPI.IPBSCDO\_NUMBER XNG\_IPBSCDO\_NUMBER, CPI.BTS\_NUMBER XNG\_BTS\_NUMBER,

null xng\_cluster\_number, cpi.span\_number xng\_span\_number,

cpi.bandwidth xng\_bandwidth, xng\_path\_status, xng\_path\_type,

'Xng Only' match\_code, parse\_status match\_status,

NULL extract\_date, cpi.ROWID cpi\_rowid

FROM VZW\_MOTO\_UBS\_CELL\_PATHS cpi

JOIN UBS\_XNG\_ONLY\_PATH\_ID XOPI

on CPI.ROWID = XOPI.CPI\_ROWID

JOIN XNG\_REPORTS.BSM\_MTX\_MAP BMM

on CPI.SWITCH\_ID = SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6) || SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2)

AND UPPER(BMM.MTX\_STATUS )='LIVE'

WHERE CPI.XNG\_PATH\_TYPE <> 'EVDO'

OR (CPI.XNG\_PATH\_TYPE = 'EVDO' AND CPI.BANDWIDTH = 'DS1')

),

full\_audit AS

(

SELECT \* FROM ubs\_ne\_only\_and\_both unb

UNION

SELECT \* FROM UBS\_XNG\_ONLY UX

)

SELECT ne\_omcr, ne\_bts\_number, ne\_cluster\_number,

ne\_bts\_router\_group, ne\_bandwidth, ne\_span\_number,

NE\_SPAN\_TYPE, ne\_access\_node\_number, ne\_ag\_node\_number,

ne\_ag\_span\_number, ne\_t1\_line\_number,

NE\_BTS\_TYPE, ne\_bts\_status,

termination\_type, switch\_id, mtx,

XNG\_PATH\_NAME, XNG\_PATH\_INST\_ID, XNG\_IPBSCDO\_NUMBER,

xng\_bts\_number, xng\_cluster\_number, xng\_span\_number,

xng\_bandwidth, xng\_path\_status, xng\_path\_type,

MATCH\_CODE, MATCH\_STATUS, extract\_date,

TRUNC (SYSDATE) AS AUDIT\_DATE

FROM FULL\_AUDIT FA;

commit;

dbms\_output.put\_line('create\_ubs\_audit() completd successfully');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in create\_ubs\_audit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- Procedure to create regional summary

PROCEDURE moto\_cdma\_regional\_summary IS

CURSOR UPDATECOMPLIANCE IS

SELECT summ.rowid rd,summ.\* FROM MOTO\_CDMA\_AUDIT\_REG\_SUMM\_WRK summ;

sqlStmt VARCHAR2(32767);

sqlStmtT1Comp VARCHAR2(32767);

sqlStmtEbhComp VARCHAR2(32767);

sqlStmtOvlComp VARCHAR2(32767);

t1Comp NUMBER;

ebhComp NUMBER;

overallComp NUMBER;

t1Reported NUMBER;

ebhReported NUMBER;

BEGIN

sqlStmt := 'truncate table ' || CDMA\_REGION\_SUMM;

execute IMMEDIATE sqlStmt;

sqlStmt := 'INSERT INTO '|| CDMA\_REGION\_SUMM ||'

(area,region, T1\_DISCOVERED,T1\_MATCHED\_LIVE,T1\_MATCHED\_NONLIVE,

EBH\_DISCOVERED,EBH\_MATCHED\_LIVE,EBH\_MATCHED\_NONLIVE,ebh\_matched\_live\_csr,ebh\_matched\_nonlive\_csr,

t1\_reported, ebh\_reported, t1\_comp, ebh\_comp, overall\_comp, ubs\_count, non\_ubs\_count

)

WITH mtx\_region\_map AS

(SELECT DISTINCT cdmp.area, cdmp.region, m.vsm\_device\_name\_mtx mtx,

( SUBSTR (vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (vsm\_device\_name\_mtx, 11, 2)

) switch\_id

FROM xng\_reports.clli\_domain\_map\_v cdmp

LEFT OUTER JOIN

(SELECT \*

FROM xng\_reports.bsm\_mtx\_map bmm

WHERE bmm.vsm\_device\_name\_bsm LIKE ''%OMCR%''

AND bmm.mtx\_status = ''Live'') m

ON SUBSTR (m.vsm\_device\_name\_mtx, 1, 6) = SUBSTR (cdmp.clli, 1, 6)

WHERE cdmp.area not in (''NNO'', ''OSS''))

SELECT mrm.area, mrm.region,

SUM (NVL (cds.t1\_discovered, 0)) AS t1\_discovered,

SUM (NVL (cds.t1\_matched\_live, 0)) AS t1\_matched\_live,

SUM (NVL (cds.t1\_matched\_nonlive, 0)) AS t1\_matched\_nonlive,

SUM (NVL (cds.ebh\_discovered, 0)) AS ebh\_discovered,

SUM (NVL (cds.ebh\_matched\_live, 0)) AS ebh\_matched\_live,

SUM (NVL (cds.ebh\_matched\_nonlive, 0)) AS ebh\_matched\_nonlive,

SUM (NVL (cds.ebh\_matched\_live\_csr, 0)) AS ebh\_matched\_live\_csr,

SUM (NVL (cds.ebh\_matched\_nonlive\_csr, 0)) AS ebh\_matched\_nonlive\_csr,

SUM (NVL (cds.t1\_discovered, 0)) - SUM (NVL (cds.t1\_matched\_nonlive, 0)) t1\_reported,

SUM (NVL (cds.ebh\_discovered, 0)) - SUM (NVL (cds.ebh\_matched\_nonlive\_csr, 0)) ebh\_reported,

case when SUM (NVL (cds.t1\_discovered, 0)) - SUM (NVL (cds.t1\_matched\_nonlive, 0)) > 0

then SUM (NVL (cds.t1\_matched\_live, 0)) / (SUM (NVL (cds.t1\_discovered, 0)) - SUM (NVL (cds.t1\_matched\_nonlive, 0)))\*100

else 0

end t1\_comp,

case when SUM (NVL (cds.ebh\_discovered, 0)) - SUM (NVL (cds.ebh\_matched\_nonlive\_csr, 0)) > 0

then SUM (NVL (cds.ebh\_matched\_live\_csr, 0)) / (SUM (NVL (cds.ebh\_discovered, 0)) - SUM (NVL (cds.ebh\_matched\_nonlive\_csr, 0)))\*100

else 0

end ebh\_comp,

case when SUM(nvl(cds.t1\_discovered,0) + nvl(cds.ebh\_discovered,0))- SUM(NVL (cds.ebh\_matched\_nonlive\_csr, 0)+NVL (cds.t1\_matched\_nonlive, 0)) > 0

then ((sum (nvl(cds.t1\_matched\_live,0)+nvl(cds.ebh\_matched\_live\_csr,0)))/(SUM(nvl(cds.t1\_discovered,0) + nvl(cds.ebh\_discovered,0))- SUM(NVL (cds.ebh\_matched\_nonlive\_csr, 0)+NVL (cds.t1\_matched\_nonlive, 0))))\*100

else 0

end overall\_comp,

SUM (NVL (cds.ubs\_count, 0)) AS ubs\_count,

SUM (NVL (cds.non\_ubs\_count, 0)) AS non\_ubs\_count

FROM mtx\_region\_map mrm

LEFT OUTER JOIN '|| CDMA\_DEVICE\_SUMM ||' cds

ON mrm.mtx = cds.VSM\_DEVICE\_NAME\_MTX

GROUP BY ROLLUP (mrm.area, mrm.region)

ORDER BY mrm.area, mrm.region';

execute IMMEDIATE sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlStmt);

dbms\_output.put\_line(SubStr('Error in populate\_cdma\_regional\_summary();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

-- Devicd Summary

PROCEDURE moto\_cdma\_device\_summary IS

CURSOR UPDATECOMPLIANCE IS

select summ.rowid rd,summ.\* from moto\_cdma\_audit\_dev\_summ\_wrk summ;

sqlStmt VARCHAR2(32767);

t1Comp NUMBER;

ebhComp NUMBER;

overallComp NUMBER;

BEGIN

sqlStmt := 'truncate table ' || CDMA\_DEVICE\_SUMM;

execute immediate sqlStmt;

insert into MOTO\_CDMA\_AUDIT\_DEV\_SUMM\_WRK

(AREA,REGION,LEAF\_DOMAIN,VSM\_DEVICE\_NAME\_MTX,SWITCH\_ID,T1\_DISCOVERED,T1\_MATCHED\_LIVE,T1\_MATCHED\_NONLIVE,

ebh\_discovered,ebh\_matched\_live,ebh\_matched\_nonlive, ebh\_matched\_live\_csr,ebh\_matched\_nonlive\_csr,

ubs\_count, non\_ubs\_count

)

WITH mtx\_region\_map AS

(SELECT DISTINCT cdmp.area, cdmp.region, cdmp.leaf\_domain\_name,

m.vsm\_device\_name\_mtx mtx,

( SUBSTR (vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (vsm\_device\_name\_mtx, 11, 2)

) switch\_id

FROM clli\_domain\_map\_v cdmp

RIGHT OUTER JOIN

(SELECT \*

FROM xng\_reports.bsm\_mtx\_map bmm

WHERE bmm.vsm\_device\_name\_bsm LIKE '%OMCR'

AND bmm.mtx\_status = 'Live') m

ON SUBSTR (m.vsm\_device\_name\_mtx, 1, 6) = SUBSTR (cdmp.clli, 1, 6)

WHERE cdmp.area not in ('NNO', 'OSS')),

AUDIT\_DETAILS AS

( SELECT \* FROM XNG\_REPORTS.MOTO\_CDMA\_NE\_VS\_XNG\_AUDIT\_WRK

where NE\_BTS\_STATUS IN ('OOS','INS', 'INS\_ACTIVE')

and match\_code IN ('BOTH', 'NE Only')

),

T1\_DETAILS AS

( select \* FROM audit\_details mnvxa

where mnvxa.termination\_type = 'T1'

),

t1\_discovered AS

(SELECT MTX, COUNT (1) T1\_DISCOVERED

FROM (SELECT DISTINCT MTX, NE\_BTS\_NUMBER,

NE\_CLUSTER\_NUMBER, NE\_SPAN\_NUMBER

from T1\_DETAILS )

GROUP BY mtx),

EBH\_DETAILS AS

( select \* FROM audit\_details mnvxa

where mnvxa.termination\_type = 'Ethernet'

),

ethernet\_discovered AS

(SELECT MTX, COUNT (1) ETHERNET\_DISCOVERED

FROM (SELECT DISTINCT MTX, NE\_BTS\_NUMBER, ne\_cluster\_number

from ebh\_details)

GROUP BY mtx),

t1\_matched\_live AS

(SELECT mtx, COUNT (1) t1\_matched\_live

FROM (SELECT DISTINCT MTX, NE\_BTS\_NUMBER, NE\_CLUSTER\_NUMBER, NE\_SPAN\_NUMBER

from T1\_DETAILS

WHERE XNG\_PATH\_STATUS = 'Live'

AND match\_code = 'BOTH')

GROUP BY mtx),

t1\_matched\_nonlive AS

(SELECT mtx, COUNT (1) t1\_matched\_nonlive

FROM (SELECT DISTINCT MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER,

MNVXA.NE\_CLUSTER\_NUMBER, MNVXA.NE\_SPAN\_NUMBER

FROM t1\_details mnvxa

WHERE mnvxa.xng\_path\_status <> 'Live'

AND mnvxa.match\_code = 'BOTH'

MINUS

SELECT DISTINCT MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER,

MNVXA.NE\_CLUSTER\_NUMBER, MNVXA.NE\_SPAN\_NUMBER

FROM t1\_details mnvxa

WHERE mnvxa.xng\_path\_status = 'Live'

AND mnvxa.match\_code = 'BOTH'

)

GROUP BY mtx),

-- ethernet\_matched\_live AS

-- (SELECT mtx, COUNT (1) ethernet\_matched\_live

-- FROM (SELECT DISTINCT MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER, MNVXA.NE\_CLUSTER\_NUMBER

-- from ebh\_details mnvxa

-- WHERE MNVXA.XNG\_PATH\_STATUS = 'Live'

-- AND mnvxa.match\_code = 'BOTH')

-- GROUP BY mtx),

ethernet\_matched\_live\_csr AS

(SELECT mtx, COUNT (1) ethernet\_matched\_live\_csr

FROM (SELECT DISTINCT MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER, MNVXA.NE\_CLUSTER\_NUMBER

FROM ebh\_details mnvxa

join VZWNET.CIRC\_PATH\_INST cpi on mnvxa.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mnvxa.xng\_path\_status = 'Live'

AND mnvxa.match\_code = 'BOTH'

and mnvxa.XNG\_BANDWIDTH like '%bps'

and EI.TYPE = 'CSR')

GROUP BY mtx),

-- ethernet\_matched\_nonlive AS

-- (SELECT MTX, COUNT (1) ETHERNET\_MATCHED\_NONLIVE

-- FROM (SELECT DISTINCT MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER, MNVXA.NE\_CLUSTER\_NUMBER

-- FROM ebh\_details mnvxa

-- WHERE mnvxa.xng\_path\_status <> 'Live'

-- AND mnvxa.match\_code = 'BOTH'

-- MINUS

-- SELECT DISTINCT MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER, MNVXA.NE\_CLUSTER\_NUMBER

-- FROM ebh\_details mnvxa

-- WHERE mnvxa.xng\_path\_status = 'Live'

-- AND mnvxa.match\_code = 'BOTH')

-- GROUP BY mtx),

ethernet\_matched\_nonlive\_csr AS

(SELECT mtx, COUNT (1) ethernet\_matched\_nonlive\_csr

FROM (SELECT DISTINCT MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER, MNVXA.NE\_CLUSTER\_NUMBER

FROM ebh\_details mnvxa

join VZWNET.CIRC\_PATH\_INST cpi on mnvxa.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mnvxa.xng\_path\_status <> 'Live'

AND mnvxa.match\_code = 'BOTH'

and mnvxa.XNG\_BANDWIDTH like '%bps'

and EI.TYPE = 'CSR'

MINUS

SELECT DISTINCT MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER, MNVXA.NE\_CLUSTER\_NUMBER

FROM ebh\_details mnvxa

join VZWNET.CIRC\_PATH\_INST cpi on mnvxa.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mnvxa.xng\_path\_status = 'Live'

AND mnvxa.match\_code = 'BOTH'

and mnvxa.XNG\_BANDWIDTH like '%bps'

and EI.TYPE = 'CSR')

GROUP BY MTX),

CONF\_BOTH\_COUNT AS -- for CONCONF BOTH

(SELECT MTX, COUNT (1) CONF\_BOTH\_COUNT

FROM (SELECT distinct MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER, MNVXA.NE\_SPAN\_NUMBER

FROM t1\_details mnvxa

WHERE MNVXA.NE\_BTS\_TYPE LIKE 'UB%'

AND MNVXA.NE\_CLUSTER\_NUMBER IS NULL

and match\_code='BOTH'

)

GROUP BY MTX),

CONF\_NE\_COUNT AS -- for CONCONF NE only

(SELECT mtx, COUNT (1) CONF\_NE\_COUNT

FROM (SELECT distinct mnvxa.mtx, mnvxa.ne\_bts\_number, mnvxa.NE\_SPAN\_NUMBER

FROM t1\_details mnvxa

WHERE mnvxa.ne\_bts\_type like 'UB%'

AND MNVXA.NE\_CLUSTER\_NUMBER IS NULL

AND MATCH\_CODE like 'NE%'

)

GROUP BY MTX) ,

ubs\_count AS

(SELECT MTX, COUNT (1) UBS\_COUNT

FROM (SELECT distinct MNVXA.MTX, MNVXA.NE\_BTS\_NUMBER, MNVXA.NE\_CLUSTER\_NUMBER, MNVXA.NE\_SPAN\_NUMBER

FROM audit\_details mnvxa

WHERE mnvxa.ne\_bts\_type like 'UB%'

)

GROUP BY mtx),

non\_ubs\_count AS

(SELECT mtx, COUNT (1) non\_ubs\_count

FROM (SELECT distinct mnvxa.mtx, mnvxa.ne\_bts\_number, mnvxa.ne\_cluster\_number, mnvxa.ne\_span\_number

FROM audit\_details mnvxa

WHERE mnvxa.ne\_bts\_type not like 'UB%'

)

GROUP BY mtx)

SELECT mrm.area, mrm.region, mrm.leaf\_domain\_name, mrm.mtx, mrm.switch\_id,

NVL (t1\_discovered, 0) AS t1\_discovered,

NVL (t1\_matched\_live, 0) AS t1\_matched\_live,

NVL (t1\_matched\_nonlive, 0) AS t1\_matched\_nonlive,

NVL (ETHERNET\_DISCOVERED, 0) AS Ebh\_DISCOVERED,

NVL (CONF\_BOTH\_COUNT, 0) AS EBH\_MATCHED\_LIVE,

NVL (CONF\_NE\_COUNT, 0) AS ebh\_matched\_nonlive,

NVL (ETHERNET\_MATCHED\_LIVE\_CSR, 0) AS Ebh\_MATCHED\_LIVE\_CSR,

NVL (ETHERNET\_MATCHED\_NONLIVE\_CSR, 0) AS ebh\_MATCHED\_NONLIVE\_CSR,

NVL (ubs\_count, 0) AS ubs\_count,

NVL (non\_ubs\_count, 0) AS non\_ubs\_count

FROM mtx\_region\_map mrm LEFT OUTER JOIN t1\_discovered td ON mrm.mtx = td.mtx

LEFT OUTER JOIN t1\_matched\_live tm ON mrm.mtx = tm.mtx

LEFT OUTER JOIN t1\_matched\_nonlive tum ON mrm.mtx = tum.mtx

LEFT OUTER JOIN ETHERNET\_DISCOVERED ED ON MRM.MTX = ED.MTX

LEFT OUTER JOIN CONF\_BOTH\_COUNT EM ON MRM.MTX = EM.MTX

LEFT OUTER JOIN CONF\_NE\_COUNT eum ON mrm.mtx = eum.mtx

LEFT OUTER JOIN ethernet\_matched\_live\_csr emc ON mrm.mtx = emc.mtx

LEFT OUTER JOIN ethernet\_matched\_nonlive\_csr eumc ON mrm.mtx = eumc.mtx

LEFT OUTER JOIN ubs\_count ubs ON mrm.mtx = ubs.mtx

LEFT OUTER JOIN NON\_UBS\_COUNT NUBS ON MRM.MTX = NUBS.MTX

ORDER BY mrm.mtx;

commit;

SQLSTMT := 'update '|| CDMA\_DEVICE\_SUMM ||' set t1\_comp =:t, ebh\_comp =:e, overall\_comp =:o

where rowid = :a';

for cur in updateCompliance

loop

T1COMP := NVL(CUR.T1\_DISCOVERED,0) - NVL(CUR.T1\_MATCHED\_NONLIVE,0);

ebhComp := nvl(cur.ebh\_discovered,0) - nvl(cur.ebh\_matched\_nonlive\_csr,0);

OVERALLCOMP :=NVL(CUR.T1\_DISCOVERED,0) + NVL(CUR.EBH\_DISCOVERED,0) ;

overallComp := overallComp - nvl(cur.t1\_matched\_nonlive,0) - nvl(cur.ebh\_matched\_nonlive\_csr,0);

if(t1Comp >0) then

t1Comp :=(nvl(cur.t1\_matched\_live,0)/t1Comp) \*100;

end if;

IF(EBHCOMP>0) THEN

ebhComp :=(nvl(cur.ebh\_matched\_live\_csr,0)/ebhComp) \*100;

end if;

IF(OVERALLCOMP>0) THEN

overallComp :=((nvl(cur.t1\_matched\_live,0)+nvl(cur.ebh\_matched\_live\_csr,0))/overallComp) \*100;

--dbms\_output.put\_line('good EBH '||overallComp);

end if;

EXECUTE IMMEDIATE SQLSTMT USING T1COMP, EBHCOMP, OVERALLCOMP, CUR.RD;

end loop;

commit;

MOTO\_CDMA\_REGIONAL\_SUMMARY();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlStmt);

dbms\_output.put\_line(SubStr('Error in moto\_cdma\_device\_summary();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

PROCEDURE moto\_evdo\_audit

IS

-- One Live and N non Live match

CURSOR t1\_nl\_match IS

SELECT aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier, ne\_span\_number

FROM moto\_evdo\_ne\_vs\_xng\_audit\_wrk

WHERE match\_code='BOTH' AND xng\_path\_status='Live'

AND termination\_type = 'T1';

-- One Live and N non Live match

CURSOR ebh\_nl\_match IS

SELECT aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier

FROM moto\_evdo\_ne\_vs\_xng\_audit\_wrk

WHERE match\_code='BOTH' AND xng\_path\_status='Live'

AND termination\_type = 'Ethernet';

-- T1 spans matching multiple live paths in xng or matching multiple non live paths in xng incase a live match does not exist are marked NE Only

CURSOR t1\_evdo\_multiples\_xng IS

WITH

dup AS

(

SELECT DISTINCT aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier, ne\_span\_number,

count(CASE WHEN xng\_path\_status = 'Live' THEN 1 ELSE NULL END) over (Partition by aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier, ne\_span\_number) as live\_count,

count(CASE WHEN xng\_path\_status != 'Live' THEN 1 ELSE NULL END) over (Partition by aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier, ne\_span\_number) as non\_live\_count

FROM moto\_evdo\_ne\_vs\_xng\_audit\_wrk src

WHERE src.match\_code='BOTH'

AND src.termination\_type='T1'

)

SELECT tgt.rowid

FROM moto\_evdo\_ne\_vs\_xng\_audit\_wrk tgt

JOIN dup

ON dup.aems = tgt.aems

AND dup.ne\_ipbscdo\_number= tgt.ne\_ipbscdo\_number

AND dup.ne\_mccdo\_identifier = tgt.ne\_mccdo\_identifier

AND dup.ne\_span\_number = tgt.ne\_span\_number

AND (dup.live\_count > 1 OR (dup.live\_count = 0 AND dup.non\_live\_count > 1)

);

-- Ethernet spans matching multiple live paths in xng or matching multiple non live paths in xng incase a live match does not exist are marked NE Only

CURSOR ethernet\_evdo\_multiples\_xng IS

WITH

dup AS

(

SELECT DISTINCT aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier, ne\_span\_number,

count(CASE WHEN xng\_path\_status = 'Live' THEN 1 ELSE NULL END) over (Partition by aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier) as live\_count,

count(CASE WHEN xng\_path\_status != 'Live' THEN 1 ELSE NULL END) over (Partition by aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier) as non\_live\_count

FROM moto\_evdo\_ne\_vs\_xng\_audit\_wrk src

WHERE src.match\_code='BOTH'

AND src.termination\_type='Ethernet'

)

SELECT tgt.rowid

FROM moto\_evdo\_ne\_vs\_xng\_audit\_wrk tgt

JOIN dup

ON dup.aems = tgt.aems

AND dup.ne\_ipbscdo\_number= tgt.ne\_ipbscdo\_number

AND dup.ne\_mccdo\_identifier = tgt.ne\_mccdo\_identifier

AND (dup.live\_count > 1 OR (dup.live\_count = 0 AND dup.non\_live\_count > 1)

);

--JC

-- Update those paths that dont have a CSR port

cursor update\_mevdo\_wrk\_no\_csr\_port is

WITH

MXA\_PATHS as

(

select mxa.XNG\_PATH\_INST\_ID

from moto\_evdo\_ne\_vs\_xng\_audit\_wrk mxa

where mxa.TERMINATION\_TYPE = 'Ethernet'

AND mxa.XNG\_BANDWIDTH LIKE '%bps'

and mxa.match\_code = 'BOTH'

and mxa.XNG\_PATH\_STATUS = 'Live'

)

,

PATHS\_W\_CSR as

(

select cpi.xng\_path\_inst\_id

from MXA\_PATHS cpi

join VZWNET.circ\_path\_element cpe

on cpi.xng\_path\_inst\_id = cpe.CIRC\_PATH\_INST\_ID

join vzwnet.epa p

on CPE.PORT\_INST\_ID = P.PORT\_INST\_ID

join VZWNET.equip\_inst ei

on p.equip\_INST\_ID = ei.equip\_INST\_ID

and ei.type = 'CSR'

)

select MXAP.xng\_path\_inst\_id

from MXA\_PATHS mxap

MINUS

select pc.XNG\_PATH\_INST\_ID

from PATHS\_W\_CSR pc;

sql\_stmt varchar2(32000);

BEGIN

sql\_stmt:='TRUNCATE TABLE moto\_evdo\_ne\_vs\_xng\_audit\_wrk ';

BEGIN

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in XngVsNortel();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

COMMIT;

sql\_stmt:='

INSERT INTO moto\_evdo\_ne\_vs\_xng\_audit\_wrk

(aems, ne\_ipbscdo\_number,ne\_mccdo\_identifier,ne\_span\_number,ne\_status, termination\_type,

match\_code,match\_status,xng\_path\_name,xng\_path\_inst\_id,

xng\_ipbscdo\_number,xng\_mccdo\_identifier,xng\_span\_number,

xng\_path\_type,xng\_path\_status,xng\_bandwidth,

extract\_date, audit\_date)

WITH ethernet\_terminations AS

(SELECT met.vsm\_device\_name\_aems aems,

met.ipbscdo\_number ne\_ipbscdo\_number,

met.mccdo\_number ne\_mccdo\_identifier, met.extract\_date,

met.ROWID ne\_rowid

FROM moto\_evdo\_ethernet\_inv met, moto\_vsm\_name\_aems mvn

where met.vsm\_device\_name\_aems = mvn.vsm\_device\_name\_aems

AND met.ipbscdo\_number = mvn.ipbscdo\_number

AND mvn.status = ''Live''),

ethernet\_audit AS

(SELECT met.aems, met.ne\_ipbscdo\_number, met.ne\_mccdo\_identifier,

NULL ne\_span\_number, NULL ne\_status, ''Ethernet'' termination\_type,

CASE

WHEN cpi.ipbscdo\_number IS NULL

THEN ''NE Only''

ELSE ''BOTH''

END match\_code,

CASE

WHEN cpi.span\_number IS NOT NULL

AND UPPER (cpi.bandwidth) LIKE ''%DS1%''

THEN ''Pathname Invalid Format, Path Invalid Bandwidth''

WHEN cpi.span\_number IS NOT NULL

AND UPPER (cpi.bandwidth) NOT LIKE ''%DS1%''

THEN ''Pathname Invalid Format''

WHEN cpi.span\_number IS NULL

AND UPPER (cpi.bandwidth) LIKE ''%DS1%''

THEN ''Path Invalid Bandwidth''

ELSE parse\_status

END match\_status,

cpi.path\_name xng\_path\_name,

cpi.circ\_path\_inst\_id xng\_path\_inst\_id,

cpi.ipbscdo\_number xng\_ipbscdo\_number,

cpi.mccdo\_number xng\_mccdo\_identifier,

cpi.span\_number xng\_span\_number, cpi.xng\_path\_type,

cpi.xng\_path\_status, cpi.bandwidth xng\_bandwidth,

met.extract\_date, cpi.ROWID cpi\_rowid, met.ne\_rowid

FROM ethernet\_terminations met LEFT OUTER JOIN vzw\_moto\_evdo\_cell\_paths cpi

ON met.ne\_ipbscdo\_number = cpi.ipbscdo\_number

AND met.ne\_mccdo\_identifier = cpi.mccdo\_number

),

t1\_terminations AS

(SELECT mtt.vsm\_device\_name\_aems aems,

mtt.ipbscdo\_number ne\_ipbscdo\_number,

mtt.mccdo\_number ne\_mccdo\_identifier,

mtt.span\_number ne\_span\_number, mtt.status ne\_status,

mtt.extract\_date, mtt.ROWID ne\_rowid

FROM moto\_evdo\_t1\_inv mtt, moto\_vsm\_name\_aems mvn

WHERE mtt.vsm\_device\_name\_aems = mvn.vsm\_device\_name\_aems

AND mtt.ipbscdo\_number = mvn.ipbscdo\_number

AND mvn.status = ''Live''),

t1\_audit AS

(SELECT mtt.aems, mtt.ne\_ipbscdo\_number, mtt.ne\_mccdo\_identifier,

mtt.ne\_span\_number, mtt.ne\_status, ''T1'' termination\_type,

CASE

WHEN cpi.ipbscdo\_number IS NULL

THEN ''NE Only''

ELSE ''BOTH''

END match\_code,

CASE

WHEN UPPER (cpi.bandwidth) NOT LIKE ''%DS1%''

THEN ''Path Invalid Bandwidth''

ELSE parse\_status

END match\_status,

cpi.path\_name xng\_path\_name,

cpi.circ\_path\_inst\_id xng\_path\_inst\_id,

cpi.ipbscdo\_number xng\_ipbscdo\_number,

cpi.mccdo\_number xng\_mccdo\_identifier,

cpi.span\_number xng\_span\_number, cpi.xng\_path\_type,

cpi.xng\_path\_status, cpi.bandwidth xng\_bandwidth,

mtt.extract\_date, cpi.ROWID cpi\_rowid, mtt.ne\_rowid

FROM t1\_terminations mtt LEFT OUTER JOIN vzw\_moto\_evdo\_cell\_paths cpi

ON mtt.ne\_ipbscdo\_number = cpi.ipbscdo\_number

AND mtt.ne\_mccdo\_identifier = cpi.mccdo\_number

AND mtt.ne\_span\_number = cpi.span\_number

),

ne\_only\_and\_xng AS

(SELECT \* FROM ethernet\_audit ea

UNION

SELECT \* FROM t1\_audit t1a),

xng\_only\_path\_id AS

(SELECT ROWID cpi\_rowid FROM vzw\_moto\_evdo\_cell\_paths cpi

MINUS

SELECT cpi\_rowid FROM ne\_only\_and\_xng),

xng\_only AS

(SELECT mvn.vsm\_device\_name\_aems aems, NULL ne\_ipbscdo\_number, NULL ne\_mccdo\_identifier,

NULL ne\_status, NULL ne\_span\_number,

CASE

WHEN cpi.bandwidth = ''DS1''

THEN ''T1''

ELSE ''Ethernet''

END termination\_type,

''Xng Only'' match\_code, parse\_status match\_status,

cpi.path\_name xng\_path\_name,

cpi.circ\_path\_inst\_id xng\_path\_inst\_id,

cpi.ipbscdo\_number xng\_ipbscdo\_number,

cpi.mccdo\_number xng\_mccdo\_identifier,

cpi.span\_number xng\_span\_number, cpi.xng\_path\_type,

cpi.xng\_path\_status, cpi.bandwidth xng\_bandwidth,

NULL extract\_date, cpi.ROWID cpi\_rowid, NULL ne\_rowid

FROM vzw\_moto\_evdo\_cell\_paths cpi, xng\_only\_path\_id xopi,moto\_vsm\_name\_aems mvn

WHERE cpi.ROWID = xopi.cpi\_rowid

and cpi.ipbscdo\_number=mvn.ipbscdo\_number),

full\_audit AS

(SELECT \*

FROM ne\_only\_and\_xng

UNION

SELECT \*

FROM xng\_only)

SELECT aems, ne\_ipbscdo\_number, ne\_mccdo\_identifier, ne\_span\_number,

ne\_status, termination\_type, match\_code, match\_status,xng\_path\_name,xng\_path\_inst\_id,

xng\_ipbscdo\_number, xng\_mccdo\_identifier, xng\_span\_number,

xng\_path\_type, xng\_path\_status, xng\_bandwidth, extract\_date,

TRUNC (SYSDATE) AS audit\_date

FROM full\_audit fa';

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Moto EVDO audit table created');

FOR cursorRec IN t1\_evdo\_multiples\_xng LOOP

UPDATE moto\_evdo\_ne\_vs\_xng\_audit\_wrk

set match\_status = nvl2(match\_status,match\_status||',Multiple spans found in Granite same tokens','Multiple spans found in Granite for same tokens') , match\_code='NE Only'

WHERE rowid = cursorRec.ROWID;

END LOOP;

COMMIT;

FOR cursorRec IN ethernet\_evdo\_multiples\_xng LOOP

UPDATE moto\_evdo\_ne\_vs\_xng\_audit\_wrk

set match\_status = nvl2(match\_status,match\_status||',Multiple spans found in Granite for same tokens','Multiple spans found in Granite for same tokens') , match\_code='NE Only'

WHERE rowid = cursorRec.ROWID;

END LOOP;

COMMIT;

UPDATE moto\_evdo\_ne\_vs\_xng\_audit\_wrk

SET match\_code = 'NE Only'

WHERE match\_code = 'BOTH' AND match\_status IS NOT NULL;

COMMIT;

FOR cursorRec IN t1\_nl\_match LOOP

UPDATE moto\_evdo\_ne\_vs\_xng\_audit\_wrk mnvxa

SET match\_status = nvl2(match\_status,'Live Path Exists,'||match\_status,'Live Path Exists')

WHERE mnvxa.aems=cursorRec.aems

AND mnvxa.ne\_ipbscdo\_number=cursorRec.ne\_ipbscdo\_number

AND mnvxa.ne\_mccdo\_identifier=cursorRec.ne\_mccdo\_identifier

AND mnvxa.ne\_span\_number=cursorRec.ne\_span\_number

AND mnvxa.termination\_type='T1'

AND mnvxa.xng\_path\_status <> 'Live';

END LOOP;

COMMIT;

FOR cursorRec IN ebh\_nl\_match LOOP

UPDATE moto\_evdo\_ne\_vs\_xng\_audit\_wrk mnvxa

SET match\_status = nvl2(match\_status,'Live Path Exists,'||match\_status,'Live Path Exists')

WHERE mnvxa.aems=cursorRec.aems

AND mnvxa.ne\_ipbscdo\_number=cursorRec.ne\_ipbscdo\_number

AND mnvxa.ne\_mccdo\_identifier=cursorRec.ne\_mccdo\_identifier

AND mnvxa.termination\_type='Ethernet'

AND mnvxa.xng\_path\_status <> 'Live';

END LOOP;

COMMIT;

--JC

FOR cursorRec IN update\_mevdo\_wrk\_no\_csr\_port LOOP

UPDATE moto\_evdo\_ne\_vs\_xng\_audit\_wrk mxa

SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS =

CASE

when MATCH\_STATUS = 'BOTH'

then 'No CSR port on the e-pipe'

else

nvl2(match\_status, 'No CSR port on the e-pipe, '||match\_status, 'No CSR port on the e-pipe')

END

WHERE

MXA.XNG\_PATH\_INST\_ID = cursorRec.XNG\_PATH\_INST\_ID

;

END LOOP;

commit;

--JC

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in moto\_evdo\_audit;: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

--

PROCEDURE moto\_evdo\_device\_summary IS

cursor updateCompliance is

select summ.rowid rd,summ.\* from moto\_evdo\_audit\_dev\_summ\_wrk summ;

sqlStmt VARCHAR2(32767);

sqlStmtT1Comp VARCHAR2(32767);

sqlStmtEbhComp VARCHAR2(32767);

sqlStmtOvlComp VARCHAR2(32767);

t1Comp NUMBER;

ebhComp NUMBER;

overallComp NUMBER;

BEGIN

sqlStmt := 'truncate table moto\_evdo\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_evdo\_audit\_dev\_summ\_wrk

(area,region,leaf\_domain,vsm\_device\_name\_aems,ipbscdo\_number,t1\_discovered,t1\_matched\_live,t1\_matched\_nonlive,ebh\_discovered,ebh\_matched\_live,ebh\_matched\_nonlive,ebh\_matched\_live\_csr,ebh\_matched\_nonlive\_csr

)

WITH aems\_region\_map AS

(SELECT DISTINCT cdmp.area, cdmp.region, cdmp.leaf\_domain\_name,

mvn.vsm\_device\_name\_aems aems,

mvn.ipbscdo\_number ne\_ipbscdo\_number

FROM xng\_reports.clli\_domain\_map\_v cdmp LEFT OUTER JOIN moto\_vsm\_name\_aems mvn

ON SUBSTR (mvn.vsm\_device\_name\_aems, 1, 6) =

SUBSTR (cdmp.clli, 1, 6)

WHERE cdmp.area <> ''NNO'' AND mvn.status = ''Live''),

t1\_discovered AS

(SELECT ne\_ipbscdo\_number, COUNT (1) t1\_discovered

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier, mea.ne\_span\_number

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''T1''

AND mea.ne\_status IN (''INS'', ''OOS'')

AND mea.match\_code IN (''BOTH'', ''NE Only''))

GROUP BY ne\_ipbscdo\_number),

ethernet\_discovered AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_discovered

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''Ethernet''

AND mea.match\_code IN (''BOTH'', ''NE Only''))

GROUP BY ne\_ipbscdo\_number),

t1\_matched\_live AS

(SELECT ne\_ipbscdo\_number, COUNT (1) t1\_matched\_live

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier, mea.ne\_span\_number

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''T1''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH'')

GROUP BY ne\_ipbscdo\_number),

t1\_matched\_nonlive AS

(SELECT ne\_ipbscdo\_number, COUNT (1) t1\_matched\_nonlive

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier, mea.ne\_span\_number

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''T1''

AND mea.xng\_path\_status <> ''Live''

AND mea.match\_code = ''BOTH''

MINUS

SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier, mea.ne\_span\_number

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''T1''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH'')

GROUP BY ne\_ipbscdo\_number),

ethernet\_matched\_live AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_matched\_live

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH'')

GROUP BY ne\_ipbscdo\_number),

ethernet\_matched\_live\_csr AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_matched\_live\_csr

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

join VZWNET.CIRC\_PATH\_INST cpi on mea.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH''

and mea.XNG\_BANDWIDTH like ''%bps''

and EI.TYPE = ''CSR'' )

GROUP BY ne\_ipbscdo\_number),

ethernet\_matched\_nonlive AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_matched\_nonlive

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status <> ''Live''

AND mea.match\_code = ''BOTH''

MINUS

SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH'')

GROUP BY ne\_ipbscdo\_number),

ethernet\_matched\_nonlive\_csr AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_matched\_nonlive\_csr

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

join VZWNET.CIRC\_PATH\_INST cpi on mea.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID OR CPI.CIRC\_PATH\_INST\_ID = P.NEXT\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status <> ''Live''

AND mea.match\_code = ''BOTH''

and mea.XNG\_BANDWIDTH like ''%bps''

and EI.TYPE = ''CSR''

MINUS

SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

join VZWNET.CIRC\_PATH\_INST cpi on mea.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH''

and mea.XNG\_BANDWIDTH like ''%bps''

and EI.TYPE = ''CSR'' )

GROUP BY ne\_ipbscdo\_number)

SELECT mrm.area, mrm.region, mrm.leaf\_domain\_name, mrm.aems,

mrm.ne\_ipbscdo\_number, NVL (t1\_discovered, 0) AS t1\_discovered,

NVL (t1\_matched\_live, 0) AS t1\_matched\_live,

NVL (t1\_matched\_nonlive, 0) AS t1\_matched\_nonlive,

NVL (ethernet\_discovered, 0) AS ethernet\_discovered,

NVL (ethernet\_matched\_live, 0) AS ethernet\_matched\_live,

NVL (ethernet\_matched\_nonlive, 0) AS ethernet\_matched\_nonlive,

NVL (ethernet\_matched\_live\_csr, 0) AS ethernet\_matched\_live\_csr,

NVL (ethernet\_matched\_nonlive\_csr, 0) AS ethernet\_matched\_nonlive\_csr

FROM aems\_region\_map mrm LEFT OUTER JOIN t1\_discovered td

ON mrm.ne\_ipbscdo\_number = td.ne\_ipbscdo\_number

LEFT OUTER JOIN t1\_matched\_live tm

ON mrm.ne\_ipbscdo\_number = tm.ne\_ipbscdo\_number

LEFT OUTER JOIN t1\_matched\_nonlive tum

ON mrm.ne\_ipbscdo\_number = tum.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_discovered ed

ON mrm.ne\_ipbscdo\_number = ed.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_matched\_live em

ON mrm.ne\_ipbscdo\_number = em.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_matched\_nonlive eum

ON mrm.ne\_ipbscdo\_number = eum.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_matched\_live\_csr emc

ON mrm.ne\_ipbscdo\_number = emc.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_matched\_nonlive\_csr eumc

ON mrm.ne\_ipbscdo\_number = eumc.ne\_ipbscdo\_number

ORDER BY mrm.ne\_ipbscdo\_number';

execute immediate sqlStmt;

commit;

sqlStmtT1Comp := 'update moto\_evdo\_audit\_dev\_summ\_wrk set t1\_comp =:x where rowid = :a';

sqlStmtEbhComp := 'update moto\_evdo\_audit\_dev\_summ\_wrk set ebh\_comp =:x where rowid = :a';

sqlStmtOvlComp := 'update moto\_evdo\_audit\_dev\_summ\_wrk set overall\_comp =:x where rowid = :a';

for cur in updateCompliance

loop

t1Comp := nvl(cur.t1\_discovered,0) - nvl(cur.t1\_matched\_nonlive,0);

ebhComp := nvl(cur.ebh\_discovered,0) - nvl(cur.ebh\_matched\_nonlive\_csr,0);

overallComp :=nvl(cur.t1\_discovered,0) + nvl(cur.ebh\_discovered,0) ;

overallComp := overallComp - nvl(cur.t1\_matched\_nonlive,0) - nvl(cur.ebh\_matched\_nonlive\_csr,0);

if(t1Comp >0) then

t1Comp :=(nvl(cur.t1\_matched\_live,0)/t1Comp) \*100;

t1Comp := round(t1Comp,2);

execute immediate sqlStmtT1Comp using t1Comp,cur.rd;

--dbms\_output.put\_line('Non null '|| t1Comp);

end if;

if(ebhComp>0) then

ebhComp :=(nvl(cur.ebh\_matched\_live\_csr,0)/ebhComp) \*100;

ebhComp := round(ebhComp,2);

execute immediate sqlStmtEbhComp using ebhComp,cur.rd;

--dbms\_output.put\_line('good EBH '||ebhComp);

end if;

if(overallComp>0) then

overallComp :=((nvl(cur.t1\_matched\_live,0)+nvl(cur.ebh\_matched\_live\_csr,0))/overallComp) \*100;

overallComp := round(overallComp,2);

execute immediate sqlStmtOvlComp using overallComp,cur.rd;

--dbms\_output.put\_line('good EBH '||overallComp);

end if;

end loop;

commit;

dbms\_output.put\_line('moto evdo device summary created');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlStmt);

dbms\_output.put\_line(SubStr('Error in populate\_evdo\_device\_summary();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

PROCEDURE moto\_evdo\_regional\_summary IS

CURSOR updateCompliance IS

SELECT summ.rowid rd,summ.\* FROM moto\_evdo\_audit\_reg\_summ\_wrk summ;

sqlStmt VARCHAR2(32767);

sqlStmtT1Comp VARCHAR2(32767);

sqlStmtEbhComp VARCHAR2(32767);

sqlStmtOvlComp VARCHAR2(32767);

t1Comp NUMBER;

ebhComp NUMBER;

overallComp NUMBER;

t1Reported NUMBER;

ebhReported NUMBER;

BEGIN

sqlStmt := 'truncate table moto\_evdo\_audit\_reg\_summ\_wrk';

execute IMMEDIATE sqlStmt;

sqlStmt := 'INSERT INTO moto\_evdo\_audit\_reg\_summ\_wrk

(area,region,t1\_discovered,t1\_matched\_live,t1\_matched\_nonlive,ebh\_discovered,ebh\_matched\_live,ebh\_matched\_nonlive,ebh\_matched\_live\_csr,ebh\_matched\_nonlive\_csr

)

/\* Formatted on 2011/03/03 13:08 (Formatter Plus v4.8.8) \*/

WITH aems\_region\_map AS

(SELECT DISTINCT cdmp.area, cdmp.region, cdmp.leaf\_domain\_name,

mvn.vsm\_device\_name\_aems aems,

mvn.ipbscdo\_number ne\_ipbscdo\_number

FROM xng\_reports.clli\_domain\_map\_v cdmp LEFT OUTER JOIN moto\_vsm\_name\_aems mvn

ON SUBSTR (mvn.vsm\_device\_name\_aems, 1, 6) =

SUBSTR (cdmp.clli, 1, 6)

WHERE cdmp.area <> ''NNO''),

t1\_discovered AS

(SELECT ne\_ipbscdo\_number, COUNT (1) t1\_discovered

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier, mea.ne\_span\_number

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''T1''

AND mea.ne\_status IN (''INS'', ''OOS'')

AND mea.match\_code IN (''BOTH'', ''NE Only''))

GROUP BY ne\_ipbscdo\_number),

ethernet\_discovered AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_discovered

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''Ethernet''

AND mea.match\_code IN (''BOTH'', ''NE Only''))

GROUP BY ne\_ipbscdo\_number),

t1\_matched\_live AS

(SELECT ne\_ipbscdo\_number, COUNT (1) t1\_matched\_live

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier, mea.ne\_span\_number

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''T1''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH'')

GROUP BY ne\_ipbscdo\_number),

t1\_matched\_nonlive AS

(SELECT ne\_ipbscdo\_number, COUNT (1) t1\_matched\_nonlive

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier, mea.ne\_span\_number

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''T1''

AND mea.xng\_path\_status <> ''Live''

AND mea.match\_code = ''BOTH''

MINUS

SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier, mea.ne\_span\_number

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''T1''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH'')

GROUP BY ne\_ipbscdo\_number),

ethernet\_matched\_live AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_matched\_live

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH'')

GROUP BY ne\_ipbscdo\_number),

ethernet\_matched\_live\_csr AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_matched\_live\_csr

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

join VZWNET.CIRC\_PATH\_INST cpi on mea.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH''

AND mea.XNG\_BANDWIDTH like ''%bps''

AND EI.TYPE = ''CSR'')

GROUP BY ne\_ipbscdo\_number),

ethernet\_matched\_nonlive AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_matched\_nonlive

FROM (SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status <> ''Live''

AND mea.match\_code = ''BOTH''

MINUS

SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH'')

GROUP BY ne\_ipbscdo\_number),

ethernet\_matched\_nonlive\_csr AS

(SELECT ne\_ipbscdo\_number, COUNT (1) ethernet\_matched\_nonlive\_csr

FROM ((SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

join VZWNET.CIRC\_PATH\_INST cpi on mea.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID OR CPI.CIRC\_PATH\_INST\_ID = P.NEXT\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status <> ''Live''

AND mea.match\_code = ''BOTH''

AND mea.XNG\_BANDWIDTH like ''%bps''

AND EI.TYPE = ''CSR'')

MINUS

(SELECT DISTINCT mea.aems, mea.ne\_ipbscdo\_number,

mea.ne\_mccdo\_identifier

FROM xng\_reports.moto\_evdo\_ne\_vs\_xng\_audit\_wrk mea

join VZWNET.CIRC\_PATH\_INST cpi on mea.xng\_path\_inst\_id = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE mea.termination\_type = ''Ethernet''

AND mea.xng\_path\_status = ''Live''

AND mea.match\_code = ''BOTH''

AND mea.XNG\_BANDWIDTH like ''%bps''

AND EI.TYPE = ''CSR''))

GROUP BY ne\_ipbscdo\_number)

SELECT mrm.area, mrm.region, SUM (NVL (t1\_discovered, 0)) AS t1\_discovered,

SUM (NVL (t1\_matched\_live, 0)) AS t1\_matched\_live,

SUM (NVL (t1\_matched\_nonlive, 0)) AS t1\_matched\_nonlive,

SUM (NVL (ethernet\_discovered, 0)) AS ethernet\_discovered,

SUM (NVL (ethernet\_matched\_live, 0)) AS ethernet\_matched\_live,

SUM (NVL (ethernet\_matched\_nonlive, 0)) AS ethernet\_matched\_nonlive,

SUM (NVL (ethernet\_matched\_live\_csr, 0)) AS ethernet\_matched\_live\_csr,

SUM (NVL (ethernet\_matched\_nonlive\_csr, 0)) AS ethernet\_matched\_nonlive\_csr

FROM aems\_region\_map mrm LEFT OUTER JOIN t1\_discovered td

ON mrm.ne\_ipbscdo\_number = td.ne\_ipbscdo\_number

LEFT OUTER JOIN t1\_matched\_live tm

ON mrm.ne\_ipbscdo\_number = tm.ne\_ipbscdo\_number

LEFT OUTER JOIN t1\_matched\_nonlive tum

ON mrm.ne\_ipbscdo\_number = tum.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_discovered ed

ON mrm.ne\_ipbscdo\_number = ed.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_matched\_live em

ON mrm.ne\_ipbscdo\_number = em.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_matched\_nonlive eum

ON mrm.ne\_ipbscdo\_number = eum.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_matched\_live\_csr emc

ON mrm.ne\_ipbscdo\_number = emc.ne\_ipbscdo\_number

LEFT OUTER JOIN ethernet\_matched\_nonlive\_csr eumc

ON mrm.ne\_ipbscdo\_number = eumc.ne\_ipbscdo\_number

GROUP BY ROLLUP (mrm.area, mrm.region)

ORDER BY mrm.area, mrm.region';

execute IMMEDIATE sqlStmt;

COMMIT;

sqlStmtT1Comp := 'update moto\_evdo\_audit\_reg\_summ\_wrk set t1\_comp =:x ,t1\_reported =:y where rowid = :a';

sqlStmtEbhComp := 'update moto\_evdo\_audit\_reg\_summ\_wrk set ebh\_comp =:x,ebh\_reported =:y where rowid =:a';

sqlStmtOvlComp := 'update moto\_evdo\_audit\_reg\_summ\_wrk set overall\_comp =:x where rowid =:a';

for cur in updateCompliance

loop

t1Reported := nvl(cur.t1\_discovered,0) - nvl(cur.t1\_matched\_nonlive,0);

ebhReported := nvl(cur.ebh\_discovered,0) - nvl(cur.ebh\_matched\_nonlive\_csr,0);

overallComp :=nvl(cur.t1\_discovered,0) + nvl(cur.ebh\_discovered,0) ;

overallComp := overallComp - nvl(cur.t1\_matched\_nonlive,0) - nvl(cur.ebh\_matched\_nonlive\_csr,0);

if(t1Reported >0) then

t1Comp :=(nvl(cur.t1\_matched\_live,0)/t1Reported) \*100;

t1Comp := round(t1Comp,2);

execute immediate sqlStmtT1Comp using t1Comp,t1Reported,cur.rd;

--dbms\_output.put\_line('Non null '|| t1Comp);

end if;

if(ebhReported>0) then

ebhComp :=(nvl(cur.ebh\_matched\_live\_csr,0)/ebhReported) \*100;

ebhComp := round(ebhComp,2);

execute immediate sqlStmtEbhComp using ebhComp,ebhReported,cur.rd;

--dbms\_output.put\_line('good EBH '||ebhComp);

end if;

if(overallComp>0) then

overallComp :=((nvl(cur.t1\_matched\_live,0)+nvl(cur.ebh\_matched\_live\_csr,0))/overallComp) \*100;

overallComp := round(overallComp,2);

execute immediate sqlStmtOvlComp using overallComp,cur.rd;

--dbms\_output.put\_line('good EBH '||overallComp);

end if;

end loop;

COMMIT;

dbms\_output.put\_line('moto evdo regional summary created');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlStmt);

dbms\_output.put\_line(SubStr('Error in populate\_evdo\_regional\_summary();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

PROCEDURE copy\_moto\_cdma\_wrk IS

sqlStmt varchar2(32000);

v\_ready xng\_reports.all\_processes.is\_ready%type;

BEGIN

select is\_ready

into v\_ready

from xng\_reports.all\_processes

where process\_name = 'MOTO\_CDMA\_AUDIT';

IF v\_ready = 'Y' then

sqlStmt := 'delete from moto\_cdma\_ne\_vs\_xng\_audit';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_cdma\_ne\_vs\_xng\_audit select \* from moto\_cdma\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_cdma\_ne\_vs\_xng\_audit');

sqlStmt := 'delete from moto\_cdma\_audit\_reg\_summ';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_cdma\_audit\_reg\_summ select \* from moto\_cdma\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_cdma\_audit\_reg\_summ');

sqlStmt := 'delete from moto\_cdma\_audit\_dev\_summ';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_cdma\_audit\_dev\_summ select \* from moto\_cdma\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_cdma\_audit\_dev\_summ');

END IF;

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('MOTO\_CDMA\_AUDIT',4000,SubStr('Error in copy\_moto\_cdma\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('MOTO\_CDMA\_AUDIT','STATUS\_FAILURE','N');

END;

PROCEDURE copy\_moto\_evdo\_wrk IS

sqlStmt varchar2(32000);

v\_ready xng\_reports.all\_processes.is\_ready%type;

BEGIN

select is\_ready

into v\_ready

from xng\_reports.all\_processes

where process\_name = 'MOTO\_EVDO\_AUDIT';

IF v\_ready = 'Y' then

sqlStmt := 'delete from moto\_evdo\_ne\_vs\_xng\_audit';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_evdo\_ne\_vs\_xng\_audit select \* from moto\_evdo\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_evdo\_ne\_vs\_xng\_audit');

sqlStmt := 'delete from moto\_evdo\_audit\_reg\_summ';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_evdo\_audit\_reg\_summ select \* from moto\_evdo\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_evdo\_audit\_reg\_summ');

sqlStmt := 'delete from moto\_evdo\_audit\_dev\_summ';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_evdo\_audit\_dev\_summ select \* from moto\_evdo\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_evdo\_audit\_dev\_summ');

END IF;

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('MOTO\_EVDO\_AUDIT',4000,SubStr('Error in copy\_moto\_evdo\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('MOTO\_EVDO\_AUDIT','STATUS\_FAILURE','N');

END;

PROCEDURE truncate\_moto\_cdma\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from moto\_cdma\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting moto\_cdma\_ne\_vs\_xng\_audit\_wrk');

sqlStmt := 'delete from moto\_cdma\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting moto\_cdma\_audit\_reg\_summ\_wrk');

sqlStmt := 'delete from moto\_cdma\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting moto\_cdma\_audit\_dev\_summ\_wrk');

END;

PROCEDURE truncate\_moto\_evdo\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from moto\_evdo\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

dbms\_output.put\_line('Done deleting moto\_evdo\_ne\_vs\_xng\_audit\_wrk');

sqlStmt := 'delete from moto\_evdo\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting moto\_evdo\_audit\_reg\_summ\_wrk');

sqlStmt := 'delete from moto\_evdo\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting moto\_evdo\_audit\_dev\_summ\_wrk');

END;

PROCEDURE restore\_moto\_cdma\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from moto\_cdma\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_cdma\_ne\_vs\_xng\_audit\_wrk select \* from moto\_cdma\_ne\_vs\_xng\_audit';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_cdma\_ne\_vs\_xng\_audit\_wrk');

sqlStmt := 'delete from moto\_cdma\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_cdma\_audit\_reg\_summ\_wrk select \* from moto\_cdma\_audit\_reg\_summ';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_cdma\_audit\_reg\_summ\_wrk');

sqlStmt := 'delete from moto\_cdma\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_cdma\_audit\_dev\_summ\_wrk select \* from moto\_cdma\_audit\_dev\_summ';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_cdma\_audit\_dev\_summ\_wrk');

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('MOTO\_CDMA\_AUDIT',4000,SubStr('Error in restore\_moto\_cdma\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('MOTO\_CDMA\_AUDIT','STATUS\_FAILURE','N');

END;

PROCEDURE restore\_moto\_evdo\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from moto\_evdo\_ne\_vs\_xng\_audit\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_evdo\_ne\_vs\_xng\_audit\_wrk select \* from moto\_evdo\_ne\_vs\_xng\_audit';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_evdo\_ne\_vs\_xng\_audit\_wrk');

sqlStmt := 'delete from moto\_evdo\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_evdo\_audit\_reg\_summ\_wrk select \* from moto\_evdo\_audit\_reg\_summ';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_evdo\_audit\_reg\_summ\_wrk');

sqlStmt := 'delete from moto\_evdo\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into moto\_evdo\_audit\_dev\_summ\_wrk select \* from moto\_evdo\_audit\_dev\_summ';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating moto\_evdo\_audit\_dev\_summ\_wrk');

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('MOTO\_EVDO\_AUDIT',4000,SubStr('Error in restore\_moto\_evdo\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('MOTO\_EVDO\_AUDIT','STATUS\_FAILURE','N');

END;

END;

/

--------------------------------------------------------

-- DDL for Package Body MPLS\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."MPLS\_AUDIT"

IS

--

-- To modify this template, edit file PKGBODY.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package body

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter procedure, function bodies as shown below

PROCEDURE copy\_mpls\_data\_from\_wk

IS

sql\_stmt varchar2(32000);

begin

sql\_stmt:='truncate table '||MPLS\_SUMMARY\_BY\_DEVICE;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into '||MPLS\_SUMMARY\_BY\_DEVICE||' select \* from '||MPLS\_SUMMARY\_BY\_DEVICE\_WK;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating '||MPLS\_SUMMARY\_BY\_DEVICE);

sql\_stmt:='truncate table '||MPLS\_SUMMARY\_BY\_REGION;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into '||MPLS\_SUMMARY\_BY\_REGION||' select \* from '||MPLS\_SUMMARY\_BY\_REGION\_WK;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating '||MPLS\_SUMMARY\_BY\_REGION);

sql\_stmt:='truncate table '||MPLS\_NNO\_SUMMARY\_BY\_DEVICE;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into '||MPLS\_NNO\_SUMMARY\_BY\_DEVICE||' select \* from '||MPLS\_NNO\_SUMMARY\_BY\_DEVICE\_WK;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating '||MPLS\_NNO\_SUMMARY\_BY\_DEVICE);

sql\_stmt:='truncate table '||MPLS\_NNO\_SUMMARY\_BY\_REGION;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into '||MPLS\_NNO\_SUMMARY\_BY\_REGION||' select \* from '||MPLS\_NNO\_SUMMARY\_BY\_REGION\_WK;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating '||MPLS\_NNO\_SUMMARY\_BY\_REGION);

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in copy\_mpls\_data\_from\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end copy\_mpls\_data\_from\_wk;

--RESTORE FROM PROD TO WORK TABLES

PROCEDURE copy\_mpls\_data\_from\_prod\_to\_wk

IS

sql\_stmt varchar2(32000);

begin

sql\_stmt:='truncate table '||MPLS\_SUMMARY\_BY\_DEVICE\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into '||MPLS\_SUMMARY\_BY\_DEVICE\_WK||' select \* from '||MPLS\_SUMMARY\_BY\_DEVICE;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating '||MPLS\_SUMMARY\_BY\_DEVICE\_WK);

sql\_stmt:='truncate table '||MPLS\_SUMMARY\_BY\_REGION\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into '||MPLS\_SUMMARY\_BY\_REGION\_WK||' select \* from '||MPLS\_SUMMARY\_BY\_REGION;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating '||MPLS\_SUMMARY\_BY\_REGION\_WK);

sql\_stmt:='truncate table '||MPLS\_NNO\_SUMMARY\_BY\_DEVICE\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into '||MPLS\_NNO\_SUMMARY\_BY\_DEVICE\_WK||' select \* from '||MPLS\_NNO\_SUMMARY\_BY\_DEVICE;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating '||MPLS\_NNO\_SUMMARY\_BY\_DEVICE\_WK);

sql\_stmt:='truncate table '||MPLS\_NNO\_SUMMARY\_BY\_REGION\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt:='insert into '||MPLS\_NNO\_SUMMARY\_BY\_REGION\_WK||' select \* from '||MPLS\_NNO\_SUMMARY\_BY\_REGION;

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating '||MPLS\_NNO\_SUMMARY\_BY\_REGION\_WK);

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in copy\_mpls\_data\_from\_prod\_to\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end copy\_mpls\_data\_from\_prod\_to\_wk;

END;

/

--------------------------------------------------------

-- DDL for Package Body MSPP\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."MSPP\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: MSPP\_AUDIT

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 7/7/2011 Satya Modugula 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE copy\_mspp\_data\_from\_wk\_to\_prod

IS

sql\_stmt VARCHAR2 (32000);

v\_ready xng\_reports.all\_processes.is\_ready%TYPE;

BEGIN

SELECT is\_ready

INTO v\_ready

FROM xng\_reports.all\_processes

WHERE process\_name = 'INC\_MSPP\_AUDIT';

IF v\_ready = 'Y'

THEN

sql\_stmt := 'truncate table ' || INC\_MSPP\_XCONN\_AUDIT\_DETAILS;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| INC\_MSPP\_XCONN\_AUDIT\_DETAILS

|| ' select \* from '

|| INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || INC\_MSPP\_XCONN\_AUDIT\_DETAILS);

sql\_stmt := 'truncate table ' || INC\_MSPP\_XCONN\_AUDIT\_SUMMARY;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| INC\_MSPP\_XCONN\_AUDIT\_SUMMARY

|| ' select \* from '

|| INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || INC\_MSPP\_XCONN\_AUDIT\_SUMMARY);

sql\_stmt := 'truncate table ' || MSPP\_CIRC\_PATH\_ELEMENTS;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| MSPP\_CIRC\_PATH\_ELEMENTS

|| ' select \* from '

|| MSPP\_CIRC\_PATH\_ELEMENTS\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || MSPP\_CIRC\_PATH\_ELEMENTS);

sql\_stmt := 'truncate table ' || MSPP\_SUMMARY\_REGION\_MAP;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| MSPP\_SUMMARY\_REGION\_MAP

|| ' select \* from '

|| MSPP\_SUMMARY\_REGION\_MAP\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || MSPP\_SUMMARY\_REGION\_MAP);

sql\_stmt := 'truncate table ' || MSPP\_VSM\_MATCH;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| MSPP\_VSM\_MATCH

|| ' select \* from '

|| MSPP\_VSM\_MATCH\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || MSPP\_VSM\_MATCH);

sql\_stmt := 'truncate table ' || INC\_MSPP\_ADM\_COMP\_SUMMARY;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| INC\_MSPP\_ADM\_COMP\_SUMMARY

|| ' select \* from '

|| INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || INC\_MSPP\_ADM\_COMP\_SUMMARY);

sql\_stmt := 'truncate table ' || INC\_MSPP\_COMPLIANCE\_SUMMARY;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| INC\_MSPP\_COMPLIANCE\_SUMMARY

|| ' select \* from '

|| INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || INC\_MSPP\_COMPLIANCE\_SUMMARY);

END IF;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line (sql\_stmt);

DBMS\_OUTPUT.put\_line (SUBSTR ( 'Error in copy\_mspp\_data\_from\_wk\_to\_prod(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror ('INC\_MSPP\_AUDIT',

4000,

SUBSTR ( 'Error in copy\_mspp\_data\_from\_wk\_to\_prod(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend ('INC\_MSPP\_AUDIT', 'STATUS\_FAILURE', 'N');

-- RAISE;

END copy\_mspp\_data\_from\_wk\_to\_prod;

PROCEDURE copy\_mspp\_data\_from\_prod\_to\_wk

IS

sql\_stmt VARCHAR2 (32000);

BEGIN

sql\_stmt := 'truncate table ' || INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK

|| ' select \* from '

|| INC\_MSPP\_XCONN\_AUDIT\_DETAILS;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK);

sql\_stmt := 'truncate table ' || INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK

|| ' select \* from '

|| INC\_MSPP\_XCONN\_AUDIT\_SUMMARY;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK);

sql\_stmt := 'truncate table ' || MSPP\_CIRC\_PATH\_ELEMENTS\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| MSPP\_CIRC\_PATH\_ELEMENTS\_WK

|| ' select \* from '

|| MSPP\_CIRC\_PATH\_ELEMENTS;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || MSPP\_CIRC\_PATH\_ELEMENTS\_WK);

sql\_stmt := 'truncate table ' || MSPP\_SUMMARY\_REGION\_MAP\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| MSPP\_SUMMARY\_REGION\_MAP\_WK

|| ' select \* from '

|| MSPP\_SUMMARY\_REGION\_MAP;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || MSPP\_SUMMARY\_REGION\_MAP\_WK);

sql\_stmt := 'truncate table ' || MSPP\_VSM\_MATCH\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| MSPP\_VSM\_MATCH\_WK

|| ' select \* from '

|| MSPP\_VSM\_MATCH;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || MSPP\_VSM\_MATCH\_WK);

sql\_stmt := 'truncate table ' || INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK

|| ' select \* from '

|| INC\_MSPP\_ADM\_COMP\_SUMMARY;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK);

sql\_stmt := 'truncate table ' || INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK

|| ' select \* from '

|| INC\_MSPP\_COMPLIANCE\_SUMMARY;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Done populating ' || INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK);

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line (sql\_stmt);

DBMS\_OUTPUT.put\_line (SUBSTR ( 'Error in copy\_mspp\_data\_from\_prod\_to\_wk(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror ('INC\_MSPP\_AUDIT',

4000,

SUBSTR ( 'Error in copy\_mspp\_data\_from\_prod\_to\_wk(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend ('INC\_MSPP\_AUDIT', 'STATUS\_FAILURE', 'N');

--RAISE;

END copy\_mspp\_data\_from\_prod\_to\_wk;

PROCEDURE truncate\_mspp\_wk

IS

sql\_stmt VARCHAR2 (32000);

BEGIN

sql\_stmt := 'delete from ' || INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Deleted from ' || INCMSPP\_XCONN\_AUDIT\_DETAILS\_WK);

sql\_stmt := 'delete from ' || INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Deleted from ' || INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK);

sql\_stmt := 'delete from ' || MSPP\_CIRC\_PATH\_ELEMENTS\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Deleted from ' || MSPP\_CIRC\_PATH\_ELEMENTS\_WK);

sql\_stmt := 'delete from ' || MSPP\_SUMMARY\_REGION\_MAP\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Deleted from ' || MSPP\_SUMMARY\_REGION\_MAP\_WK);

sql\_stmt := 'delete from ' || MSPP\_VSM\_MATCH\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Deleted from ' || MSPP\_VSM\_MATCH\_WK);

sql\_stmt := 'delete from ' || INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Deleted from ' || INC\_MSPP\_ADM\_COMP\_SUMMARY\_WK);

sql\_stmt := 'delete from ' || INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

DBMS\_OUTPUT.put\_line ('Deleted from ' || INC\_MSPP\_COMPLIANCE\_SUMMARY\_WK);

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line (sql\_stmt);

DBMS\_OUTPUT.put\_line (SUBSTR ( 'Error in truncate\_mspp\_wk(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror ('INC\_MSPP\_AUDIT',

4000,

SUBSTR ( 'Error in truncate\_mspp\_wk(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend ('INC\_MSPP\_AUDIT', 'STATUS\_FAILURE', 'N');

--RAISE;

END truncate\_mspp\_wk;

END MSPP\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package Body MWR\_DISCOVERY

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."MWR\_DISCOVERY" is

PROCEDURE load\_xng\_mwr\_paths is

sqlStmt varchar2(32000);

message varchar2(250);

methodName varchar2(30) := 'load\_xng\_mwr\_paths';

BEGIN

message := 'Error: '|| methodName ||'(): Can''t truncate table: XNG\_REPORTS.XNG\_MWR\_PATHS\_WK';

sqlStmt := 'truncate table XNG\_REPORTS.XNG\_MWR\_PATHS\_WK' ;

execute immediate sqlStmt;

message := 'Error: '|| methodName ||'(): Can''t insert into table: XNG\_REPORTS.XNG\_MWR\_PATHS\_WK';

sqlStmt:='

insert into XNG\_REPORTS.XNG\_MWR\_PATHS\_WK columns(EQUIP\_INST\_ID ,PORT\_INST\_ID,

CIRC\_PATH\_INST\_ID, NEXT\_PATH\_INST\_ID,MWR\_LIVE\_IN\_XNG, PATH\_LIVE\_IN\_XNG, IS\_GIGE)

WITH

GET\_ALL\_MWR as (

select ei.equip\_inst\_id, decode (ei.status, ''Live'', ''Y'', ''N'') mwr\_live\_in\_xng

from XNG\_MWR\_PARSED\_WK csr, vzwnet.equip\_inst ei

where ei.EQUIP\_INST\_ID = csr.EQUIP\_INST\_ID

)

,

GET\_MWR\_PORTS as

(

select p.site\_inst\_id, ei.\*, p.PORT\_INST\_ID, p.PORT\_HUM\_ID, p.BANDWIDTH

, p.CIRC\_PATH\_INST\_ID, p.NEXT\_PATH\_INST\_ID

, decode(cpi.STATUS, ''Live'', ''Y'', ''N'') path\_live\_in\_xng

from GET\_ALL\_MWR ei

join vzwnet.epa p

on ei.equip\_inst\_id = p.equip\_inst\_id

left outer join vzwnet.circ\_path\_inst cpi

on cpi.CIRC\_PATH\_INST\_ID = p.CIRC\_PATH\_INST\_ID

left outer join vzwnet.circ\_path\_inst npi

on npi.CIRC\_PATH\_INST\_ID = p.Next\_PATH\_INST\_ID

)select equip\_inst\_id,port\_inst\_id, circ\_path\_inst\_id, next\_path\_inst\_id, mwr\_live\_in\_xng, path\_live\_in\_xng, null

from GET\_MWR\_PORTS gmp

where

circ\_path\_inst\_id is not null or next\_path\_inst\_id is not null'

;

execute immediate sqlStmt;

commit;

sqlStmt:=' update xng\_reports.xng\_mwr\_paths\_wk wk set wk.is\_gige=''Y'' where wk.circ\_path\_inst\_id in

( select wk.circ\_path\_inst\_id from xng\_mwr\_paths\_wk m , xng\_reports.xng\_csr\_gige\_paths\_wk gige

where ( M.CIRC\_PATH\_INST\_ID = GIGE.CIRC\_PATH\_INST\_ID

or

M.CIRC\_PATH\_INST\_ID = GIGE.NEXT\_PATH\_INST\_ID)

)';

execute immediate sqlStmt;

commit;

sqlStmt:=' update xng\_reports.xng\_mwr\_paths\_wk wk set wk.is\_gige=''Y'' where wk.next\_path\_inst\_id in

( select wk.next\_path\_inst\_id from xng\_mwr\_paths\_wk m , xng\_reports.xng\_csr\_gige\_paths\_wk gige

where ( M.NEXT\_PATH\_INST\_ID = GIGE.CIRC\_PATH\_INST\_ID

or

M.NEXT\_PATH\_INST\_ID = GIGE.NEXT\_PATH\_INST\_ID)

)';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr(message||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END load\_xng\_mwr\_paths;

END mwr\_discovery;

/

--------------------------------------------------------

-- DDL for Package Body N2N\_OVERALL\_COMPLIANCE

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."N2N\_OVERALL\_COMPLIANCE" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: n2n\_overall\_compliance

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 1/27/2012 1. Created this package body.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE takeMonthlySnapshot IS

PROCESS\_NAME varchar2(100) := 'N2N\_OBS\_MONTHLY';

sqlStmt varchar2(200);

BEGIN

watchdog.updateProcessStart(PROCESS\_NAME);

sqlStmt := 'truncate table N2N\_OVERALL\_SUMM\_INTER';

execute immediate sqlStmt;

-- save a copy of the current from the view. view has deltas as well.

-- so cant do all this in one step

insert into N2N\_OVERALL\_SUMM\_INTER columns (AREA, REGION, LIVE, RPTD, ENB\_TOTAL\_COMPLIANCE,

E2BTS\_TOTAL\_COMPLIANCE, EBH\_TOTAL\_COMPLIANCE,

T1\_TOTAL\_COMPLIANCE, CSR\_TOTAL\_COMPLIANCE,

OVERALL\_COMPLIANCE, REGIONAL\_MONTHLY\_DELTA, VLAN\_TOTAL\_COMPLIANCE

)

select AREA, REGION, LIVE, RPTD, ENB\_TOTAL\_COMPLIANCE,

E2BTS\_TOTAL\_COMPLIANCE, EBH\_TOTAL\_COMPLIANCE,

T1\_TOTAL\_COMPLIANCE, CSR\_TOTAL\_COMPLIANCE,

OVERALL\_COMPLIANCE, REGIONAL\_MONTHLY\_DELTA, VLAN\_TOTAL\_COMPLIANCE

from N2N\_OVERALL\_COMPLIANCE\_SUMM;

commit;

sqlStmt := 'truncate table N2N\_OVERALL\_SUMM\_PREV\_MONTH';

execute immediate sqlStmt;

--create monthly snapshot from intemediate file

insert into N2N\_OVERALL\_SUMM\_PREV\_MONTH columns (AREA, REGION, LIVE, RPTD, ENB\_TOTAL\_COMPLIANCE,

E2BTS\_TOTAL\_COMPLIANCE, EBH\_TOTAL\_COMPLIANCE,

T1\_TOTAL\_COMPLIANCE, CSR\_TOTAL\_COMPLIANCE,

OVERALL\_COMPLIANCE, REGIONAL\_MONTHLY\_DELTA, VLAN\_TOTAL\_COMPLIANCE

)

select AREA, REGION, LIVE, RPTD, ENB\_TOTAL\_COMPLIANCE,

E2BTS\_TOTAL\_COMPLIANCE, EBH\_TOTAL\_COMPLIANCE,

T1\_TOTAL\_COMPLIANCE, CSR\_TOTAL\_COMPLIANCE,

OVERALL\_COMPLIANCE, REGIONAL\_MONTHLY\_DELTA, VLAN\_TOTAL\_COMPLIANCE

from N2N\_OVERALL\_SUMM\_INTER;

commit;

--sqlStmt := 'truncate table N2N\_OVERALL\_SUMM\_INTER';

--execute immediate sqlStmt;

watchdog.updateProcessEnd(PROCESS\_NAME,'STATUS\_SUCCESS','Y');

END;

END n2n\_overall\_compliance;

/

--------------------------------------------------------

-- DDL for Package Body NCM\_CSR\_VLAN\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDIT"

IS

PROCEDURE audit\_ci\_csr\_vlans

IS

methodname VARCHAR2 (30) := 'audit\_CI\_csr\_vlans ';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE ('delete NCM\_CSR\_VLAN\_AUDIT\_WK where CSR\_vendor = ''CI''');

COMMIT;

sqlstmt :=

'insert into NCM\_CSR\_VLAN\_AUDIT\_WK columns (CSR\_Device\_Name, CSR\_vendor

, ne\_hostName, displayed\_name, device\_ip, ne\_vlan\_number

, match\_code, match\_status

, xng\_vlan\_number

, vlan\_inst\_id

, vlan\_status

, description

)

WITH

ci\_vlans AS

(SELECT xnvp.\*, xne.CSR\_DEVICE\_NAME

FROM xng\_CSR\_vlan\_paths\_wk xnvp , XNG\_CSR\_PARSED\_WK xne

where xne.EQUIP\_INST\_ID = xnvp.EQUIP\_INST\_ID and

xne.CSR\_DEVICE\_NAME like ''%-CI-%'' AND xnvp.vlan\_number IS NOT NULL and length(xnvp.vlan\_number) > 3),

matches AS

(SELECT aud.csr\_device\_name, ncm.VLAN\_NUMBER ne\_vlan\_number,

ncm.SUB\_INTERFACE\_NAME, ncm.DEVICE\_IP, ''BOTH'' match\_code,

NULL match\_status,

xnvp.vlan\_number xng\_vlan\_number, xnvp.vlan\_inst\_id,

xnvp.vlan\_status, aud.CSR\_VENDOR, ncm.description

FROM NCM\_CSR\_VLAN\_WK ncm JOIN NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

ON upper(aud.CSR\_DEVICE\_NAME) = upper(ncm.hostname)

JOIN ci\_vlans xnvp

ON aud.EQUIP\_INST\_ID = xnvp.EQUIP\_INST\_ID

WHERE ncm.port\_type = ''GigEthernet''

and REGEXP\_LIKE (ncm.port\_ip, ''^([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})$'')

AND ncm.VLAN\_NUMBER = xnvp.vlan\_number and

ncm.DEVICE\_IP IS NOT NULL and length(ncm.VLAN\_NUMBER) > 3 and ncm.DEVICE\_IP <> ''0.0.0.0'' and ncm.port\_ip <> ''0.0.0.0'' ),

ne\_only AS

(SELECT aud.CSR\_DEVICE\_NAME, ncm.SUB\_INTERFACE\_NAME, ncm.DEVICE\_IP,

ncm.VLAN\_NUMBER ne\_vlan\_number, ''NE Only'' match\_code,

''Missing in Granite'' match\_status, null,

null xng\_vlan\_number, NULL vlan\_inst\_id, NULL vlan\_status,

aud.CSR\_VENDOR, ncm.description

FROM NCM\_CSR\_VLAN\_WK ncm JOIN NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

ON aud.CSR\_DEVICE\_NAME = ncm.hostname

WHERE ncm.VLAN\_NUMBER IS NOT NULL

AND ncm.port\_type = ''GigEthernet''

and REGEXP\_LIKE (ncm.port\_ip, ''^([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})$'')

AND ncm.DEVICE\_IP IS NOT NULL and length(ncm.VLAN\_NUMBER) > 3 and ncm.DEVICE\_IP <> ''0.0.0.0'' and ncm.port\_ip <> ''0.0.0.0''

AND NOT EXISTS (

SELECT 1

FROM ci\_vlans xnvp

WHERE aud.EQUIP\_INST\_ID = xnvp.EQUIP\_INST\_ID

AND upper(xnvp.csr\_device\_name) = upper(aud.csr\_device\_name)

AND ncm.vlan\_number = xnvp.vlan\_number)),

xng\_only AS

(SELECT aud.csr\_device\_name, NULL SUB\_INTERFACE\_NAME, NULL DEVICE\_IP,

null ne\_vlan\_number, ''Xng Only'' match\_code,

''Missing in NE'' match\_status,

xnvp.vlan\_number xng\_vlan\_number, xnvp.vlan\_inst\_id, xnvp.vlan\_status,

aud.CSR\_VENDOR, Null description

FROM ci\_vlans xnvp JOIN NCM\_HPOV\_EH\_GI\_CI\_CSR\_AUDIT\_WK aud

ON aud.EQUIP\_INST\_ID = xnvp.EQUIP\_INST\_ID

AND AUD.CSR\_DEVICE\_NAME = xnvp.CSR\_DEVICE\_NAME

and not exists(select 1 from NCM\_CSR\_VLAN\_WK ncm

where upper(aud.csr\_device\_name) = upper(ncm.hostname)

AND ncm.VLAN\_NUMBER = xnvp.vlan\_number

and ncm.VLAN\_NUMBER IS NOT NULL and ncm.port\_type = ''GigEthernet'' and length(ncm.VLAN\_NUMBER) > 3 and ncm.DEVICE\_IP <> ''0.0.0.0'' and ncm.port\_ip <> ''0.0.0.0'' and REGEXP\_LIKE (ncm.port\_ip, ''^([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})$'')))

SELECT CSR\_DEVICE\_NAME, CSR\_vendor, CSR\_device\_name, SUB\_INTERFACE\_NAME,

DEVICE\_IP, ne\_vlan\_number, match\_code, match\_status,

xng\_vlan\_number, vlan\_inst\_id, vlan\_status, description

FROM matches

UNION

SELECT CSR\_DEVICE\_NAME, CSR\_vendor, CSR\_device\_name, SUB\_INTERFACE\_NAME,

DEVICE\_IP, ne\_vlan\_number, match\_code, match\_status,

xng\_vlan\_number, vlan\_inst\_id, vlan\_status, description

FROM ne\_only

UNION

SELECT CSR\_DEVICE\_NAME, CSR\_vendor, NULL, SUB\_INTERFACE\_NAME, DEVICE\_IP,

ne\_vlan\_number, match\_code, match\_status, xng\_vlan\_number,

vlan\_inst\_id, vlan\_status, description

FROM xng\_only

';

EXECUTE IMMEDIATE (sqlstmt);

COMMIT;

insert\_ci\_csr\_vlan\_issues ();

EXCEPTION

WHEN OTHERS

THEN

MESSAGE := 'Error: ' || methodname || '(): Can''t audit CI VLANs';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

/\*watchdog.logerror (methodName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

); \*/

RAISE;

END;

END audit\_ci\_csr\_vlans;

PROCEDURE insert\_ci\_csr\_vlan\_issues

IS

methodname VARCHAR2 (30) := 'insert\_CI\_CSR\_vlan\_issues';

MESSAGE VARCHAR2 (300);

sqlstmt VARCHAR2 (32767);

CURSOR dup\_l\_vlan

IS

-- if the same DEVICE\_NAME, nxa.VLAN\_number has > 1 Live status then

-- mark it as dup

WITH vlan\_status AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM ncm\_csr\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

dup\_l\_vlans AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number,

COUNT (vlan\_inst\_id)

FROM vlan\_status nxa

WHERE status = 'Live' AND match\_code <> 'Xng Only'

GROUP BY nxa.csr\_device\_name, nxa.xng\_vlan\_number, status

HAVING COUNT (vlan\_inst\_id) > 1)

SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM dup\_l\_vlans dup JOIN vlan\_status nxa

ON upper(nxa.csr\_device\_name) = upper(dup.csr\_device\_name)

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

ORDER BY nxa.csr\_device\_name, nxa.xng\_vlan\_number;

-- if the same csr\_device\_name, nxa.VLAN\_number has > 1 status then

-- and 1 is Live and other is not then mark NL as 'NE Only' & 'Live path exists'

CURSOR live\_vlan\_exists

IS

WITH vlan\_status AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM ncm\_csr\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

l\_and\_nl\_vlans AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status = 'Live'

INTERSECT

SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status <> 'Live')

SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, nxa.vlan\_inst\_id

FROM l\_and\_nl\_vlans dup JOIN vlan\_status nxa

ON upper(nxa.csr\_device\_name) = upper(dup.csr\_device\_name)

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

WHERE nxa.status <> 'Live'

ORDER BY nxa.csr\_device\_name, nxa.xng\_vlan\_number;

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE ('delete CSR\_VLAN\_AUDIT\_ISSUES\_WK where csr\_vendor = ''CI''');

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t delete rows in table: CSR\_VLAN\_AUDIT\_ISSUES\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

-- mark duplicate in Granite as NE Only, or Live Path Exists

-- if there is a path revision then the same VLAN will be present twice.

-- BUT that is NOT a duplicate

-- if NOT a revision path and VLAN# is present multiple times THEN it is dup

-- mark duplicate in Granite as NE Only, or Live Path Exists

-- else 2 DUP vlans on a csr then Duplicate

sqlstmt :=

'update NCM\_CSR\_VLAN\_AUDIT\_WK

set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

, match\_status = nvl2(match\_status, match\_status|| '', Dup in GI'', ''Dup in GI'')

where vlan\_inst\_id = :p\_vlan\_inst\_id

and XNG\_VLAN\_NUMBER = :p\_xng\_vlan\_number

and upper(CSR\_DEVICE\_NAME) = upper(:p\_csr\_device\_name)

';

FOR rec IN dup\_l\_vlan

LOOP

BEGIN

EXECUTE IMMEDIATE sqlstmt

USING rec.vlan\_inst\_id,

rec.xng\_vlan\_number,

rec.csr\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update ''Dup in GI'' in table: NCM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

BEGIN

sqlstmt :=

'insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

select distinct W.CSR\_DEVICE\_NAME

, case when ne\_vlan\_number is null then xng\_vlan\_number

else ne\_vlan\_number

end vlan\_number,1,''CI''

from NCM\_CSR\_VLAN\_AUDIT\_wk w

where W.MATCH\_STATUS like ''%Dup in GI%''';

EXECUTE IMMEDIATE sqlstmt;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK issue Duplicate in GI ';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

COMMIT;

BEGIN

-- check the ne\_hostname matches MTCE std, else update match\_code = ''NE Only''

-- and match\_status = ''ne\_hostname not MTCE Stds compliant''

sqlstmt :=

' update NCM\_CSR\_VLAN\_AUDIT\_WK aud set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

where exists (select 1

from NCM\_CSR\_VLAN\_WK i

where upper(AUD.CSR\_DEVICE\_NAME) = upper(I.HOSTNAME)

and port\_type = ''GigEthernet''

and length(i.VLAN\_NUMBER) > 3

and i.DEVICE\_IP <> ''0.0.0.0'' and i.port\_ip <> ''0.0.0.0''

AND REGEXP\_LIKE (i.port\_ip, ''^([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})$'')

and parse\_status like ''%11%'')';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

sqlstmt :=

'insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

select distinct W.CSR\_DEVICE\_NAME

, case when ne\_vlan\_number is null then xng\_vlan\_number

else ne\_vlan\_number

end vlan\_number,17,''CI''

from NCM\_CSR\_VLAN\_AUDIT\_wk w, NCM\_CSR\_VLAN\_WK n

where

W.CSR\_VENDOR = ''CI''

and upper(W.CSR\_DEVICE\_NAME) = upper(N.HOSTNAME)

and n.port\_type = ''GigEthernet''

AND REGEXP\_LIKE (n.port\_ip, ''^([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})$'')

and length(n.VLAN\_NUMBER) > 3

and n.DEVICE\_IP <> ''0.0.0.0'' and n.port\_ip <> ''0.0.0.0''

and n.parse\_status like ''%11%''';

--and regexp\_substr(parse\_status, '[[:digit:]]+') = '11'

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update name does not match MTCE Std';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- check if ne\_hostnames' don't match first 6-char match some MTSO CLLI

-- then match\_code = ''NE Only'' and match\_status = ''1st 6-chars of don''''t match any MTSO CLLIs''

sqlstmt :=

'insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.CSR\_device\_name,

SUBSTR (ei.csr\_device\_name, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM NCM\_CSR\_VLAN\_AUDIT\_wk ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.csr\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.csr\_device\_name, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.csr\_device\_name

FROM NCM\_CSR\_VLAN\_AUDIT\_wk ei

MINUS

SELECT ei.csr\_device\_name

FROM NCM\_CSR\_VLAN\_AUDIT\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.csr\_device\_name, 1, 8)

MINUS

SELECT ei.csr\_device\_name

FROM NCM\_CSR\_VLAN\_AUDIT\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.csr\_device\_name, 1, 6)

),

ISSUES AS

(

SELECT cm.csr\_device\_name, 22 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.csr\_device\_name, 2 issue\_id

FROM bad\_clli bc

)

SELECT distinct i.csr\_device\_name, W.NE\_VLAN\_NUMBER, i.issue\_id, ''CI''

FROM ISSUES i, NCM\_CSR\_VLAN\_AUDIT\_wk w

where upper(i.csr\_device\_name) = upper(W.CSR\_DEVICE\_NAME)

and W.NE\_VLAN\_NUMBER is not null

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

NULL;

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t mark 1st 6-char don''t match MTSO CLLIs';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in sam vlan extract

sqlstmt :=

' insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

select distinct aud.CSR\_DEVICE\_NAME, xng\_vlan\_number, 13, ''CI''

from NCM\_CSR\_VLAN\_AUDIT\_wk aud

where aud.ne\_hostname is null';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant insert issue ''Not in NCM VLAN Extract''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in GI

sqlstmt :=

' insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

select distinct aud.CSR\_DEVICE\_NAME, ne\_vlan\_number, 15, ''CI''

from NCM\_CSR\_VLAN\_AUDIT\_wk aud

where aud.vlan\_inst\_id is null';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname

|| '(): Cant insert issue ''Not in GI''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- duplicate in NE

sqlstmt :=

' insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

SELECT csr\_device\_name, ne\_vlan\_number, 10, ''CI''

FROM ncm\_csr\_vlan\_audits

WHERE ne\_vlan\_number > 0

GROUP BY csr\_device\_name, ne\_vlan\_number

HAVING COUNT (\*) > 1';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant insert issue ''Duplicate in NE''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

-- if there is a path revision then the same VLAN will be present twice.

-- BUT that is NOT a duplicate

-- if NOT a revision path and VLAN# is present multiple times THEN it is dup

-- if 1 == Live and the other <> Live then mark <> Live as 'Live in Xng'

sqlstmt :=

'update NCM\_CSR\_VLAN\_AUDIT\_WK

set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

, match\_status = nvl2(match\_status, ''Live Path Exists,'' ||match\_status , ''Live Path Exists'')

where vlan\_inst\_id = :p\_vlan\_inst\_id

and XNG\_VLAN\_NUMBER = :p\_xng\_vlan\_number

and upper(CSR\_DEVICE\_NAME) = upper(:p\_csr\_device\_name)

--and csr\_vendor = :p\_csr\_vendor

';

FOR rec IN live\_vlan\_exists

LOOP

BEGIN

EXECUTE IMMEDIATE sqlstmt

USING rec.vlan\_inst\_id,

rec.xng\_vlan\_number,

rec.csr\_device\_name; --, p\_csr\_vendor;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update ''Live Path Exists'' in table: NCM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

BEGIN

-- IF BOTH and has issues then mark it NE

sqlstmt :=

' update NCM\_CSR\_VLAN\_AUDIT\_WK aud

set match\_code = ''NE Only''

where exists (select 1

from

CSR\_VLAN\_AUDIT\_ISSUEs\_wk vi,

CSR\_issue n

where aud.MATCH\_CODE = ''BOTH''

and aud.csr\_vendor = ''CI''

and VI.CSR\_VENDOR=''CI''

and upper(AUD.CSR\_DEVICE\_NAME) = upper(VI.CSR\_DEVICE\_NAME)

and AUD.NE\_VLAN\_NUMBER = VI.VLAN\_NUMBER

and N.IS\_CRITICAL=''Y''

and N.CSR\_ISSUE\_ID = vi.CSR\_ISSUE\_ID

)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant update match\_status to ''NE Only''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in NCM vlan extract

sqlstmt :=

' insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

WITH vlan\_status AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, status, match\_code,

nxa.vlan\_inst\_id

FROM NCM\_csr\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

l\_and\_nl\_vlans AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status = ''Live''

INTERSECT

SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status <> ''Live''),

live\_vlan\_exists AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, nxa.vlan\_inst\_id

FROM l\_and\_nl\_vlans dup JOIN vlan\_status nxa

ON upper(nxa.csr\_device\_name) = upper(dup.csr\_device\_name)

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

WHERE nxa.status <> ''Live'')

SELECT DISTINCT aud.csr\_device\_name, xng\_vlan\_number, 31, ''CI''

FROM NCM\_csr\_vlan\_audit\_wk aud

WHERE vlan\_inst\_id IS NOT NULL

AND match\_code = ''NE Only''

AND NOT EXISTS (

SELECT 1

FROM csr\_vlan\_audit\_issues\_wk nvai

WHERE nvai.vlan\_number = aud.xng\_vlan\_number

AND upper(aud.csr\_device\_name) = upper(nvai.csr\_device\_name)

AND nvai.csr\_vendor = ''CI'')

AND EXISTS (

SELECT 1

FROM live\_vlan\_exists lle

WHERE upper(lle.csr\_device\_name) = upper(aud.csr\_device\_name)

AND aud.xng\_vlan\_number = lle.xng\_vlan\_number)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant insert issue ''Live VLAN Exists''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

END insert\_ci\_csr\_vlan\_issues;

PROCEDURE assign\_area\_region\_to\_csr\_vlan

IS

methodname VARCHAR2 (30) := 'assign\_area\_region\_to\_csr\_vlan';

MESSAGE VARCHAR2 (300);

updstmt VARCHAR2 (32767);

processname VARCHAR2 (30) := 'NCM\_GI\_CSR\_VLAN\_AUDIT';

CURSOR area\_region\_dev

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS')

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

csr\_device\_name, clli\_6

FROM ncm\_csr\_vlan\_audit\_wk aud JOIN domains di

ON di.clli\_6 = SUBSTR (aud.csr\_device\_name, 1, 6)

;

BEGIN

updstmt :=

'update NCM\_CSR\_VLAN\_AUDIT\_WK set area = :area

, region = :region, market = :market

, leaf\_domain\_inst\_id = :leaf\_domain\_inst\_id

, clli = :clli\_6

where csr\_device\_name = :csr\_device\_name';

FOR rec IN area\_region\_dev

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli\_6,

rec.csr\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update clli for '

|| rec.csr\_device\_name

|| ' in table: NCM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (methodname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

BEGIN

-- if there still are some devices which cant be mapped to any domain then mark those as unknown

updstmt :=

' update NCM\_CSR\_VLAN\_AUDIT\_WK

set area = ''unknown''

, region = ''unknown''

, market = ''unknown''

, leaf\_domain\_inst\_id = 0

, clli = SUBSTR (csr\_device\_name, 1, 6)

where AREA is null ';

EXECUTE IMMEDIATE updstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from to unknown in table: NCM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END assign\_area\_region\_to\_csr\_vlan;

PROCEDURE load\_vlan\_csr\_clli\_summary

IS

methodname VARCHAR2 (30) := 'load\_vlan\_csr\_clli\_summary';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

EXECUTE IMMEDIATE 'truncate table NCM\_CSR\_VLAN\_CLLI\_SUMM\_WK';

-- generate device summary

sqlstmt :=

' insert into NCM\_CSR\_VLAN\_CLLI\_SUMM\_WK

columns (area, region, market

, clli

, ne\_total\_csr\_vlan

, matched\_l\_csr\_vlan

, matched\_nl\_csr\_vlan

, mismatched\_csr\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_COMPLIANCE)

WITH

csr as

(

select distinct area

, region, market, leaf\_domain\_inst\_id, clli

, csr\_device\_name, csr\_vendor

from NCM\_CSR\_VLAN\_AUDIT\_WK ng

where ng.CSR\_DEVICE\_NAME is not null

),

csr\_VLAN as

(

select distinct hostname CSR\_DEVICE\_NAME, VLAN\_NUMBER

from NCM\_CSR\_VLAN\_WK ng

where ng.VLAN\_NUMBER > 0 and port\_type = ''GigEthernet''

and REGEXP\_LIKE (ng.port\_ip, ''^([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})$'')

and length(ng.VLAN\_NUMBER) > 3

and ng.DEVICE\_IP <> ''0.0.0.0'' and ng.port\_ip <> ''0.0.0.0''

)

,

CNT\_csr\_VLANs as

(

select distinct ng.CSR\_DEVICE\_NAME, count(vlan\_number) total\_ne\_csr\_vlan

from csr\_VLAN nv

join csr ng

on upper(ng.CSR\_DEVICE\_NAME) = upper(nv.CSR\_DEVICE\_NAME)

group by ng.CSR\_DEVICE\_NAME

),

live\_match as

(

select distinct aud.CSR\_DEVICE\_NAME, aud.ne\_vlan\_number, vlan\_status

from NCM\_CSR\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.csr\_vendor = ''CI''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status = ''Live''

)

,nonlive\_match as

(

select distinct aud.csr\_DEVICE\_NAME, aud.ne\_vlan\_number

from NCM\_CSR\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.csr\_vendor = ''CI''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status <> ''Live''

minus

select distinct aud.csr\_DEVICE\_NAME, aud.ne\_vlan\_number

from NCM\_CSR\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.csr\_vendor = ''CI''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status = ''Live''

) ,

CNT\_BOTH\_LIVE\_VLANs as

( -- good matches

select csr\_DEVICE\_NAME, nvl (count(csr\_DEVICE\_NAME), 0) matched\_l\_csr\_vlan

from live\_match aud

group by csr\_DEVICE\_NAME

),

CNT\_BOTH\_NONLIVE\_VLANs as

( -- good matches

select csr\_DEVICE\_NAME, nvl( count(csr\_DEVICE\_NAME), 0) matched\_nl\_csr\_vlan

-- , nvl(sum(case when vlan\_status = ''Live'' then 1 else 0 end), 0) matched\_l\_csr\_vlan

-- , nvl(sum(case when vlan\_status <> ''Live'' then 1 else 0 end), 0) matched\_nl\_csr\_vlan

from nonlive\_match aud

group by aud.csr\_DEVICE\_NAME

) ,

MISMATCH\_VLANs as

( -- bad matches

select distinct aud.csr\_DEVICE\_NAME, aud.ne\_vlan\_number, aud.xng\_vlan\_number, vlan\_status

from NCM\_CSR\_VLAN\_AUDIT\_WK aud

where match\_code <> ''BOTH''

and aud.csr\_vendor = ''CI''

and (aud.ne\_vlan\_number > 0 or aud.xng\_vlan\_number > 0)

AND NOT EXISTS (

SELECT 1

FROM csr\_vlan\_audit\_issues\_wk nvai

WHERE nvai.vlan\_number = aud.xng\_vlan\_number

AND upper(aud.csr\_device\_name) = upper(nvai.csr\_device\_name)

AND nvai.csr\_vendor = ''CI'' and nvai.CSR\_ISSUE\_ID = 31)

)

,

CNT\_MISMATCH\_VLANs as

( -- bad match cnts

select aud.CSR\_DEVICE\_NAME

, count(1) mismatch\_cnt

from MISMATCH\_VLANs aud

group by aud.csr\_DEVICE\_NAME

)

select area, region, market,clli

, sum(nvl(nv.total\_ne\_csr\_vlan, 0)) ne\_total\_csr\_vlan

, sum(nvl(matched\_l\_csr\_vlan, 0)) matched\_l\_csr\_vlan

, sum(nvl(matched\_nl\_csr\_vlan, 0)) matched\_nl\_csr\_vlan

, sum(nvl(mismatch\_cnt, 0)) mismatched\_csr\_vlan

, round(sum(NVL(matched\_nl\_csr\_vlan, 0)) / decode(sum(NVL(matched\_l\_csr\_vlan, 0)), 0, 1, sum(nvl(matched\_l\_csr\_vlan, 0)))

\* 100, 2) gi\_nl\_2\_total\_vlan\_pct

, round (sum(nvl(matched\_l\_csr\_vlan,0)) / decode (sum(nvl( nv.total\_ne\_csr\_vlan, 0)) - sum(nvl(matched\_nl\_csr\_vlan, 0)), 0 , 1

, sum(nvl(nv.total\_ne\_csr\_vlan,0)) - sum(nvl(matched\_nl\_csr\_vlan,0)))

\* 100 , 2) vlan\_COMPLIANCE

from csr ng

left outer join CNT\_csr\_VLANs nv

on upper(nv.csr\_DEVICE\_NAME) = upper(ng.csr\_DEVICE\_NAME)

left outer join CNT\_BOTH\_LIVE\_VLANs lmv

on upper(lmv.csr\_DEVICE\_NAME) = upper(ng.csr\_DEVICE\_NAME)

left outer join CNT\_BOTH\_NONLIVE\_VLANs nlmv

on upper(nlmv.csr\_DEVICE\_NAME) = upper(ng.csr\_DEVICE\_NAME)

left outer join CNT\_MISMATCH\_VLANs mv

on upper(mv.csr\_DEVICE\_NAME) = upper(ng.csr\_DEVICE\_NAME)

group by ng.area, ng.region, ng.market, csr\_Vendor, clli';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname || '(): Can''t generate clli summary';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END load\_vlan\_csr\_clli\_summary;

PROCEDURE load\_csr\_vlan\_regional\_summ

IS

methodname VARCHAR2 (30) := 'load\_vlan\_regional\_summary';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

EXECUTE IMMEDIATE 'truncate table NCM\_csr\_VLAN\_REGION\_SUMM\_WK';

-- generate device summary

sqlstmt :=

' insert into NCM\_CSR\_VLAN\_REGION\_SUMM\_WK

columns (area, region

, ne\_total\_csr\_vlan

, matched\_l\_csr\_vlan

, matched\_nl\_csr\_vlan

, mismatched\_csr\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_compliance)

select dlr.area, dlr.region

, sum(ne\_total\_csr\_vlan) ne\_total\_csr\_vlan

, sum(matched\_l\_csr\_vlan) matched\_l\_csr\_vlan

, sum(matched\_nl\_csr\_vlan) matched\_nl\_csr\_vlan

, sum(mismatched\_csr\_vlan) mismatched\_csr\_vlan

, round(avg(gi\_nl\_2\_total\_vlan\_pct), 2) gi\_nl\_2\_total\_vlan\_pct

, round(avg(vlan\_COMPLIANCE), 2) vlan\_compliance

from domains\_leaf\_reporting dlr

left outer join NCM\_CSR\_VLAN\_CLLI\_SUMM\_WK ng

on dlr.market = ng.market

where dlr.AREA not in (''NNO'', ''OSS'')

group by rollup (dlr.area, dlr.region)

ORDER BY AREA, REGION ';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname || '(): Can''t generate regional summary';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END load\_csr\_vlan\_regional\_summ;

END;

/

--------------------------------------------------------

-- DDL for Package Body NCM\_GI\_NGMLS\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."NCM\_GI\_NGMLS\_AUDIT"

IS

--

-- To modify this template, edit file PKGBODY.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package body

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter procedure, function bodies as shown below

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure audit\_ngMLS\_equip \*

\* \*

\* Purpose: Audit ngMLS Equipment against Granite Inventory \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE audit\_ngMLS\_equip IS

methodName varchar2(30) := 'audit\_ngMLS\_equip';

message varchar2(500);

BEGIN

dbms\_output.PUT\_LINE('FYI: In '||methodName);

audit\_CI\_ngMLS\_equip();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr(message||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure auditCIngMLS \*

\* \*

\* Purpose: this method compares Xng data to discovered ncm data \*

\* and records only the matched equip \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure audit\_CI\_ngMLS\_equip is

methodName varchar2(30) := 'audit\_CI\_ngMLS\_equip';

message varchar2(500);

sqlStmt varchar2(32767);

BEGIN

-- verified with shirley --- assumption is they wont embed all ngMLSs within a contianer of type ngMLS. This logic will break if they so that.

-- does it matter if the the equipment name in GI does not match Std... still need feedback from Shirley ??????????????

dbms\_output.put\_line('FYI: In '||methodName);

begin

delete from NCM\_GI\_CI\_NGMLS\_AUDIT\_WK nxa

where ngMLS\_vendor = 'CI';

delete from ngMLS\_device\_AUDIT\_issues\_wk iss

where ngMLS\_vendor = 'CI';

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t delete records from table: NCM\_GI\_CI\_NGMLS\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

sqlStmt := 'insert into ncm\_gi\_ci\_ngMLS\_audit\_wk

columns (ngMLS\_vendor, ngMLS\_device\_name

, ncm\_ngMLS\_hostname, ncm\_ngMLS\_partition, ncm\_ngMLS\_device\_ip

, ncm\_ngMLS\_vlan\_hostname

, gi\_device\_name, ne\_inst\_id

, Live\_in\_xng, eq\_status

, match\_code

, match\_status

)

WITH

NCM\_ngMLSs as

(

select distinct hostname, ncm.partition, ncm.device\_ip ncm\_ngMLS\_device\_ip

from ncm\_ngMLS\_wk ncm

)

,

NCM\_ngVLAN as

(

select distinct hostname, ncm.partition, ncm.device\_ip ncm\_ngMLS\_vlan\_device\_ip

from ncm\_ngMLS\_vlan\_wk ncm

where vlan\_number is not null

)

,

Xng as

(

select ngMLS.\*

from xng\_ngMLS\_equip\_wk ngMLS

where ngMLS\_vendor = ''CI''

and ngMLS.eq\_class\_type = ''NE''

)

,

NCM\_ngMLSS\_ngVLAN\_FULL\_JOIN as

(

select case when ng.hostname is not null then ng.hostname

when nv.hostname is not null then nv.hostname

end ngMLS\_device\_name

, ng.hostname ncm\_ngMLS\_hostname, ng.partition ncm\_ngMLS\_partition, ng.ncm\_ngMLS\_device\_ip

, nv.hostname ncm\_ngMLS\_vlan\_hostname --, ncm\_ngMLS\_vlan\_partition, ncm\_ngMLS\_vlan\_device\_ip

, case when ng.hostname is not null

and nv.hostname is not null

then ''BOTH''

when ng.hostname is not null then ''NE Only''

when nv.hostname is not null then ''NE Only''

else ''xxNE Only''

end match\_code

, case when ng.hostname is not null

and nv.hostname is not null

then null

when ng.hostname is null then ''12'' --nvl2(match\_status, ''12, ''||match\_status, ''12'')

when nv.hostname is null then ''13'' -- nvl2(match\_status, ''14, 13, ''||match\_status, ''14, 13'')

else ''What is this??'' --null -- when does this condition appear

end match\_status

from NCM\_ngVLAN nv

full outer join NCM\_ngMLSs ng

on ng.hostname = nv.hostname

)

select distinct ''CI'' ngMLS\_vendor,

case when nfj.ngMLS\_device\_name is not null then nfj.ngMLS\_device\_name

when xng.ngMLS\_device\_name is not null then xng.ngMLS\_device\_name

else ''zzzz''||xng.descr

end ngMLS\_device\_name

, ncm\_ngMLS\_hostname, ncm\_ngMLS\_partition, ncm\_ngMLS\_device\_ip

, ncm\_ngMLS\_vlan\_hostname --, ncm\_ngMLS\_vlan\_partition, ncm\_ngMLS\_vlan\_device\_ip

, xng.descr gi\_device\_name, equip\_inst\_id

, case when status = ''Live'' then ''Y''

when status is not null then ''N''

end Live\_in\_xng

, status

, case when nfj.ngMLS\_device\_name is not null

and xng.ne\_inst\_id is not null

and match\_code = ''BOTH''

then ''BOTH''

-- it is in both NCM extracts but not in Xng then NE Only

when nfj.ngMLS\_device\_name is not null

and xng.ne\_inst\_id is null

and match\_code = ''BOTH''

then ''NE Only''

when nfj.ngMLS\_device\_name is null and xng.ne\_inst\_id is not null then ''Xng Only''

else match\_code -- when the device is not found in both NCM extracts

end match\_code

-- , case when xng.ne\_inst\_id is null then nvl2(match\_status, ''15, ''||match\_status, ''15'')

-- else nvl2(parse\_status, parse\_status || '', '' ||match\_status, match\_status) --null -- when does this condition appear

-- end match\_status

, parse\_status match\_status

from NCM\_ngMLSS\_ngVLAN\_FULL\_JOIN nfj

full outer join Xng

on xng.descr like ''%''||nfj.ngMLS\_device\_name ||''%''

';

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert insert CI audit into table: NCM\_GI\_CI\_NGMLS\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

BEGIN

insert\_CI\_ngMLS\_equip\_issues();

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t record CI device errors';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

END;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure insert\_CI\_ngMLS\_equip\_issues \*

\* \*

\* Purpose: if it is a match then check if there are a few problems \*

\* duplicate equipment in Granite \*

\* NE name is not standard \*

\* NE name does not match any MTSO CLLI \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure insert\_CI\_ngMLS\_equip\_issues is

methodName varchar2(30) := 'insert\_CI\_ngMLS\_equip\_issues';

message varchar2(500);

sqlStmt varchar2(32767);

BEGIN

dbms\_output.PUT\_LINE('FYI: In '||methodName);

begin

-- if some ne\_hostname matched > 1 granite ne\_inst\_id then mark it

--Duplicate in NE

sqlStmt := 'insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select hostname, 10, ''CI''

from NCM\_ngMLS\_wk i

group by i.hostname

having count(device\_ip) > 1';

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK issue Duplicate in NE ';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

BEGIN

-- if some ne\_hostname matched > 1 granite ne\_inst\_id then mark it

--Duplicate in GI

sqlStmt := ' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_wk

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 1, ''CI''

from NCM\_GI\_CI\_NGMLS\_AUDIT\_WK i

where match\_code <> ''Xng Only''

group by i.ngMLS\_device\_name

having count(1) > 1'

;

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t mark Duplicate in Granite';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

-- check the ne\_hostname matches MTCE std, else update match\_code = ''NE Only''

-- and match\_status = ''ne\_hostname not MTCE Stds compliant''

sqlStmt:= ' update ncm\_gi\_ci\_ngMLS\_audit\_wk aud set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

where exists (select 1

from ncm\_ngmls\_wk i

where ngmls\_vendor = ''CI''

and AUD.NCM\_NGMLS\_HOSTNAME = I.HOSTNAME

and parse\_status like ''%11%''

)'

;

EXECUTE IMMEDIATE sqlStmt;

commit;

sqlStmt:= ' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 11, ngMLS\_vendor

from ncm\_ngmls\_wk n, ncm\_gi\_ci\_ngmls\_audit\_wk w

where

W.NGMLS\_VENDOR = ''CI''

and W.NCM\_NGMLS\_HOSTNAME = N.HOSTNAME

and n.parse\_status like ''%11%'''

;

--and regexp\_substr(parse\_status, '[[:digit:]]+') = '11'

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t update name does not match MTCE Std';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

dbms\_output.put\_line('FYI: Checking GI Data Std');

sqlStmt:= ' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 7, ngMLS\_vendor

from xng\_ngmls\_equip\_wk

where ngmls\_vendor = ''CI''

and eq\_class\_type = ''NE''

and parse\_status like ''%7%'''

;

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t update eq name does not match GI Std';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

-- check if ne\_hostnames' don't match first 6-char match some MTSO CLLI

-- then match\_code = ''NE Only'' and match\_status = ''1st 6-chars of don''''t match any MTSO CLLIs''

sqlStmt := 'insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.ngMLS\_device\_name,

SUBSTR (ei.ngMLS\_device\_name, 1, 8) ngMLS\_clli, cdm6.clli\_6 vsm\_clli

FROM NCM\_GI\_CI\_NGMLS\_AUDIT\_WK ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.ngMLS\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.ngMLS\_device\_name, 1, 8) = cdm.clli)),

BAD\_CLLI AS

(

SELECT ei.ngMLS\_device\_name

FROM NCM\_GI\_CI\_NGMLS\_AUDIT\_WK ei

MINUS

SELECT ei.ngMLS\_device\_name

FROM NCM\_GI\_CI\_NGMLS\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.ngMLS\_device\_name, 1, 8)

MINUS

SELECT ei.ngMLS\_device\_name

FROM NCM\_GI\_CI\_NGMLS\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.ngMLS\_device\_name, 1, 6)

),

ISSUES AS

(

SELECT cm.ngMLS\_device\_name, 3 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.ngMLS\_device\_name, 2 issue\_id

FROM bad\_clli bc

)

SELECT i.ngMLS\_device\_name, i.issue\_id, ''CI''

FROM ISSUES i

';

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

null;

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t mark 1st 6-char don''t match MTSO CLLIs';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

-- device not found in ncm vlan extract

sqlStmt := ' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 13, ''CI''

from NCM\_GI\_CI\_NGMLS\_AUDIT\_WK aud

where ncm\_ngMLS\_vlan\_hostname is null'

;

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Cant insert issue ''Not in NCM VLAN Extract''';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

-- device not found in GI

sqlStmt := ' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 15, ''CI''

from NCM\_GI\_CI\_NGMLS\_AUDIT\_WK aud

where aud.ne\_inst\_id is null'

;

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Cant insert issue ''Not in GI''';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- add more errors for not parseable etc????????????????????

-- IF BOTH and has issues then mark it NE

sqlStmt := ' update ncm\_gi\_ci\_ngMLS\_audit\_wk aud

set match\_code = ''NE Only''

where exists (select 1

from ngMLS\_device\_AUDIT\_ISSUEs\_wk cdai, ngmls\_issue n

where aud.MATCH\_CODE = ''BOTH''

and aud.ngMLS\_vendor = ''CI''

and N.IS\_CRITICAL=''Y''

and N.NGMLS\_ISSUE\_ID = CDAI.NGMLS\_ISSUE\_ID

and AUD.NGMLS\_DEVICE\_NAME = CDAI.NGMLS\_DEVICE\_NAME

)';

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Cant update match\_status to ''NE Only'' issue ';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END insert\_CI\_ngMLS\_equip\_issues;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* audit VLAN per ngMLS \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE audit\_ngMLS\_vlans IS

methodName varchar2(30) := 'audit\_ngMLS\_vlans';

message varchar2(500);

BEGIN

dbms\_output.PUT\_LINE('FYI: In '||methodName);

BEGIN

audit\_CI\_ngMLS\_vlans();

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t audit VLANs';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

END audit\_ngMLS\_vlans;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* audit CI VLAN per ngMLS \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE audit\_CI\_ngMLS\_vlans IS

methodName varchar2(30) := 'audit\_CI\_ngMLS\_vlans';

message varchar2(500);

sqlStmt varchar2(32767);

BEGIN

dbms\_output.PUT\_LINE('FYI: In '||methodName);

BEGIN

execute immediate ('delete NCM\_NGMLS\_VLAN\_AUDIT\_WK where ngMLS\_vendor = ''CI''');

commit;

sqlStmt := 'insert into NCM\_NGMLS\_VLAN\_AUDIT\_WK columns (ngMLS\_Device\_Name, ngMLS\_vendor

, ne\_hostName, ne\_sub\_interface\_name, ne\_port\_ip, ne\_vlan\_number

, match\_code, match\_status

, NE\_INST\_ID

, xng\_vlan\_number

, vlan\_inst\_id

, vlan\_status

)

WITH

CI\_VLANs as

(

select xnvp.\*

from XNG\_NGMLS\_VLAN\_PATHS\_wk xnvp

join XNG\_NGMLS\_EQUIP\_wk xne

on xne.NE\_INST\_ID = xnvp.NE\_INST\_ID

where xne.ngMLS\_vendor = ''CI''

and xnvp.vlan\_number is not null

)

,

MATCHES as

(

select aud.ngMLS\_Device\_Name, ncm.VLAN\_NUMBER ne\_vlan\_number

, sub\_interface\_name, port\_ip

, ''BOTH'' match\_code

, null match\_status

, xnvp.NE\_INST\_ID

, xnvp.vlan\_number xng\_vlan\_number

, xnvp.vlan\_inst\_id

, xnvp.vlan\_status

, aud.ngMLS\_vendor

from ncm\_ngmls\_vlan\_wk ncm

join ncm\_gi\_ci\_ngMLS\_audit\_wk aud

on aud.ngMLS\_Device\_Name = ncm.hostname

join CI\_VLANs xnvp

on aud.NE\_INST\_ID = xnvp.NE\_INST\_ID

and ncm.VLAN\_NUMBER = xnvp.vlan\_number

where ncm.port\_ip is not null

)

,

NE\_ONLY as

(

select aud.ngMLS\_Device\_Name

, ncm.sub\_interface\_name, ncm.port\_ip

, ncm.VLAN\_NUMBER ne\_vlan\_number, ''NE Only'' match\_code

, ''Missing in Granite'' match\_status

, xnvp.NE\_INST\_ID

, xnvp.vlan\_number xng\_vlan\_number

, xnvp.vlan\_inst\_id

, null vlan\_status

, aud.ngMLS\_vendor

from ncm\_ngmls\_vlan\_wk ncm

join ncm\_gi\_ci\_ngMLS\_audit\_wk aud

on aud.ngMLS\_Device\_Name = ncm.hostname

left outer join CI\_VLANs xnvp

on aud.NE\_INST\_ID = xnvp.NE\_INST\_ID

and ncm.VLAN\_NUMBER = xnvp.vlan\_number

where xnvp.vlan\_number is null

and ncm.VLAN\_NUMBER is not null

and ncm.port\_ip is not null

)

,

XNG\_ONLY as

(

select aud.ngMLS\_Device\_Name

, null sub\_interface\_name, null port\_ip

, ncm.VLAN\_NUMBER ne\_vlan\_number, ''Xng Only'' match\_code

, ''Missing in NE'' match\_status

, xnvp.NE\_INST\_ID

, xnvp.vlan\_number xng\_vlan\_number

, xnvp.vlan\_inst\_id

, xnvp.vlan\_status

, aud.ngMLS\_vendor

from CI\_VLANs xnvp

join ncm\_gi\_ci\_ngMLS\_audit\_wk aud

on aud.ne\_inst\_id = xnvp.ne\_inst\_id

left outer join ncm\_ngmls\_vlan\_wk ncm

on aud.ngMLS\_Device\_Name = ncm.hostname

and ncm.VLAN\_NUMBER = xnvp.vlan\_number

where ncm.vlan\_number is null

)

select ngMLS\_Device\_Name, ngMLS\_vendor

, ngMLS\_Device\_Name, sub\_interface\_name, port\_ip, ne\_vlan\_number

, match\_code, match\_status

, NE\_INST\_ID

, xng\_vlan\_number, vlan\_inst\_id, vlan\_status

from MATCHES

union

select ngMLS\_Device\_Name, ngMLS\_vendor

, ngMLS\_Device\_Name, sub\_interface\_name, port\_ip, ne\_vlan\_number

, match\_code, match\_status

, NE\_INST\_ID

, xng\_vlan\_number, vlan\_inst\_id, vlan\_status

from NE\_ONLY

union

select ngMLS\_Device\_Name, ngMLS\_vendor

, null, sub\_interface\_name, port\_ip, ne\_vlan\_number

, match\_code, match\_status

, NE\_INST\_ID

, xng\_vlan\_number, vlan\_inst\_id, vlan\_status

from Xng\_ONLY

';

execute immediate (sqlStmt);

commit;

insert\_CI\_ngMLS\_vlan\_issues();

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t audit CI VLANs';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

/\*watchdog.logerror (methodName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

); \*/

RAISE;

END;

END audit\_CI\_ngMLS\_vlans;

/\*

if there are any issues with the VLANs then record the errors in here

put them in a table. That way when compelte EBH has to use these, it will be easy.

\*/

procedure insert\_CI\_ngMLS\_vlan\_issues is

methodName varchar2(30) := 'insert\_CI\_ngMLS\_vlan\_issues';

message varchar2(300);

sqlStmt varchar2(32767);

CURSOR DUP\_L\_VLAN is

-- if the same DEVICE\_NAME, nxa.VLAN\_number has > 1 Live status then

-- mark it as dup

WITH

VLAN\_STATUS as

(

select nxa.ngMLS\_DEVICE\_NAME, nxa.xng\_vlan\_number, status, match\_code

, nxa.vlan\_inst\_id

from NCM\_NGMLS\_VLAN\_AUDIT\_WK nxa

join vzwnet.circ\_path\_inst cpi

on cpi.CIRC\_PATH\_INST\_ID = nxa.VLAN\_INST\_ID

)

,

DUP\_L\_VLANS as

(

select nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, count (vlan\_inst\_id)

from VLAN\_STATUS nxa

where status = 'Live'

and match\_code <> 'Xng Only'

group by nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, status

having count(vlan\_inst\_id) > 1

)

SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, status, match\_code

, nxa.vlan\_inst\_id

from DUP\_L\_VLANS dup

join VLAN\_STATUS nxa

on nxa.ngmls\_device\_name = dup.ngmls\_device\_name

and nxa.xng\_vlan\_number = dup.xng\_vlan\_number

order by nxa.ngmls\_device\_name, nxa.xng\_vlan\_number

;

-- if the same ngmls\_device\_name, nxa.VLAN\_number has > 1 status then

-- and 1 is Live and other is not then mark NL as 'NE Only' & 'Live path exists'

CURSOR LIVE\_VLAN\_EXISTS is

WITH

VLAN\_STATUS as

(

select nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, status, match\_code

, nxa.vlan\_inst\_id

from NCM\_NGMLS\_VLAN\_AUDIT\_WK nxa

join vzwnet.circ\_path\_inst cpi

on cpi.CIRC\_PATH\_INST\_ID = nxa.VLAN\_INST\_ID

)

,

L\_and\_NL\_VLANS as

(

select nxa.ngmls\_device\_name, nxa.xng\_vlan\_number

from VLAN\_STATUS nxa

where status = 'Live'

intersect

select nxa.ngmls\_device\_name, nxa.xng\_vlan\_number

from VLAN\_STATUS nxa

where status <> 'Live'

)

SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, nxa.vlan\_inst\_id

from L\_and\_NL\_VLANS dup

join VLAN\_STATUS nxa

on nxa.ngmls\_device\_name = dup.ngmls\_device\_name

and nxa.xng\_vlan\_number = dup.xng\_vlan\_number

where nxa.status <> 'Live'

order by nxa.ngmls\_device\_name, nxa.xng\_vlan\_number

;

BEGIN

dbms\_output.PUT\_LINE('FYI: In '||methodName);

begin

execute immediate ('delete NGMLS\_VLAN\_AUDIT\_ISSUES\_WK where ngMLS\_vendor = ''CI''');

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t delete rows in table: NGMLS\_VLAN\_AUDIT\_ISSUES\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

-- mark duplicate in Granite as NE Only, or Live Path Exists

-- if there is a path revision then the same VLAN will be present twice.

-- BUT that is NOT a duplicate

-- if NOT a revision path and VLAN# is present multiple times THEN it is dup

-- mark duplicate in Granite as NE Only, or Live Path Exists

-- else 2 DUP vlans on a csr then Duplicate

sqlStmt := 'update NCM\_NGMLS\_VLAN\_AUDIT\_WK

set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

, match\_status = nvl2(match\_status, match\_status|| '', Dup in GI'', ''Dup in GI'')

where vlan\_inst\_id = :p\_vlan\_inst\_id

';

for rec in DUP\_L\_VLAN

loop

BEGIN

execute immediate sqlStmt using rec.vlan\_inst\_id;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update ''Dup in GI'' in table: NCM\_NGMLS\_VLAN\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

end loop;

commit;

begin

sqlStmt := 'insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct W.NGMLS\_DEVICE\_NAME

, case when ne\_vlan\_number is null then xng\_vlan\_number

else ne\_vlan\_number

end vlan\_number,1,''CI''

from NCM\_NGMLS\_VLAN\_AUDIT\_wk w

where W.MATCH\_STATUS like ''%Dup in GI%''';

EXECUTE IMMEDIATE sqlStmt;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK issue Duplicate in GI ';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

commit;

BEGIN

-- check the ne\_hostname matches MTCE std, else update match\_code = ''NE Only''

-- and match\_status = ''ne\_hostname not MTCE Stds compliant''

sqlStmt:= ' update NCM\_NGMLS\_VLAN\_AUDIT\_WK aud set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

where exists (select 1

from ncm\_ngmls\_vlan\_wk i

where ngmls\_vendor = ''CI''

and AUD.NGMLS\_DEVICE\_NAME = I.HOSTNAME

and parse\_status like ''%11%''

)'

;

EXECUTE IMMEDIATE sqlStmt;

commit;

sqlStmt := 'insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct W.NGMLS\_DEVICE\_NAME

, case when ne\_vlan\_number is null then xng\_vlan\_number

else ne\_vlan\_number

end vlan\_number,17,''CI''

from NCM\_NGMLS\_VLAN\_AUDIT\_wk w, ncm\_ngmls\_vlan\_wk n

where

W.NGMLS\_VENDOR = ''CI''

and W.NGMLS\_DEVICE\_NAME = N.HOSTNAME

and n.parse\_status like ''%11%'''

;

--and regexp\_substr(parse\_status, '[[:digit:]]+') = '11'

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t update name does not match MTCE Std';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

-- check if ne\_hostnames' don't match first 6-char match some MTSO CLLI

-- then match\_code = ''NE Only'' and match\_status = ''1st 6-chars of don''''t match any MTSO CLLIs''

sqlStmt := 'insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.ngMLS\_device\_name,

SUBSTR (ei.ngMLS\_device\_name, 1, 8) ngMLS\_clli, cdm6.clli\_6 vsm\_clli

FROM NCM\_NGMLS\_VLAN\_AUDIT\_wk ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.ngMLS\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.ngMLS\_device\_name, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.ngMLS\_device\_name

FROM NCM\_NGMLS\_VLAN\_AUDIT\_wk ei

MINUS

SELECT ei.ngMLS\_device\_name

FROM NCM\_NGMLS\_VLAN\_AUDIT\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.ngMLS\_device\_name, 1, 8)

MINUS

SELECT ei.ngMLS\_device\_name

FROM NCM\_NGMLS\_VLAN\_AUDIT\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.ngMLS\_device\_name, 1, 6)

),

ISSUES AS

(

SELECT cm.ngMLS\_device\_name, 22 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.ngMLS\_device\_name, 2 issue\_id

FROM bad\_clli bc

)

SELECT distinct i.ngMLS\_device\_name, W.NE\_VLAN\_NUMBER, i.issue\_id, ''CI''

FROM ISSUES i, NCM\_NGMLS\_VLAN\_AUDIT\_wk w

where i.ngMLS\_device\_name = W.NGMLS\_DEVICE\_NAME

and W.NE\_VLAN\_NUMBER is not null

';

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

null;

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t mark 1st 6-char don''t match MTSO CLLIs';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

-- device not found in ncm vlan extract

sqlStmt := ' insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct aud.NGMLS\_DEVICE\_NAME, xng\_vlan\_number, 13, ''CI''

from NCM\_NGMLS\_VLAN\_AUDIT\_wk aud

where aud.ne\_hostname is null'

;

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Cant insert issue ''Not in NCM VLAN Extract''';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

-- device not found in GI

sqlStmt := ' insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct aud.NGMLS\_DEVICE\_NAME, ne\_vlan\_number, 15, ''CI''

from NCM\_NGMLS\_VLAN\_AUDIT\_wk aud

where aud.vlan\_inst\_id is null'

;

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Cant insert issue ''Not in GI''';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

-- if there is a path revision then the same VLAN will be present twice.

-- BUT that is NOT a duplicate

-- if NOT a revision path and VLAN# is present multiple times THEN it is dup

-- if 1 == Live and the other <> Live then mark <> Live as 'Live in Xng'

sqlStmt := 'update NCM\_NGMLS\_VLAN\_AUDIT\_WK

set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

, match\_status = nvl2(match\_status, ''Live Path Exists,'' ||match\_status , ''Live Path Exists'')

where vlan\_inst\_id = :p\_vlan\_inst\_id

--and ngmls\_vendor = :p\_ngMLS\_vendor

';

for rec in LIVE\_VLAN\_EXISTS

loop

BEGIN

execute immediate sqlStmt using rec.vlan\_inst\_id; --, p\_ngMLS\_vendor;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update ''Live Path Exists'' in table: NCM\_NGMLS\_VLAN\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

end loop;

commit;

-- IF BOTH and has issues then mark it NE

sqlStmt := ' update NCM\_NGMLS\_VLAN\_AUDIT\_WK aud

set match\_code = ''NE Only''

where exists (select 1

from

ngMLS\_VLAN\_AUDIT\_ISSUEs\_wk vi,

ngmls\_issue n

where aud.MATCH\_CODE = ''BOTH''

and aud.ngMLS\_vendor = ''CI''

and VI.NGMLS\_VENDOR=''CI''

and AUD.NGMLS\_DEVICE\_NAME = VI.NGMLS\_DEVICE\_NAME

and AUD.NE\_VLAN\_NUMBER = VI.VLAN\_NUMBER

and N.IS\_CRITICAL=''Y''

and N.NGMLS\_ISSUE\_ID = vi.NGMLS\_ISSUE\_ID

)';

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Cant update match\_status to ''NE Only''';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END insert\_CI\_ngMLS\_vlan\_issues;

-- END ;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* assign area / region to all devices

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure assign\_area\_region\_to\_devices is

methodName varchar2(30) := 'assign\_area\_region\_to\_devices';

message varchar2(300);

updStmt varchar2(32767);

CURSOR area\_region\_dev is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

select di.area, di.region, di.market, di.leaf\_domain\_inst\_id, ngMLS\_device\_name

from NCM\_GI\_CI\_NGMLS\_AUDIT\_WK aud

join DOMAINS di

on di.clli\_6 = substr(aud.NGMLS\_DEVICE\_NAME, 1, 6)

;

CURSOR eq\_area\_region\_dev is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

select aud.area, aud.region, aud.market, di.leaf\_domain\_inst\_id, ngMLS\_device\_name

from NCM\_GI\_CI\_NGMLS\_AUDIT\_WK aud

join vzwnet.equip\_domain\_map edm

on edm.equip\_inst\_id = aud.NE\_INST\_ID

join DOMAINS di

on di.leaf\_domain\_inst\_id = edm.DOMAIN\_INST\_ID

where aud.area is null

;

BEGIN

updStmt := 'update NCM\_GI\_CI\_NGMLS\_AUDIT\_WK set area = :area

, region = :region, market = :market

, leaf\_domain\_inst\_id = :leaf\_domain\_inst\_id

where ngMLS\_device\_name = :ngMLS\_device\_name';

for rec in area\_region\_dev

loop

BEGIN

execute immediate updStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.ngMLS\_device\_name;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update area/region for '|| rec.ngMLS\_device\_name||' in table: NCM\_GI\_CI\_NGMLS\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.logerror (methodName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

rollback;

RAISE;

END;

end loop;

-- area region for put xng\_only based on the area/region of ne\_inst\_id???????????

for rec in eq\_area\_region\_dev

loop

BEGIN

execute immediate updStmt using rec.area, rec.region, rec.market, rec.leaf\_domain\_inst\_id, rec.ngMLS\_device\_name;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update area/region from Xng Only for '|| rec.ngMLS\_device\_name||' in table: NCM\_GI\_CI\_NGMLS\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

end loop;

-- if there still are some devices which cant be mapped to any domain then mark those as unknown

updStmt := ' update NCM\_GI\_CI\_NGMLS\_AUDIT\_WK

set area = ''unknown''

, region = ''unknown''

, market = ''unknown''

, leaf\_domain\_inst\_id = 0

where area is null ';

execute immediate updStmt;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update area/region from to unknown in table: NCM\_GI\_CI\_NGMLS\_AUDIT\_WK';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END assign\_area\_region\_to\_devices;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NCM ngmls-extract, ncm vlan\_extract vs Xng, store summary by region

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

procedure load\_ngMLS\_regional\_summary is

-- get the all ngMLSs found in NCM ngMLS Extract

cursor ngMLSs\_NCM is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

,

NCM as

(

select distinct nvl(area, 'unknown') area, nvl(region, 'unknown') region, hostname

from ncm\_ngMLS\_wk ncm

left outer join DOMAINS di

on substr(hostname, 1, 6) = di.clli\_6

)

select area, region

, count (hostname) ncm\_ngMLSs

from NCM

group by rollup(area, region)

;

-- get the all ngMLSs found in NCM ngMLS-VLAN Extract

cursor ngMLSs\_VLAN\_NCM is

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

,

NCM as

(

select distinct nvl(area, 'unknown') area, nvl(region, 'unknown') region, hostname

from ncm\_ngMLS\_vlan\_wk ncm

left outer join DOMAINS di

on substr(hostname, 1, 6) = di.clli\_6

)

select area, region

, count (1) ncm\_ngMLSs

from NCM

group by rollup(area, region)

;

-- make sure duplicate matches in GI are not considered as good match-------??????

-- DUP means > 1 Live ones in Xng

-- if 1 is live and others are not, then that is NOT a DIP. Shirley's logic

-- this should be identifiable from the audit

-- get mismatches between ncm-ngMLS-vlan extract and gi

CURSOR NCM\_GI\_MISMATCH\_ngMLSs is

WITH

--NCM GI mismatch details

NCM\_GI\_MISMATCH as

(

-- any issue with the device is a mismatch

select distinct cdai.ngMLS\_device\_name -- distinct

from ngMLS\_device\_AUDIT\_ISSUEs\_wk cdai

where ngMLS\_vendor = 'CI'

)

,

GUD\_MATCHES as

(

-- LR 11/07

select ngMLS\_device\_name

from ncm\_gi\_ci\_ngMLS\_audit\_wk aud

where match\_code <> 'Xng Only'

minus

select cdai.ngMLS\_device\_name

from ngMLS\_device\_AUDIT\_ISSUEs\_wk cdai, ngmls\_issue n

where ngMLS\_vendor = 'CI'

and N.IS\_CRITICAL='Y'

and N.NGMLS\_ISSUE\_ID = CDAI.NGMLS\_ISSUE\_ID

)

,

NCM\_GI\_MATCH as

(

select aud.ngMLS\_device\_name

, case when live\_in\_xng = 'Y' then 1 else 0 end ncm\_gi\_l\_match

, case when live\_in\_xng <> 'Y' then 1 else 0 end ncm\_gi\_nl\_match

from ncm\_gi\_ci\_ngMLS\_audit\_wk aud

join GUD\_MATCHES gud

on gud.ngMLS\_device\_name = aud.ngMLS\_device\_name

)

,

MATCH\_COUNTS AS

(

select

case when ma.ngMLS\_device\_name is not null then substr(ma.ngMLS\_device\_name, 1, 6)

when ms.ngMLS\_device\_name is not null then substr(ms.ngMLS\_device\_name, 1, 6)

end clli\_6

, count(ma.ngMLS\_device\_name) ncm\_gi\_match

, sum(ma.ncm\_gi\_l\_match) ncm\_gi\_l\_match

, sum(ma.ncm\_gi\_nl\_match) ncm\_gi\_nl\_match

, count(ms.ngMLS\_device\_name) ncm\_gi\_mismatch

from NCM\_GI\_MATCH ma

full outer join NCM\_GI\_MISMATCH ms

on ma.ngMLS\_device\_name = ms.ngMLS\_device\_name

group by

case when ma.ngMLS\_device\_name is not null then substr(ma.ngMLS\_device\_name, 1, 6)

when ms.ngMLS\_device\_name is not null then substr(ms.ngMLS\_device\_name, 1, 6)

end

)

,

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region

from clli\_domain\_map\_v

where region <> 'NNO'

and region <> 'OSS'

)

select area, region

, sum(ncm\_gi\_match) NCM\_GI\_mATCH

, sum(ncm\_gi\_l\_match) NCM\_GI\_l\_mATCH

, sum(ncm\_gi\_nl\_match) NCM\_GI\_nl\_mATCH

, sum(ncm\_gi\_mismatch) NCM\_GI\_MISMATCH

from MATCH\_COUNTS cnt

join DOMAINS di

on cnt.clli\_6 = di.clli\_6

group by rollup(area, region)

;

methodName varchar2(30) := 'load\_ngMLS\_regional\_summary ';

message varchar2(300);

updStmt varchar2(32767);

updAreaStmt varchar2(32767);

updVzwStmt varchar2(32767);

insStmt varchar2(32767);

insAreaStmt varchar2(32767);

insVzwStmt varchar2(32767);

sqlStmt varchar2(32767);

l\_cnt number;

BEGIN

dbms\_output.put\_line('FYI: In '||methodName);

sqlStmt := ' delete from ncm\_gi\_ci\_ngMLS\_region\_summ\_wk ';

execute immediate sqlStmt;

commit;

BEGIN

-- from xng insert all CLLI and xng# in the table

sqlStmt := 'insert into ncm\_gi\_ci\_ngMLS\_region\_summ\_wk columns (area, region, GI\_NGMLSS, gi\_l\_ngMLSs, gi\_nl\_ngMLSs)

WITH

DOMAINS AS

(

select distinct area, region, market, leaf\_domain\_inst\_id, substr(clli, 1, 6) clli\_6

from clli\_domain\_map\_v

where region <> ''NNO''

and region <> ''OSS''

union

select ''unknown'' area, ''unknown'' region, ''unknown'' market, null leaf\_domain\_inst\_id, ''unknown'' clli\_6

from dual

)

,

Xng as

(

select distinct ngMLS\_device\_name, substr(ngMLS\_device\_name, 1, 6) clli\_6 , ngMLS.ne\_inst\_id

, nvl(case when status = ''Live'' then 1 else 0 end, 0) l

, nvl(case when status <> ''Live'' then 1 else 0 end, 0) nl

from XNG\_ngMLS\_equip\_wk ngMLS

where ngMLS\_vendor like ''CI''

and eq\_class\_type = ''NE''

)

,

GI\_CNTS as

(

select nvl(area, ''unknown'') area, nvl(region, ''unknown'') region, case when di.clli\_6 is null then ''unknown'' else xng.clli\_6 end clli\_6

, sum(l) l

, sum(nl) nl

from DOMAINS di

right outer join Xng

on xng.clli\_6 = di.clli\_6

group by nvl(area, ''unknown''), nvl(region, ''unknown'')

, case when di.clli\_6 is null then ''unknown'' else xng.clli\_6 end

)

select di.area, di.region

, sum(nvl(l, 0)) + sum (nvl(nl, 0)) gi\_l\_ngMLS

, sum(l) gi\_l\_ngMLS

, sum(nl) gi\_nl\_ngMLS

from DOMAINS di

left outer join GI\_CNTS xng

on xng.clli\_6 = di.clli\_6

group by rollup(di.area, di.region)

order by di.area, di.region

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '

||' GI metrics in table: ncm\_gi\_ci\_ngMLS\_region\_summ\_wk';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

-- insert new or update metric in the same table for total\_ngMLSs found in NCM in the ngMLS-extract

updStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_ngMLSs = :ncm\_ngMLSs

where area = :area

and region = :region

';

updAreaStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_ngMLSs = :ncm\_ngMLSs

where area = :area

and region is null

';

updVzwStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_ngMLSs = :ncm\_ngMLSs

where region is null

and area is null

';

for rec in ngMLSs\_NCM

loop

BEGIN

-- dbms\_output.put\_line(rec.area ||'/'|| rec.region||'/'||rec.ncm\_ngMLSs);

if rec.area is null then

-- dbms\_output.put\_line('In ngMLSs\_NCM VZW');

execute immediate updVzwStmt using rec.ncm\_ngMLSs;

else

if rec.region is null then

-- dbms\_output.put\_line('In ngMLSs\_NCM Area');

execute immediate updAreaStmt using rec.ncm\_ngMLSs, rec.area;

else

-- dbms\_output.put\_line('In ngMLSs\_NCM Region');

execute immediate updStmt using rec.ncm\_ngMLSs, rec.area, rec.region;

end if;

end if;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '

|| rec.area ||'/'|| rec.region||'/'||rec.ncm\_ngMLSs

||' for NCM-ngMLS-extract in table: ncm\_gi\_ci\_ngMLS\_region\_summ\_wk';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

end loop;

commit;

-- insert new or update metric in the same table for ncm total\_ngMLSs

updStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_ngMLSs\_w\_vlan = :ncm\_ngMLSs\_w\_vlan

where area = :area

and region = :region

';

updAreaStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_ngMLSs\_w\_vlan = :ncm\_ngMLSs\_w\_vlan

where area = :area

and region is null

';

updVzwStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_ngMLSs\_w\_vlan = :ncm\_ngMLSs\_w\_vlan

where area is null

and region is null

';

for rec in ngMLSs\_VLAN\_NCM

loop

BEGIN

if rec.area is null then execute immediate updVzwStmt using rec.ncm\_ngMLSs;

else

if rec.region is null then execute immediate updAreaStmt using rec.ncm\_ngMLSs, rec.area;

else execute immediate updStmt using rec.ncm\_ngMLSs, rec.area, rec.region;

end if;

end if;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update '

|| rec.area ||'/'|| rec.region||'/'|| rec.ncm\_ngMLSs

||' for NCM-ngMLS-VLAN-extract in table: ncm\_gi\_ci\_ngMLS\_region\_summ\_wk';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

end loop;

-- calc matches / mismatches metric between ncm and gi

updStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_gi\_mismatch = :ncm\_gi\_mismatch

, ncm\_gi\_match = :ncm\_gi\_match

, ncm\_gi\_l\_match = :ncm\_gi\_l\_match

, ncm\_gi\_nl\_match = :ncm\_gi\_nl\_match

where area = :area

and region = :region

';

updAreaStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_gi\_mismatch = :ncm\_gi\_mismatch

, ncm\_gi\_match = :ncm\_gi\_match

, ncm\_gi\_l\_match = :ncm\_gi\_l\_match

, ncm\_gi\_nl\_match = :ncm\_gi\_nl\_match

where area = :area

and region is null

';

updVzwStmt := 'update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_gi\_mismatch = :ncm\_gi\_mismatch

, ncm\_gi\_match = :ncm\_gi\_match

, ncm\_gi\_l\_match = :ncm\_gi\_l\_match

, ncm\_gi\_nl\_match = :ncm\_gi\_nl\_match

where area is null

and region is null

';

for rec in NCM\_GI\_MISMATCH\_ngMLSs

loop

BEGIN

if rec.area is null then

execute immediate updVzwStmt using rec.ncm\_gi\_mismatch, rec.ncm\_gi\_match, rec.ncm\_gi\_l\_match, rec.ncm\_gi\_nl\_match;

else

if rec.region is null then

execute immediate updAreaStmt using rec.ncm\_gi\_mismatch, rec.ncm\_gi\_match, rec.ncm\_gi\_l\_match, rec.ncm\_gi\_nl\_match, rec.area;

else

execute immediate updStmt using rec.ncm\_gi\_mismatch, rec.ncm\_gi\_match, rec.ncm\_gi\_l\_match, rec.ncm\_gi\_nl\_match, rec.area, rec.region;

end if;

end if;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update NCM-GI mismatch in table: ncm\_gi\_ci\_ngMLS\_dev\_summ\_wk';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

-- RAISE;

END;

END LOOP;

commit;

-- calculate the percentages

BEGIN

update ncm\_gi\_ci\_ngMLS\_region\_summ\_wk

set ncm\_ngMLSs\_wo\_vlan = nvl(ncm\_ngMLSs,0) - nvl(ncm\_ngMLSs\_w\_vlan,0)

, ncm\_ngMLSs\_w\_vlan\_per = round(nvl(ncm\_ngMLSs\_w\_vlan, 0) / decode (nvl(ncm\_ngMLSs,0), 0, 1, nvl(ncm\_ngMLSs,0)) \* 100, 2)

, ncm\_gi\_per = round(nvl(ncm\_gi\_match, 0) \* 2 / decode ((nvl(ncm\_ngMLSs,0) + nvl(gi\_ngMLSs,0)), 0, 1, nvl(ncm\_ngMLSs,0) + nvl(gi\_ngMLSs,0)) \* 100, 2)

;

commit;

EXCEPTION

WHEN OTHERS then

message := 'Error: '|| methodName ||'(): Can''t update percentages in table: ncm\_gi\_ci\_ngMLS\_region\_summ\_wk';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

rollback;

RAISE;

END;

END load\_ngMLS\_regional\_summary;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure load\_vlan\_device\_summary \*

\* \*

\* Purpose: Realize device summary report \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE load\_vlan\_device\_summary IS

methodName varchar2(30) := 'load\_vlan\_device\_summary';

message varchar2(300);

sqlStmt varchar2(32767);

BEGIN

dbms\_output.PUT\_LINE('FYI: In '||methodName);

-- generate device summary

load\_CI\_vlan\_device\_summary();

END load\_vlan\_device\_summary;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure load\_CI\_vlan\_device\_summary \*

\* \*

\* Purpose: Realize Cisco device summary report \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE load\_CI\_vlan\_device\_summary IS

methodName varchar2(30) := 'load\_CI\_vlan\_device\_summary';

message varchar2(300);

sqlStmt varchar2(32767);

BEGIN

dbms\_output.PUT\_LINE('FYI: In '||methodName);

sqlStmt:= 'delete from NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK where ngMLS\_vendor = ''CI''';

execute immediate sqlStmt;

commit;

-- generate summary for CI devices ONLY

sqlStmt := ' insert into NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK

columns (area, region, market, leaf\_domain\_inst\_id

, ngmls\_device\_name, ngMLS\_Vendor

, ne\_total\_ngmls\_vlan

, matched\_l\_ngmls\_vlan

, matched\_nl\_ngmls\_vlan

, mismatched\_ngmls\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_COMPLIANCE)

WITH

ngMLS as

(

select distinct area

, region, market, leaf\_domain\_inst\_id

, ngMLS\_device\_name, ngMLS\_vendor

from NCM\_GI\_CI\_NGMLS\_AUDIT\_WK ng

where ng.ncm\_ngmls\_vlan\_hostname is not null

),

ngMLS\_VLAN as

(

select distinct hostname NGMLS\_DEVICE\_NAME, vlan\_number

from ncm\_ngmls\_vlan\_wk ng

where ng.vlan\_number > 0

and ng.port\_ip is not null

)

,

CNT\_ngMLS\_VLANs as

(

select distinct ng.NGMLS\_DEVICE\_NAME, count(vlan\_number) total\_ne\_ngmls\_vlan

from ngMLS\_VLAN nv

join ngMLS ng

on ng.NGMLS\_DEVICE\_NAME = nv.NGMLS\_DEVICE\_NAME

group by ng.NGMLS\_DEVICE\_NAME

),

live\_match as

(

select distinct aud.NGMLS\_DEVICE\_NAME, aud.ne\_vlan\_number, vlan\_status

from NCM\_NGMLS\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.ngMLS\_vendor = ''CI''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status = ''Live''

)

,nonlive\_match as

(

select distinct aud.NGMLS\_DEVICE\_NAME, aud.ne\_vlan\_number

from NCM\_NGMLS\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.ngMLS\_vendor = ''CI''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status <> ''Live''

minus

select distinct aud.NGMLS\_DEVICE\_NAME, aud.ne\_vlan\_number

from NCM\_NGMLS\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.ngMLS\_vendor = ''CI''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status = ''Live''

) ,

CNT\_BOTH\_LIVE\_VLANs as

( -- good matches

select NGMLS\_DEVICE\_NAME, nvl (count(NGMLS\_DEVICE\_NAME), 0) matched\_l\_ngmls\_vlan

from live\_match aud

group by NGMLS\_DEVICE\_NAME

),

CNT\_BOTH\_NONLIVE\_VLANs as

( -- good matches

select NGMLS\_DEVICE\_NAME, nvl( count(NGMLS\_DEVICE\_NAME), 0) matched\_nl\_ngmls\_vlan

-- , nvl(sum(case when vlan\_status = ''Live'' then 1 else 0 end), 0) matched\_l\_ngmls\_vlan

-- , nvl(sum(case when vlan\_status <> ''Live'' then 1 else 0 end), 0) matched\_nl\_ngmls\_vlan

from nonlive\_match aud

group by aud.NGMLS\_DEVICE\_NAME

) ,

MISMATCH\_VLANs as

( -- bad matches

select distinct aud.NGMLS\_DEVICE\_NAME, aud.ne\_vlan\_number, aud.xng\_vlan\_number, vlan\_status

from NCM\_NGMLS\_VLAN\_AUDIT\_WK aud

left outer join xng\_reports.ngmls\_vlan\_audit\_issues d on aud.NGMLS\_DEVICE\_NAME = d.NGMLS\_DEVICE\_NAME

where match\_code <>''BOTH''

and aud.ngMLS\_vendor =''CI''

and (aud.ne\_vlan\_number > 0 or aud.xng\_vlan\_number > 0)

and (aud.NE\_VLAN\_NUMBER = D.VLAN\_NUMBER or aud.XNG\_VLAN\_NUMBER = D.VLAN\_NUMBER)

),

CNT\_MISMATCH\_VLANs as

( -- bad match cnts

select aud.NGMLS\_DEVICE\_NAME

, count(1) mismatch\_cnt

from MISMATCH\_VLANs aud

join ngMLS ng

on ng.NGMLS\_DEVICE\_NAME = aud.NGMLS\_DEVICE\_NAME

group by aud.NGMLS\_DEVICE\_NAME

)

select area, region, market, leaf\_domain\_inst\_id

, ng.ngmls\_device\_name, ngMLS\_Vendor

, sum(nvl(nv.total\_ne\_ngmls\_vlan, 0)) ne\_total\_ngmls\_vlan

, sum(nvl(matched\_l\_ngmls\_vlan, 0)) matched\_l\_ngmls\_vlan

, sum(nvl(matched\_nl\_ngmls\_vlan, 0)) matched\_nl\_ngmls\_vlan

, sum(nvl(mismatch\_cnt, 0)) mismatched\_ngmls\_vlan

, round(sum(NVL(matched\_nl\_ngmls\_vlan, 0)) / decode(sum(NVL(matched\_l\_ngmls\_vlan, 0)), 0, 1, sum(nvl(matched\_l\_ngmls\_vlan, 0)))

\* 100, 2) gi\_nl\_2\_total\_vlan\_pct

, round (sum(nvl(matched\_l\_ngmls\_vlan,0)) / decode (sum(nvl(nv.total\_ne\_ngmls\_vlan, 0)) - sum(nvl(matched\_nl\_ngmls\_vlan, 0)), 0 , 1

, sum(nvl(nv.total\_ne\_ngmls\_vlan,0)) - sum(nvl(matched\_nl\_ngmls\_vlan,0)))

\* 100 , 2) vlan\_COMPLIANCE

from ngMLS ng

left outer join CNT\_ngMLS\_VLANs nv

on nv.NGMLS\_DEVICE\_NAME = ng.NGMLS\_DEVICE\_NAME

left outer join CNT\_BOTH\_LIVE\_VLANs lmv

on lmv.NGMLS\_DEVICE\_NAME = ng.NGMLS\_DEVICE\_NAME

left outer join CNT\_BOTH\_NONLIVE\_VLANs nlmv

on nlmv.NGMLS\_DEVICE\_NAME = ng.NGMLS\_DEVICE\_NAME

left outer join CNT\_MISMATCH\_VLANs mv

on mv.NGMLS\_DEVICE\_NAME = ng.NGMLS\_DEVICE\_NAME

group by ng.area, ng.region, ng.market, leaf\_domain\_inst\_id, ngMLS\_Vendor, ng.NGMLS\_DEVICE\_NAME

';

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t summarize CI devices';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END load\_CI\_vlan\_device\_summary;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure load\_vlan\_regional\_summary \*

\* \*

\* Purpose: From device Vlan summary Realize regional summary \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE load\_vlan\_regional\_summary IS

methodName varchar2(30) := 'load\_vlan\_regional\_summary';

message varchar2(500);

sqlStmt varchar2(32767);

BEGIN

execute immediate 'truncate table NCM\_NGMLS\_VLAN\_REGION\_SUMM\_WK';

-- generate device summary

/\*sqlStmt:= ' insert into NCM\_NGMLS\_VLAN\_REGION\_SUMM\_WK

columns (area, region

, ne\_total\_ngmls\_vlan

, matched\_l\_ngmls\_vlan

, matched\_nl\_ngmls\_vlan

, mismatched\_ngmls\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_compliance)

select dlr.area, dlr.region

, sum(ne\_total\_ngmls\_vlan) ne\_total\_ngmls\_vlan

, sum(matched\_l\_ngmls\_vlan) matched\_l\_ngmls\_vlan

, sum(matched\_nl\_ngmls\_vlan) matched\_nl\_ngmls\_vlan

, sum(mismatched\_ngmls\_vlan) mismatched\_ngmls\_vlan

, round(avg(gi\_nl\_2\_total\_vlan\_pct), 2) gi\_nl\_2\_total\_vlan\_pct

, round(avg(vlan\_COMPLIANCE), 2) vlan\_compliance

from domains\_leaf\_reporting dlr

left outer join NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK ng

on dlr.market = ng.market

where dlr.AREA not in (''NNO'', ''OSS'')

group by rollup (dlr.area, dlr.region)

ORDER BY AREA, REGION '

;\*/

sqlStmt:= ' insert into NCM\_NGMLS\_VLAN\_REGION\_SUMM\_WK

columns (area, region

, ne\_total\_ngmls\_vlan

, matched\_l\_ngmls\_vlan

, matched\_nl\_ngmls\_vlan

, mismatched\_ngmls\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_compliance)

select dlr.area, dlr.region

, sum(ne\_total\_ngmls\_vlan) ne\_total\_ngmls\_vlan

, sum(matched\_l\_ngmls\_vlan) matched\_l\_ngmls\_vlan

, sum(matched\_nl\_ngmls\_vlan) matched\_nl\_ngmls\_vlan

, sum(mismatched\_ngmls\_vlan) mismatched\_ngmls\_vlan

, ROUND(SUM(NVL(MATCHED\_NL\_NGMLS\_VLAN, 0)) / DECODE(SUM(NVL(MATCHED\_L\_NGMLS\_VLAN, 0)), 0, 1, SUM(NVL(MATCHED\_L\_NGMLS\_VLAN, 0)))

\* 100, 2) GI\_NL\_2\_TOTAL\_VLAN\_PCT

, ROUND (SUM(NVL(MATCHED\_L\_NGMLS\_VLAN,0)) /

DECODE (SUM(NVL(NE\_TOTAL\_NGMLS\_VLAN, 0)) - SUM(NVL(MATCHED\_NL\_NGMLS\_VLAN, 0)), 0, 1, SUM(NVL(NE\_TOTAL\_NGMLS\_VLAN,0)) - SUM(NVL(MATCHED\_NL\_NGMLS\_VLAN,0)))

\* 100, 2) VLAN\_COMPLIANCE2

from domains\_leaf\_reporting dlr

left outer join NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK ng

on dlr.market = ng.market

where dlr.AREA not in (''NNO'', ''OSS'')

group by rollup (dlr.area, dlr.region)

ORDER BY AREA, REGION '

;

execute immediate sqlStmt;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t generate regional summary';

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END load\_vlan\_regional\_summary;

PROCEDURE move\_data\_from\_prod\_to\_wk is

sql\_stmt varchar2(32000);

begin

sql\_stmt:='delete from NCM\_GI\_CI\_NGMLS\_AUDIT\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NCM\_GI\_CI\_NGMLS\_AUDIT\_WK select \* from NCM\_GI\_CI\_NGMLS\_AUDIT';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NCM\_GI\_CI\_NGMLS\_AUDIT\_WK');

sql\_stmt:='delete from NCM\_GI\_CI\_NGMLS\_REGION\_SUMM\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NCM\_GI\_CI\_NGMLS\_REGION\_SUMM\_WK select \* from NCM\_GI\_CI\_NGMLS\_REGION\_SUMM';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NCM\_GI\_CI\_NGMLS\_REGION\_SUMM\_WK');

sql\_stmt:='delete from NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK select \* from NGMLS\_DEVICE\_AUDIT\_ISSUES';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK');

sql\_stmt:='delete from NCM\_NGMLS\_VLAN\_AUDIT\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NCM\_NGMLS\_VLAN\_AUDIT\_WK select \* from NCM\_NGMLS\_VLAN\_AUDIT';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NCM\_NGMLS\_VLAN\_AUDIT\_WK');

sql\_stmt:='delete from NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK select \* from NCM\_NGMLS\_VLAN\_DEVICE\_SUMM';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NCM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK');

sql\_stmt:='delete from NCM\_NGMLS\_VLAN\_REGION\_SUMM\_WK';

EXECUTE IMMEDIATE sql\_stmt;

commit;

sql\_stmt:='insert into NCM\_NGMLS\_VLAN\_REGION\_SUMM\_WK select \* from NCM\_NGMLS\_VLAN\_REGION\_SUMM';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Done populating NCM\_NGMLS\_VLAN\_REGION\_SUMM');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in move\_data\_from\_prod\_to\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

WATCHDOG.logerror('NCM\_NGMLS\_AUDIT',4000,SubStr('Error in move\_data\_from\_wk(): '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('NCM\_NGMLS\_AUDIT','STATUS\_FAILURE','N');

RAISE;

END move\_data\_from\_prod\_to\_wk;

END;

/

--------------------------------------------------------

-- DDL for Package Body NETSMART\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."NETSMART\_AUDIT" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: netsmart\_audit

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 5/2/2012 NSP Created this package.

1.1 5/11/2012 SME Realized detail query and

added report table functions

1.2 3/9/2015 SME Split audit into MSPP and ADM

methods.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure GetGraniteEquip \*

\* \*

\* Purpose: Finds all the top level Fujitsu FlashWave MSPP Equipment in Granite Inventory \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE GETGRANITEEQUIP IS

BEGIN

-- from granite get all equip that have vendor = FUJITSU and model like 'FLASHWAVE% or 'FW%'

--delete from xng\_fw\_equip;

EXECUTE IMMEDIATE 'TRUNCATE TABLE XNG\_FW\_EQUIP';

insert into xng\_fw\_equip ( NE\_INST\_ID, EQUIP\_INST\_ID, SITE\_INST\_ID,

PARENT\_EQ\_INST\_ID, EQ\_CLASS\_TYPE, DESCR, MODEL, TYPE, STATUS, VENDOR, EQ\_CLASS

)

SELECT TO\_NUMBER

(TRIM

(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH

(DECODE (LEVEL,

1, ei.equip\_inst\_id,

''

),

'/'

)

)

) ne\_inst\_id,

ei.equip\_inst\_id, ei.site\_inst\_id, ei.parent\_eq\_inst\_id,

DECODE (LEVEL,

1, 'NE',

DECODE (ei.eq\_class, 'S', 'SHELF', 'N/A')

) eq\_class\_type,

ei.descr, ei.model, ei.TYPE, ei.status,

ei.vendor, ei.eq\_class

FROM vzwnet.equip\_inst ei

WHERE (LEVEL = 1 OR ei.eq\_class = 'S')

START WITH ei.vendor = 'FUJITSU'

AND REGEXP\_LIKE (MODEL, 'FLASHWAVE|FW')

AND REGEXP\_LIKE (DESCR, '([[:alnum:]]{1,4}\s+)?[[:alnum:]]{8}MS\d-(R|N)FW\d{5}')

CONNECT BY PRIOR ei.equip\_inst\_id = ei.parent\_eq\_inst\_id;

COMMIT;

-- since we dont know top-equip from where to start looking for all children equip

-- in the previous start-with-connect-by the same equip might come is as both NE and SHELF.

-- if it does then delete those rows where it is a NE.

delete xng\_fw\_equip xfe

where xfe.EQ\_CLASS\_TYPE = 'NE'

and exists (

WITH

TO\_BE\_DELETED as

(

select equip\_inst\_id

from xng\_fw\_equip

where eq\_class\_type = 'SHELF'

intersect

select equip\_inst\_id

from xng\_fw\_equip

where eq\_class\_type = 'NE'

)

select 1

from TO\_BE\_DELETED tbd

where tbd.equip\_inst\_id = xfe.EQUIP\_INST\_ID

);

COMMIT;

-- delete dups if some equip\_inst\_id comes in > 1 as SHELF

/\*select equip\_inst\_id

from xng\_fw\_equip

where eq\_class\_type = ''SHELF''

group by equip\_inst\_id

having count(1) > 1

;

commit;

\*/

-- update the Xng NE rows that dont match Granite naming format

UPDATE XNG\_FW\_EQUIP SET PARSE\_STATUS = 'Name not Granite Stds compliant'

where not regexp\_like(descr, '^([[:alnum:]]{1,4}\s+)?[[:alnum:]]{8}MS\d-(R|N)FW\d{5}(\s\*\[[^][]+\])?$') -- granite std name

and eq\_class\_type = 'NE';

COMMIT;

END GETGRANITEEQUIP;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure GetADMGraniteEquip \*

\* \*

\* Purpose: Finds all the top level Fujitsu FlashWave ADM Equipment in Granite Inventory \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE GETADMGRANITEEQUIP IS

BEGIN

-- from granite get all equip that have vendor = FUJITSU and model like 'FLASHWAVE% or 'FW%' and is not a Flashwave MSPP device

EXECUTE IMMEDIATE 'TRUNCATE TABLE XNG\_ADM\_FW\_EQUIP';

insert into xng\_adm\_fw\_equip ( NE\_INST\_ID, EQUIP\_INST\_ID, SITE\_INST\_ID,

PARENT\_EQ\_INST\_ID, EQ\_CLASS\_TYPE, DESCR, MODEL, TYPE, STATUS, VENDOR, EQ\_CLASS

)

SELECT TO\_NUMBER

(TRIM

(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH

(DECODE (LEVEL,

1, ei.equip\_inst\_id,

''

),

'/'

)

)

) ne\_inst\_id,

ei.equip\_inst\_id, ei.site\_inst\_id, ei.parent\_eq\_inst\_id,

DECODE (LEVEL,

1, 'NE',

DECODE (ei.eq\_class, 'S', 'SHELF', 'N/A')

) eq\_class\_type,

ei.descr, ei.model, ei.TYPE, ei.status,

ei.vendor, ei.eq\_class

FROM vzwnet.equip\_inst ei

WHERE (LEVEL = 1 OR ei.eq\_class = 'S')

START WITH ei.vendor = 'FUJITSU'

AND REGEXP\_LIKE (MODEL, 'FLASHWAVE|FW')

AND NOT REGEXP\_LIKE (DESCR, '([[:alnum:]]{1,4}\s+)?[[:alnum:]]{8}MS\d-(R|N)FW\d{5}')

CONNECT BY PRIOR ei.equip\_inst\_id = ei.parent\_eq\_inst\_id;

COMMIT;

-- since we dont know top-equip from where to start looking for all children equip

-- in the previous start-with-connect-by the same equip might come is as both NE and SHELF.

-- if it does then delete those rows where it is a NE.

delete xng\_adm\_fw\_equip xfe

where xfe.EQ\_CLASS\_TYPE = 'NE'

and exists (

WITH

TO\_BE\_DELETED as

(

SELECT EQUIP\_INST\_ID

from xng\_adm\_fw\_equip

where eq\_class\_type = 'SHELF'

intersect

SELECT EQUIP\_INST\_ID

from xng\_adm\_fw\_equip

where eq\_class\_type = 'NE'

)

select 1

from TO\_BE\_DELETED tbd

where tbd.equip\_inst\_id = xfe.EQUIP\_INST\_ID

);

COMMIT;

-- delete dups if some equip\_inst\_id comes in > 1 as SHELF

/\*select equip\_inst\_id

from xng\_fw\_equip

where eq\_class\_type = ''SHELF''

group by equip\_inst\_id

having count(1) > 1

;

commit;

\*/

-- update the Xng NE rows that dont match Granite naming format

UPDATE XNG\_ADM\_FW\_EQUIP SET PARSE\_STATUS = 'Name not Granite Stds compliant'

where not regexp\_like(descr, '^([[:alnum:]]{1,4}\s+)?[[:alnum:]]{8}OS\d-(R|N)FW\d{5}(\s\*\[[^][]+\])?$') -- granite std name

and eq\_class\_type = 'NE';

COMMIT;

END GETADMGRANITEEQUIP;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure AuditFWEquip \*

\* \*

\* Purpose: Audit Fujitsu FlashWave MSPP Equipment against Granite Inventory \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE AUDITFWEQUIP IS

BEGIN

-- find matches in netsmart and Xng (look only at eq\_class\_type = NE)

--DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP';

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP (LEAF\_DOMAIN\_NAME, TID, TID\_TYPE, LOCATION, CONTACTS, IP\_ADDR,

MATCH\_CODE, MATCH\_STATUS, NE\_INST\_ID, DESCR, MODEL, VENDOR, MATCH8)

select distinct m.LEAF\_DOMAIN\_NAME, net.tid, net.tid\_type, net.location, net.contacts, net.ip\_addr

, case when xfe.parse\_status is null then 'BOTH' else 'NE Only' end match\_code

, XFE.PARSE\_STATUS

, xfe.ne\_inst\_id, xfe.descr, model, vendor, null

from NETSMART\_SNCNE net

JOIN XNG\_FW\_EQUIP XFE

ON XFE.DESCR LIKE '%'||NET.TID ||'%' -- need to make these match

LEFT OUTER JOIN CLLI\_DOMAIN\_MAP\_V M

ON SUBSTR(NET.TID,1,6) = SUBSTR(M.CLLI,1,6)

WHERE XFE.EQ\_CLASS\_TYPE = 'NE'

AND REGEXP\_LIKE(NET.TID,'^[[:alnum:]]{8}MS');

COMMIT;

-- if some TID matched > 1 granite ne\_inst\_id then mark it dup

update netsmart\_ne\_vs\_xng\_audit\_temp aud set match\_code = 'NE Only'

, match\_status = nvl2(match\_status, 'Duplicate in Granite, '||rtrim(match\_status), 'Duplicate in Granite')

WHERE EXISTS (SELECT 1

from netsmart\_ne\_vs\_xng\_audit\_temp i

where aud.TID = i.tid

group by i.tid

having count(1) > 1

);

COMMIT;

-- insert NE Only

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP (TID, TID\_TYPE, LOCATION, CONTACTS, IP\_ADDR, MATCH\_CODE, MATCH\_STATUS)

select tid, tid\_type, location, contacts, ip\_addr, 'NE Only', 'No match in Granite'

FROM (

select net.tid, net.tid\_type, net.location, net.contacts, ip\_addr

FROM NETSMART\_SNCNE NET

WHERE REGEXP\_LIKE(NET.TID,'^[[:alnum:]]{8}MS')

MINUS

SELECT AUD.TID, AUD.TID\_TYPE, AUD.LOCATION, AUD.CONTACTS, ip\_addr

from netsmart\_ne\_vs\_xng\_audit\_temp aud

);

COMMIT;

-- check the TID matches MTCE std, else update match\_code = ''NE Only'' and match\_status = ''TID not MTCE Stds compliant''

update netsmart\_ne\_vs\_xng\_audit\_temp aud set match\_code = 'NE Only',

match\_status = nvl2(match\_status, 'TID not MTCE Stds compliant, '||rtrim(match\_status), 'TID not MTCE Stds compliant')

where not regexp\_like(aud.tid, '[[:alnum:]]{8}MS[[:digit:]]-(R|N)FW[[:digit:]]{5}');

COMMIT;

-- check if TID''s don't match first 6-char match some MTSO CLLI then match\_code = ''NE Only'' and match\_status = ''First six chars of Target Id do not match any MTSO CLLIs''

UPDATE NETSMART\_NE\_VS\_XNG\_AUDIT\_temp AUD SET MATCH\_CODE = 'NE Only',

match\_status = nvl2(match\_status, rtrim(match\_status)||', First six chars of Target Id do not match any MTSO CLLIs', 'First six chars of Target Id do not match any MTSO CLLIs')

where not exists (

WITH

CLLI\_DOMAINS as

(

select distinct substr(clli, 1,6) clli\_6

FROM CLLI\_DOMAIN\_MAP\_V

WHERE UPPER(REGION) != 'NNO'

)

select 1

FROM CLLI\_DOMAINS C

where c.clli\_6 = substr(aud.tid, 1,6)

);

COMMIT;

-- check if TID''s match first 8-char match some MTSO CLLI then set match\_code = ''BOTH'' temporarily

UPDATE NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET AUD.MATCH8 = '1'

WHERE EXISTS (

WITH

CLLI\_DOMAINS as

(

select distinct clli, substr(clli, 1, 6) clli\_6

FROM CLLI\_DOMAIN\_MAP\_V

WHERE UPPER(REGION) != 'NNO'

)

select 1

FROM CLLI\_DOMAINS C

WHERE C.CLLI = SUBSTR(AUD.TID, 1,8)

);

COMMIT;

-- check if TID''s match first 6-char match some MTSO CLLI then match\_code = ''NE Only'' and match\_status = ''Only 1st 6-char of CLLI match MTSO''

UPDATE NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS = NVL2(MATCH\_STATUS, RTRIM(MATCH\_STATUS)||', Only first six chars of Target Id match a MTSO CLLI', 'Only first six chars of Target Id match a MTSO CLLI')

WHERE EXISTS (

WITH

CLLI\_DOMAINS as

(

select distinct clli, substr(clli, 1, 6) clli\_6

FROM CLLI\_DOMAIN\_MAP\_V

WHERE UPPER(REGION) != 'NNO'

)

select 1

FROM CLLI\_DOMAINS C

WHERE C.CLLI\_6 = SUBSTR(AUD.TID, 1,6) AND AUD.MATCH8 IS NULL

);

COMMIT;

-- Set Domain name for NNO Equipment

UPDATE NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP AUD SET LEAF\_DOMAIN\_NAME = 'NNO\_DOMAIN'

WHERE REGEXP\_LIKE(AUD.TID, '[[:alnum:]]{8}MS[[:digit:]]-NFW[[:digit:]]{5}');

-- insert Xng Only. These will only be the top-level equip NE\_inst\_id ONLY

-- these will be shown in details, but wont affect the summary

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP (NE\_INST\_ID, DESCR, MODEL, VENDOR, MATCH\_CODE, MATCH\_STATUS)

select ne\_inst\_id, descr, model, vendor, 'Granite Only', 'No match in NE'

from (

select ne\_inst\_id, descr, model, vendor

from xng\_fw\_equip xfe

minus

SELECT NE\_INST\_ID, DESCR, MODEL, VENDOR

from netsmart\_ne\_vs\_xng\_audit\_temp aud

);

COMMIT;

update netsmart\_ne\_vs\_xng\_audit\_temp aud set match\_status = rtrim(match\_status);

COMMIT;

--DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_WK;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_NE\_VS\_XNG\_AUDIT\_WK';

-- Move records into WK table where we already know the domain name.

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_WK

SELECT \* FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE LEAF\_DOMAIN\_NAME IS NOT NULL;

DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE LEAF\_DOMAIN\_NAME IS NOT NULL;

COMMIT;

-- Move XNG\_Only records with Leaf Domain name.

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR,

NE.MATCH8

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

WHERE

NE.TID IS NULL;

DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE NE\_INST\_ID IN (SELECT DISTINCT

NE.NE\_INST\_ID

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

WHERE

NE.TID IS NULL);

COMMIT;

-- Insert NE\_Only records with Leaf Domain name.

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR,

NE.MATCH8

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

;

DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE NE\_INST\_ID IN (SELECT DISTINCT

NE.NE\_INST\_ID

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID));

COMMIT;

-- The only items left should be NE Only Items

-- Insert NE\_Only records with only a match in the CLLI\_DOMAIN\_MAP (8 Character).

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR,

NE.MATCH8

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP NE

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.CLLI = SUBSTR(NE.TID,1,8));

DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE TID IN (SELECT DISTINCT

NE.TID

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NE

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.CLLI = SUBSTR(NE.TID,1,8)));

COMMIT;

-- Insert NE\_Only records with only a match in the CLLI\_DOMAIN\_MAP (6 Character).

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR,

NE.MATCH8

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP NE

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (SUBSTR(CDM.CLLI,1,6) = SUBSTR(NE.TID,1,6));

DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP WHERE TID IN (SELECT DISTINCT

NE.TID

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NE

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (SUBSTR(CDM.CLLI,1,6) = SUBSTR(NE.TID,1,6)));

COMMIT;

-- The only records left are those TIDs that do not match the CLLI DOMAIN MAP and have no match in Granite.

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_WK

SELECT NE.\* FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP NE;

--DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP';

COMMIT;

-- dont implement yet. even though Shirley says it is a good idea, she needs approvals from other area leads.

-- model # in tid\_type match model# in xng\_equip\_type

-- type of the NE\_inst\_id should be MSPP or FIBER MUX for 9500 and FIBRE MUX for all others

END AUDITFWEQUIP;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure AuditADMFWEquip \*

\* \*

\* Purpose: Audit Fujitsu FlashWave ADM Equipment against Granite Inventory \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE AUDITADMFWEQUIP IS

BEGIN

-- find matches in netsmart and Xng (look only at eq\_class\_type = NE)

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP';

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP (LEAF\_DOMAIN\_NAME, TID, TID\_TYPE, LOCATION, CONTACTS, IP\_ADDR,

MATCH\_CODE, MATCH\_STATUS, NE\_INST\_ID, DESCR, MODEL, VENDOR, MATCH8)

select distinct m.LEAF\_DOMAIN\_NAME, net.tid, net.tid\_type, net.location, net.contacts, net.ip\_addr

, case when xfe.parse\_status is null then 'BOTH' else 'NE Only' end match\_code

, XFE.PARSE\_STATUS

, xfe.ne\_inst\_id, xfe.descr, model, vendor, null

FROM NETSMART\_SNCNE NET

JOIN XNG\_ADM\_FW\_EQUIP XFE

ON XFE.DESCR LIKE '%'||NET.TID ||'%' -- need to make these match

LEFT OUTER JOIN CLLI\_DOMAIN\_MAP\_V M

ON SUBSTR(NET.TID,1,6) = SUBSTR(M.CLLI,1,6)

WHERE XFE.EQ\_CLASS\_TYPE = 'NE'

AND NOT REGEXP\_LIKE(NET.TID,'^[[:alnum:]]{8}MS');

COMMIT;

-- if some TID matched > 1 granite ne\_inst\_id then mark it dup

update netsmart\_adm\_ne\_vs\_xng\_adt\_tmp aud set match\_code = 'NE Only'

, match\_status = nvl2(match\_status, 'Duplicate in Granite, '||rtrim(match\_status), 'Duplicate in Granite')

WHERE EXISTS (SELECT 1

from netsmart\_adm\_ne\_vs\_xng\_adt\_tmp i

where aud.TID = i.tid

group by i.tid

having count(1) > 1

);

COMMIT;

-- insert NE Only

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP (TID, TID\_TYPE, LOCATION, CONTACTS, IP\_ADDR, MATCH\_CODE, MATCH\_STATUS)

select tid, tid\_type, location, contacts, ip\_addr, 'NE Only', 'No match in Granite'

FROM (

select net.tid, net.tid\_type, net.location, net.contacts, ip\_addr

FROM NETSMART\_SNCNE NET

WHERE NOT REGEXP\_LIKE(NET.TID,'^[[:alnum:]]{8}MS')

MINUS

SELECT AUD.TID, AUD.TID\_TYPE, AUD.LOCATION, AUD.CONTACTS, IP\_ADDR

from netsmart\_adm\_ne\_vs\_xng\_adt\_tmp aud

);

COMMIT;

-- check the TID matches MTCE std, else update match\_code = ''NE Only'' and match\_status = ''TID not MTCE Stds compliant''

update netsmart\_adm\_ne\_vs\_xng\_adt\_tmp aud set match\_code = 'NE Only',

MATCH\_STATUS = NVL2(MATCH\_STATUS, 'TID not MTCE Stds compliant, '||RTRIM(MATCH\_STATUS), 'TID not MTCE Stds compliant')

where not regexp\_like(aud.tid, '[[:alnum:]]{8}OS[[:digit:]]-(R|N)FW[[:digit:]]{5}');

COMMIT;

-- check if TID''s don't match first 6-char match some MTSO CLLI then match\_code = ''NE Only'' and match\_status = ''First six chars of Target Id do not match any MTSO CLLIs''

UPDATE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_tmp AUD SET MATCH\_CODE = 'NE Only',

match\_status = nvl2(match\_status, rtrim(match\_status)||', First six chars of Target Id do not match any MTSO CLLIs', 'First six chars of Target Id do not match any MTSO CLLIs')

where not exists (

WITH

CLLI\_DOMAINS as

(

select distinct substr(clli, 1,6) clli\_6

FROM CLLI\_DOMAIN\_MAP\_V

WHERE UPPER(REGION) != 'NNO'

)

select 1

from CLLI\_DOMAINS c

where c.clli\_6 = substr(aud.tid, 1,6)

);

COMMIT;

-- check if TID''s match first 8-char match some MTSO CLLI then set match\_code = ''BOTH'' temporarily

UPDATE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP AUD SET AUD.MATCH8 = '1'

WHERE EXISTS (

WITH

CLLI\_DOMAINS as

(

select distinct clli, substr(clli, 1, 6) clli\_6

FROM CLLI\_DOMAIN\_MAP\_V

WHERE UPPER(REGION) != 'NNO'

)

select 1

FROM CLLI\_DOMAINS C

WHERE C.CLLI = SUBSTR(AUD.TID, 1,8)

);

COMMIT;

-- check if TID''s match first 6-char match some MTSO CLLI then match\_code = ''NE Only'' and match\_status = ''Only 1st 6-char of CLLI match MTSO''

UPDATE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP AUD SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS = NVL2(MATCH\_STATUS, RTRIM(MATCH\_STATUS)||', Only first six chars of Target Id match a MTSO CLLI', 'Only first six chars of Target Id match a MTSO CLLI')

WHERE EXISTS (

WITH

CLLI\_DOMAINS as

(

select distinct clli, substr(clli, 1, 6) clli\_6

FROM CLLI\_DOMAIN\_MAP\_V

WHERE UPPER(REGION) != 'NNO'

)

select 1

FROM CLLI\_DOMAINS C

WHERE C.CLLI\_6 = SUBSTR(AUD.TID, 1,6) AND AUD.MATCH8 IS NULL

);

COMMIT;

-- Set Domain name for NNO Equipment

UPDATE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP AUD SET LEAF\_DOMAIN\_NAME = 'NNO\_DOMAIN'

WHERE REGEXP\_LIKE(AUD.TID, '[[:alnum:]]{8}OS[[:digit:]]-NFW[[:digit:]]{5}');

-- insert Xng Only. These will only be the top-level equip NE\_inst\_id ONLY

-- these will be shown in details, but wont affect the summary

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP (NE\_INST\_ID, DESCR, MODEL, VENDOR, MATCH\_CODE, MATCH\_STATUS)

SELECT NE\_INST\_ID, DESCR, MODEL, VENDOR, 'Granite Only', 'No match in NE'

from (

SELECT NE\_INST\_ID, DESCR, MODEL, VENDOR

from xng\_adm\_fw\_equip xfe

minus

SELECT NE\_INST\_ID, DESCR, MODEL, VENDOR

from netsmart\_adm\_ne\_vs\_xng\_adt\_tmp aud

);

COMMIT;

update netsmart\_adm\_ne\_vs\_xng\_adt\_tmp aud set match\_status = rtrim(match\_status);

COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK';

-- Move records into WK table where we already know the domain name.

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK

SELECT \* FROM NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP WHERE LEAF\_DOMAIN\_NAME IS NOT NULL;

DELETE FROM NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP WHERE LEAF\_DOMAIN\_NAME IS NOT NULL;

COMMIT;

-- Move XNG\_Only records with Leaf Domain name.

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR,

NE.MATCH8

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

WHERE

NE.TID IS NULL;

DELETE FROM NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP WHERE NE\_INST\_ID IN (SELECT DISTINCT

NE.NE\_INST\_ID

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

WHERE

NE.TID IS NULL);

COMMIT;

-- Insert NE\_Only records with Leaf Domain name.

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR,

NE.MATCH8

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID)

;

DELETE FROM NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP WHERE NE\_INST\_ID IN (SELECT DISTINCT

NE.NE\_INST\_ID

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK NE

JOIN

VZWNET.EQUIP\_INST E ON E.EQUIP\_INST\_ID = NE.NE\_INST\_ID

JOIN

VZWNET.EQUIP\_DOMAIN\_MAP EDM ON EDM.EQUIP\_INST\_ID = E.EQUIP\_INST\_ID

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.LEAF\_DOMAIN\_INST\_ID = EDM.DOMAIN\_INST\_ID));

COMMIT;

-- Insert NE\_Only records with only a match in the CLLI\_DOMAIN\_MAP (8 Character).

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR,

NE.MATCH8

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP NE

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.CLLI = SUBSTR(NE.TID,1,8));

DELETE FROM NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP WHERE TID IN (SELECT DISTINCT

NE.TID

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK NE

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (CDM.CLLI = SUBSTR(NE.TID,1,8)));

COMMIT;

-- Insert NE\_Only records with only a match in the CLLI\_DOMAIN\_MAP (6 Character).

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK

SELECT DISTINCT

CDM.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

NE.DESCR,

NE.MODEL,

NE.VENDOR,

NE.MATCH8

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP NE

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (SUBSTR(CDM.CLLI,1,6) = SUBSTR(NE.TID,1,6));

DELETE FROM NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP WHERE TID IN (SELECT DISTINCT

TID

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK NE

JOIN

CLLI\_DOMAIN\_MAP\_V CDM ON (SUBSTR(CDM.CLLI,1,6) = SUBSTR(NE.TID,1,6)));

COMMIT;

-- The only records left are those TIDs that do not match the CLLI DOMAIN MAP and have no match in Granite.

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK

SELECT NE.\* FROM NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP NE;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP';

COMMIT;

-- dont implement yet. even though Shirley says it is a good idea, she needs approvals from other area leads.

-- model # in tid\_type match model# in xng\_equip\_type

-- type of the NE\_inst\_id should be MSPP or FIBER MUX for 9500 and FIBRE MUX for all others

END AUDITADMFWEQUIP;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure GenSummary \*

\* \*

\* Purpose: Realize summary report \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE GENSUMMARY IS

BEGIN

--DELETE FROM NETSMART\_EQ\_SUMM\_REGION\_WK;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_EQ\_SUMM\_REGION\_WK';

-- generate summary

-- insert into netsmart\_eq\_summ\_region\_wk (area, region, TOTAL\_NE, matched\_ne, compliance)

-- WITH

-- DEVICE\_DOMAIN\_MAP as

-- (

-- -- get all discovered NNO devices

-- select net.TID, net.TID\_TYPE, 'NNO' area, 'NNO' region

-- from netsmart\_SNCNE net

-- where instr(net.TID, '-', 1) <> 0

-- and substr(net.TID, instr(net.TID, '-', 1)+1, 1) = 'N'

-- union

-- -- get all discovered regional devices

-- select net.TID, net.TID\_TYPE, area, region

-- from netsmart\_SNCNE net

-- join XNG\_REPORTS.CLLI\_DOMAIN\_MAP\_V cdmv

-- ON SUBSTR(cdmv.CLLI,1,6) = SUBSTR(net.TID,1,6)

-- where instr(net.TID, '-', 1) <> 0

-- and substr(net.TID, instr(net.TID, '-', 1)+1, 1) = 'R'

-- and cdmv.REGION <> 'NNO'

-- )

-- ,

-- UNK\_DEVICES as

-- (

-- -- get all discovered devices that match and dont map to domain

-- select net.TID, net.TID\_TYPE

-- from netsmart\_SNCNE net

-- minus

-- select TID, TID\_TYPE

-- from DEVICE\_DOMAIN\_MAP

-- )

-- ,

-- -- all discovered devices

-- ALL\_DEVICES\_DOMAIN\_MAP as

-- (

-- select TID, TID\_TYPE, area, region

-- from DEVICE\_DOMAIN\_MAP

-- union

-- select TID, TID\_TYPE, 'unknown' area, 'unknown' region

-- from UNK\_DEVICES

-- )

-- ,

-- -- get good matches per region

-- MATCHES AS

-- (

-- select distinct ddm.tid, area, region

-- from ALL\_DEVICES\_DOMAIN\_MAP ddm

-- join netsmart\_ne\_vs\_xng\_audit\_wk aud

-- on ddm.tid = aud.tid

-- where match\_code = 'BOTH'

-- )

-- ,

-- -- all reporting domains

-- DOMAINS as

-- (

-- select area, region

-- from domains\_regional\_reporting drr

-- where drr.area <> 'OSS'

-- union

-- select 'unknown' area, 'unknown' region

-- from dual

-- )

-- ,

-- -- gen metrics

-- METRIC as

-- (

-- SELECT

-- drr.area, drr.region

-- , nvl(COUNT(ddm.TID), 0) TOTAL\_NE

-- , nvl(count(m.tid), 0) matched\_ne

-- FROM DOMAINS drr

-- left outer join ALL\_DEVICES\_DOMAIN\_MAP ddm

-- on drr.region = ddm.region

-- left outer join MATCHES m

-- on drr.region = m.region

-- and ddm.tid = m.tid

-- GROUP BY ROLLUP(drr.area, drr.region)

-- )

-- SELECT drr.\*

-- , round(nvl(matched\_ne, 0) / decode(total\_ne, 0, 1, total\_ne) \* 100, 2) COMPLIANCE

-- FROM METRIC drr

-- ORDER BY area,region;

-- generate summary

-- insert into netsmart\_eq\_summ\_region\_wk (area, region, TOTAL\_NE, MATCHED\_NE, PERCENT\_MATCHED, PASSED\_AUDIT, COMPLIANCE)

-- WITH

-- DEVICE\_DOMAIN\_MAP AS

-- (

-- -- get all discovered NNO devices

-- SELECT DISTINCT NET.TID, NET.TID\_TYPE, 'NNO' AREA, 'NNO' REGION, COUNT(NET.NE\_INST\_ID) HAS\_EQUIP

-- FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NET

-- JOIN XNG\_REPORTS.CLLI\_DOMAIN\_MAP\_V CDMV

-- ON SUBSTR(CDMV.CLLI,1,6) = SUBSTR(NET.TID,1,6)

-- WHERE REGEXP\_LIKE(NET.TID, '-NFW', 'i') OR CDMV.REGION = 'NNO'

-- GROUP BY NET.TID, NET.TID\_TYPE, 'NNO','NNO'

-- UNION

-- -- get all discovered regional devices

-- SELECT DISTINCT NET.TID, NET.TID\_TYPE, CDMV.AREA, CDMV.REGION, COUNT(NET.NE\_INST\_ID) HAS\_EQUIP

-- FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NET

-- JOIN XNG\_REPORTS.CLLI\_DOMAIN\_MAP\_V CDMV

-- ON SUBSTR(CDMV.CLLI,1,6) = SUBSTR(NET.TID,1,6)

-- where NOT REGEXP\_LIKE(NET.TID, '-NFW', 'i')

-- AND CDMV.REGION <> 'NNO'

-- GROUP BY NET.TID, NET.TID\_TYPE, CDMV.AREA, CDMV.REGION

-- )

-- ,

-- UNK\_DEVICES AS

-- (

-- -- get all discovered devices that match and don't map to domain

-- SELECT DISTINCT NET.TID, NET.TID\_TYPE, COUNT(NET.NE\_INST\_ID) HAS\_EQUIP

-- FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NET

-- GROUP BY NET.TID, NET.TID\_TYPE

-- MINUS

-- SELECT DISTINCT DDM.TID, DDM.TID\_TYPE, DDM.HAS\_EQUIP

-- FROM DEVICE\_DOMAIN\_MAP DDM

-- )

-- ,

-- -- all discovered devices

-- ALL\_DEVICES\_DOMAIN\_MAP AS

-- (

-- SELECT DISTINCT DDM.TID, DDM.TID\_TYPE, DDM.AREA, DDM.REGION, DDM.HAS\_EQUIP

-- FROM DEVICE\_DOMAIN\_MAP DDM

-- UNION

-- SELECT DISTINCT UNK.TID, UNK.TID\_TYPE, 'unknown' AREA, 'unknown' REGION, UNK.HAS\_EQUIP

-- FROM UNK\_DEVICES UNK

-- )

-- ,

-- -- get good matches per region

-- MATCHES AS

-- (

-- SELECT DISTINCT DDM.TID, DDM.AREA, DDM.REGION, DDM.HAS\_EQUIP

-- FROM ALL\_DEVICES\_DOMAIN\_MAP DDM

-- JOIN NETSMART\_NE\_VS\_XNG\_AUDIT\_WK AUD

-- ON DDM.TID = AUD.TID

-- WHERE MATCH\_CODE = 'BOTH'

-- )

-- ,

-- -- all reporting domains

-- DOMAINS AS

-- (

-- SELECT AREA, REGION

-- FROM DOMAINS\_REGIONAL\_REPORTING DRR

-- WHERE DRR.AREA <> 'OSS'

-- UNION

-- SELECT 'unknown' AREA, 'unknown' REGION

-- FROM DUAL

-- )

-- ,

-- -- gen metrics

-- METRIC AS

-- (

-- SELECT

---- DECODE(GROUPING(drr.AREA),1,'zzzzzz',drr.AREA) AREA, drr.region

-- DRR.AREA, DRR.REGION

-- , NVL(COUNT(DDM.TID), 0) TOTAL\_NE

-- , SUM(DECODE(NVL(DDM.HAS\_EQUIP, 0),0,0,1)) MATCHED\_NE

-- , (CASE SUM(DECODE(NVL(DDM.HAS\_EQUIP, 0),0,0,1)) WHEN 0 THEN ' 0.00' ELSE (TO\_CHAR(ROUND((SUM(DECODE(NVL(DDM.HAS\_EQUIP, 0),0,0,1)))/COUNT(DDM.TID)\*100,2),'990.00')) END) PERCENT\_MATCHED

-- , NVL(COUNT(M.TID), 0) PASSED\_AUDIT

-- FROM DOMAINS DRR

-- LEFT OUTER JOIN ALL\_DEVICES\_DOMAIN\_MAP DDM

-- ON DRR.REGION = DDM.REGION

-- LEFT OUTER JOIN MATCHES M

-- ON DRR.REGION = M.REGION

-- AND DDM.TID = M.TID

-- GROUP BY ROLLUP(DRR.AREA, DRR.REGION)

-- )

-- SELECT DRR.\*

-- , ROUND(NVL(PASSED\_AUDIT, 0) / DECODE(TOTAL\_NE, 0, 1, TOTAL\_NE) \* 100, 2) COMPLIANCE

-- FROM METRIC DRR

-- ORDER BY DRR.AREA,DRR.REGION;

-- generate summary

insert into netsmart\_eq\_summ\_region\_wk (area, region, TOTAL\_NE, DUPLICATE\_NE, MATCHED\_NE, PERCENT\_MATCHED, PASSED\_AUDIT, NAMING\_COMPLIANCE, DISCOVERED\_XCONNECTS, LIVE\_XCONNECTS, OTHER\_XCONNECTS, TOTAL\_COMPLIANCE)

WITH DEVICEMETRICS AS

(

SELECT

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

(CASE (COUNT(TID) - 1) WHEN 0 THEN 0 ELSE 1 END) HAS\_DUPLICATES,

MAX(DECODE(NE\_INST\_ID, NULL, 0, 1)) NE\_MATCHED,

MAX(DECODE(UPPER(TRIM(MATCH\_CODE)), 'BOTH', 1, 0)) PASSED\_AUDIT,

MAX(NUM\_XCONNECT) NUM\_XCONNECT,

MAX(LIVE\_XCONNECTS) LIVE\_XCONNECTS,

MAX(OTHER\_XCONNECTS) OTHER\_XCONNECTS

FROM

TABLE(XNG\_REPORTS.NETSMART\_AUDIT.NS\_DETAIL\_TBL\_WRK(1, 10000))

WHERE

TID IS NOT NULL

GROUP BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID

)

,

DOMAINS AS

(

SELECT AREA, REGION FROM DOMAINS\_REGIONAL\_REPORTING DRR WHERE DRR.AREA <> 'OSS'

UNION

SELECT 'unknown' AREA, 'unknown' REGION FROM DUAL

)

SELECT

DRR.AREA

, DRR.REGION

, NVL(COUNT(DM.TID),0) TOTAL\_NE

, NVL(SUM(DM.HAS\_DUPLICATES), 0) DUPLICATE\_NE

, NVL(SUM(DM.NE\_MATCHED), 0) MATCHED\_NE

, (CASE SUM(DECODE(NVL(DM.NE\_MATCHED, 0),0,0,1)) WHEN 0 THEN ' 0.00' ELSE (TO\_CHAR(ROUND((SUM(DECODE(NVL(DM.NE\_MATCHED, 0),0,0,1)))/COUNT(DM.TID)\*100,2),'990.00')) END) PERCENT\_MATCHED

, NVL(SUM(DM.PASSED\_AUDIT), 0) PASSED\_AUDIT

, (CASE NVL(COUNT(DM.TID), 0) WHEN 0 THEN ' 0.00' ELSE (TO\_CHAR(ROUND((SUM(DM.PASSED\_AUDIT)/COUNT(DM.TID)\*100),2),'990.00')) END) NAMING\_COMPLIANCE

, NVL(SUM(DM.NUM\_XCONNECT), 0) DISCOVERED\_XCONNECTS

, NVL(SUM(DM.LIVE\_XCONNECTS), 0) LIVE\_XCONNECTS

, NVL(SUM(DM.OTHER\_XCONNECTS), 0) OTHER\_XCONNECTS

, CASE

WHEN NVL(SUM(DM.NUM\_XCONNECT), 0) > 0

THEN

--TO\_CHAR(ROUND(((NVL(SUM(DM.PASSED\_AUDIT), 0) + NVL(SUM(DM.LIVE\_XCONNECTS), 0) + NVL(SUM(DM.OTHER\_XCONNECTS), 0))/(NVL(COUNT(DM.TID),0) + NVL(SUM(DM.NUM\_XCONNECT), 0)))\*100,2),'990.00')

TO\_CHAR(ROUND(((NVL(SUM(DM.LIVE\_XCONNECTS), 0) + NVL(SUM(DM.OTHER\_XCONNECTS), 0))/(NVL(SUM(DM.NUM\_XCONNECT), 0)))\*100,2),'990.00')

ELSE

' 0.00'

END TOTAL\_COMPLIANCE

FROM

DOMAINS DRR

LEFT OUTER JOIN

DEVICEMETRICS DM ON DM.REGION = DRR.REGION

GROUP BY

ROLLUP(DRR.AREA, DRR.REGION)

ORDER BY

AREA,

REGION;

COMMIT;

END GENSUMMARY;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure GenADMSummary \*

\* \*

\* Purpose: Realize ADM Summary report \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE GENADMSUMMARY IS

BEGIN

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_EQ\_SUMM\_REGION\_WK';

-- generate summary

insert into netsmart\_adm\_eq\_summ\_region\_wk (area, region, TOTAL\_NE, DUPLICATE\_NE, MATCHED\_NE, PERCENT\_MATCHED, PASSED\_AUDIT, NAMING\_COMPLIANCE, DISCOVERED\_XCONNECTS, LIVE\_XCONNECTS, OTHER\_XCONNECTS, TOTAL\_COMPLIANCE)

WITH DEVICEMETRICS AS

(

SELECT

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

(CASE (COUNT(TID) - 1) WHEN 0 THEN 0 ELSE 1 END) HAS\_DUPLICATES,

MAX(DECODE(NE\_INST\_ID, NULL, 0, 1)) NE\_MATCHED,

MAX(DECODE(UPPER(TRIM(MATCH\_CODE)), 'BOTH', 1, 0)) PASSED\_AUDIT,

MAX(NUM\_XCONNECT) NUM\_XCONNECT,

MAX(LIVE\_XCONNECTS) LIVE\_XCONNECTS,

MAX(OTHER\_XCONNECTS) OTHER\_XCONNECTS

FROM

TABLE(XNG\_REPORTS.NETSMART\_AUDIT.NS\_ADM\_DETAIL\_TBL\_WRK(1, 10000))

WHERE

TID IS NOT NULL

GROUP BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID

)

,

DOMAINS AS

(

SELECT DISTINCT AREA, REGION FROM XNG\_REPORTS.CLLI\_DOMAIN\_MAP\_V DRR

UNION

SELECT 'unknown' AREA, 'unknown' REGION FROM DUAL

)

SELECT

DRR.AREA

, DRR.REGION

, NVL(COUNT(DM.TID),0) TOTAL\_NE

, NVL(SUM(DM.HAS\_DUPLICATES), 0) DUPLICATE\_NE

, NVL(SUM(DM.NE\_MATCHED), 0) MATCHED\_NE

, (CASE SUM(DECODE(NVL(DM.NE\_MATCHED, 0),0,0,1)) WHEN 0 THEN ' 0.00' ELSE (TO\_CHAR(ROUND((SUM(DECODE(NVL(DM.NE\_MATCHED, 0),0,0,1)))/COUNT(DM.TID)\*100,2),'990.00')) END) PERCENT\_MATCHED

, NVL(SUM(DM.PASSED\_AUDIT), 0) PASSED\_AUDIT

, (CASE NVL(COUNT(DM.TID), 0) WHEN 0 THEN ' 0.00' ELSE (TO\_CHAR(ROUND((SUM(DM.PASSED\_AUDIT)/COUNT(DM.TID)\*100),2),'990.00')) END) NAMING\_COMPLIANCE

, NVL(SUM(DM.NUM\_XCONNECT), 0) DISCOVERED\_XCONNECTS

, NVL(SUM(DM.LIVE\_XCONNECTS), 0) LIVE\_XCONNECTS

, NVL(SUM(DM.OTHER\_XCONNECTS), 0) OTHER\_XCONNECTS

, CASE

WHEN NVL(SUM(DM.NUM\_XCONNECT), 0) > 0

THEN

--TO\_CHAR(ROUND(((NVL(SUM(DM.PASSED\_AUDIT), 0) + NVL(SUM(DM.LIVE\_XCONNECTS), 0) + NVL(SUM(DM.OTHER\_XCONNECTS), 0))/(NVL(COUNT(DM.TID),0) + NVL(SUM(DM.NUM\_XCONNECT), 0)))\*100,2),'990.00')

TO\_CHAR(ROUND(((NVL(SUM(DM.LIVE\_XCONNECTS), 0) + NVL(SUM(DM.OTHER\_XCONNECTS), 0))/(NVL(SUM(DM.NUM\_XCONNECT), 0)))\*100,2),'990.00')

ELSE

' 0.00'

END TOTAL\_COMPLIANCE

FROM

DOMAINS DRR

LEFT OUTER JOIN

DEVICEMETRICS DM ON DM.REGION = DRR.REGION

GROUP BY

ROLLUP(DRR.AREA, DRR.REGION)

ORDER BY

AREA,

REGION;

COMMIT;

END GENADMSUMMARY;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure GenDetail \*

\* \*

\* Purpose: Realize report details table for fast lookup \*

\* \*

\* Note: No longer used due to improvements in the table function. \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- PROCEDURE GENDETAIL IS

--

-- BEGIN

--

-- DELETE FROM NETSMART\_EQ\_DETAIL\_REGION\_WK;

--

-- COMMIT;

--

-- INSERT INTO NETSMART\_EQ\_DETAIL\_REGION\_WK (AREA, REGION, LEAF\_DOMAIN\_NAME, TID, TID\_TYPE, NE\_INST\_ID, DESCR, SITE\_INST\_ID, SITE\_HUM\_ID, TYPE, VENDOR, MODEL, MATCH\_CODE, MATCH\_STATUS)

--

-- SELECT DISTINCT

-- CDM.AREA,

-- CDM.REGION,

-- CDM.LEAF\_DOMAIN\_NAME,

-- NE.TID,

-- NE.TID\_TYPE,

-- NE.NE\_INST\_ID,

-- EQ.DESCR,

-- EQ.SITE\_INST\_ID,

-- S.SITE\_HUM\_ID,

-- EQ.TYPE,

-- EQ.VENDOR,

-- EQ.MODEL,

-- NE.MATCH\_CODE,

-- NE.MATCH\_STATUS

-- FROM

-- NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NE

-- LEFT OUTER JOIN

-- XNG\_REPORTS.CLLI\_DOMAIN\_MAP\_V CDM ON (SUBSTR(CDM.CLLI,1,6) = SUBSTR(NE.TID,1,6))

-- LEFT OUTER JOIN

-- VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

-- LEFT OUTER JOIN

-- VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

-- ORDER BY AREA, REGION, LEAF\_DOMAIN\_NAME, TID, DESCR;

--

-- COMMIT;

--

-- END GENDETAIL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Audit\_XConnects \*

\* \*

\* Purpose: Audit FlashWave cross connects against XNG \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE AUDIT\_XCONNECTS IS

BEGIN

--DELETE FROM NETSMART\_FUJITSU\_EQUIPMENT;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_FUJITSU\_EQUIPMENT';

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_XCONNECT\_EQUIPMENT';

INSERT INTO NETSMART\_XCONNECT\_EQUIPMENT

SELECT

LEVEL NE\_LEVEL,

TO\_NUMBER

(TRIM

(BOTH '/' FROM SYS\_CONNECT\_BY\_PATH(

DECODE (LEVEL, 1, EI.EQUIP\_INST\_ID, '' ),

'/'

)

)

) NE\_INST\_ID,

EI.EQUIP\_INST\_ID,

EI.SITE\_INST\_ID,

EI.PARENT\_EQ\_INST\_ID,

DECODE (LEVEL,

1, 'NE',

DECODE (EI.EQ\_CLASS, 'S', 'SHELF', 'N/A')

) EQ\_CLASS\_TYPE,

EI.DESCR,

EI.MODEL,

EI.TYPE,

EI.STATUS,

EI.VENDOR,

EI.EQ\_CLASS

FROM

VZWNET.EQUIP\_INST EI

WHERE

(LEVEL = 1 OR EI.EQ\_CLASS = 'S')

START WITH

EI.VENDOR = 'FUJITSU'

AND REGEXP\_LIKE (MODEL, 'FLASHWAVE|FW')

CONNECT BY

PRIOR EI.EQUIP\_INST\_ID = EI.PARENT\_EQ\_INST\_ID;

COMMIT;

INSERT INTO NETSMART\_FUJITSU\_EQUIPMENT

SELECT

NE.TID,

NE.TID\_TYPE,

NXE.NE\_LEVEL,

NXE.NE\_INST\_ID,

NXE.EQUIP\_INST\_ID,

NXE.SITE\_INST\_ID,

NXE.PARENT\_EQ\_INST\_ID,

NXE.EQ\_CLASS\_TYPE,

NXE.DESCR,

NXE.MODEL,

NXE.TYPE,

NXE.STATUS,

NXE.VENDOR,

NXE.EQ\_CLASS

FROM

NETSMART\_XCONNECT\_EQUIPMENT NXE

JOIN

NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NE ON (NE.NE\_INST\_ID = NXE.NE\_INST\_ID)

WHERE

NE.TID IS NOT NULL;

COMMIT;

UPDATE NETSMART\_FUJITSU\_EQUIPMENT

SET PARENT\_EQ\_INST\_ID = NULL

WHERE EQUIP\_INST\_ID IN

(SELECT

NE2.EQUIP\_INST\_ID

FROM

NETSMART\_FUJITSU\_EQUIPMENT NE2

WHERE

NE2.PARENT\_EQ\_INST\_ID IS NOT NULL

AND NOT EXISTS (SELECT

NE.EQUIP\_INST\_ID

FROM

NETSMART\_FUJITSU\_EQUIPMENT NE

WHERE

NE.EQUIP\_INST\_ID = NE2.PARENT\_EQ\_INST\_ID

)

);

COMMIT;

--DELETE FROM NETSMART\_XNG\_PORTS;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_XNG\_PORTS';

INSERT INTO NETSMART\_XNG\_PORTS

SELECT DISTINCT

NE.TID,

NE.TID\_TYPE,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-') SLOT,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '') PORT\_HUM\_ID,

EP.PORT\_ACCESS\_ID,

CPE.CIRC\_PATH\_INST\_ID,

LI.LEG\_INST\_ID,

CPE.MEMBER\_NBR,

CPI.A\_SIDE\_SITE\_ID,

S1.SITE\_HUM\_ID A\_SIDE\_SITE\_HUM\_ID,

CPI.Z\_SIDE\_SITE\_ID,

S2.SITE\_HUM\_ID Z\_SIDE\_SITE\_HUM\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.BANDWIDTH,

CPI.STATUS PATH\_STATUS,

CPI.NEXT\_PATH\_INST\_ID,

CPI.PREV\_PATH\_INST\_ID,

NE.NE\_INST\_ID,

NE.EQUIP\_INST\_ID,

NE.PARENT\_EQ\_INST\_ID,

NE.SITE\_INST\_ID,

NE.VENDOR,

NE.MODEL,

NE.DESCR,

EP.PORT\_INST\_ID,

EP.CARD\_INST\_ID

FROM

NETSMART\_FUJITSU\_EQUIPMENT NE

JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.EQUIP\_INST\_ID)

JOIN

VZWNET.EPA EP ON (EP.EQUIP\_INST\_ID = EQ.EQUIP\_INST\_ID)

JOIN

VZWNET.CIRC\_PATH\_ELEMENT CPE ON CPE.PORT\_INST\_ID = EP.PORT\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_INST LI ON LI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_MEMBER PLM ON PLM.LEG\_INST\_ID = LI.LEG\_INST\_ID AND PLM.ELEMENT\_INST\_ID = CPE.ELEMENT\_INST\_ID

JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.SITE\_INST S1 ON S1.SITE\_INST\_ID = CPI.A\_SIDE\_SITE\_ID

JOIN

VZWNET.SITE\_INST S2 ON S2.SITE\_INST\_ID = CPI.Z\_SIDE\_SITE\_ID

JOIN

VZWNET.CARD\_INST CRD ON (CRD.CARD\_INST\_ID = EP.CARD\_INST\_ID)

WHERE

NE.TID IS NOT NULL

AND REGEXP\_LIKE(EP.PORT\_HUM\_ID, '-')

AND UPPER(CPI.STATUS) = 'LIVE'

ORDER BY

TID,

PORT\_HUM\_ID;

COMMIT;

---------------------------------------------------------------

-- Insert ports where a "live" port does not already exist --

---------------------------------------------------------------

INSERT INTO NETSMART\_XNG\_PORTS

SELECT DISTINCT

NE.TID,

NE.TID\_TYPE,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-') SLOT,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '') PORT\_HUM\_ID,

EP.PORT\_ACCESS\_ID,

CPE.CIRC\_PATH\_INST\_ID,

LI.LEG\_INST\_ID,

CPE.MEMBER\_NBR,

CPI.A\_SIDE\_SITE\_ID,

S1.SITE\_HUM\_ID A\_SIDE\_SITE\_HUM\_ID,

CPI.Z\_SIDE\_SITE\_ID,

S2.SITE\_HUM\_ID Z\_SIDE\_SITE\_HUM\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.BANDWIDTH,

CPI.STATUS PATH\_STATUS,

CPI.NEXT\_PATH\_INST\_ID,

CPI.PREV\_PATH\_INST\_ID,

NE.NE\_INST\_ID,

NE.EQUIP\_INST\_ID,

NE.PARENT\_EQ\_INST\_ID,

NE.SITE\_INST\_ID,

NE.VENDOR,

NE.MODEL,

NE.DESCR,

EP.PORT\_INST\_ID,

EP.CARD\_INST\_ID

FROM

NETSMART\_FUJITSU\_EQUIPMENT NE

JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.EQUIP\_INST\_ID OR EQ.PARENT\_EQ\_INST\_ID = NE.EQUIP\_INST\_ID)

JOIN

VZWNET.EPA EP ON (EP.EQUIP\_INST\_ID = EQ.EQUIP\_INST\_ID)

JOIN

VZWNET.CIRC\_PATH\_ELEMENT CPE ON CPE.PORT\_INST\_ID = EP.PORT\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_INST LI ON LI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_MEMBER PLM ON PLM.LEG\_INST\_ID = LI.LEG\_INST\_ID AND PLM.ELEMENT\_INST\_ID = CPE.ELEMENT\_INST\_ID

JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.SITE\_INST S1 ON S1.SITE\_INST\_ID = CPI.A\_SIDE\_SITE\_ID

JOIN

VZWNET.SITE\_INST S2 ON S2.SITE\_INST\_ID = CPI.Z\_SIDE\_SITE\_ID

JOIN

VZWNET.CARD\_INST CRD ON (CRD.CARD\_INST\_ID = EP.CARD\_INST\_ID)

WHERE

NE.TID IS NOT NULL

AND REGEXP\_LIKE(EP.PORT\_HUM\_ID, '-')

AND UPPER(CPI.STATUS) != 'LIVE'

AND NOT EXISTS (SELECT 1

FROM

XNG\_REPORTS.NETSMART\_XNG\_PORTS NXP

WHERE

NXP.TID = NE.TID

--AND NXP.SLOT = REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-')

AND NXP.PORT\_HUM\_ID = REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '')

)

ORDER BY

TID,

PORT\_HUM\_ID;

COMMIT;

--DELETE FROM NETSMART\_XCONNECT\_AUDIT\_WK;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_XCONNECT\_AUDIT\_WK';

INSERT INTO NETSMART\_XCONNECT\_AUDIT\_WK

SELECT

NE.TID,

NE.TID\_TYPE,

NULL MATCH\_CODE,

NULL MATCH\_STATUS,

XC.XCONNECT\_ASIDE,

XP1.NE\_INST\_ID A\_SIDE\_EQUIP\_ID,

XP1.CIRC\_PATH\_INST\_ID A\_SIDE\_PATH\_INST\_ID,

XP1.LEG\_INST\_ID A\_SIDE\_LEG\_INST\_ID,

XP1.MEMBER\_NBR A\_SIDE\_MEMBER\_NBR,

XC.XCONNECT\_ZSIDE,

XP2.NE\_INST\_ID Z\_SIDE\_EQUIP\_ID,

XP1.CIRC\_PATH\_INST\_ID Z\_SIDE\_PATH\_INST\_ID,

XP2.LEG\_INST\_ID A\_SIDE\_LEG\_INST\_ID,

XP2.MEMBER\_NBR Z\_SIDE\_MEMBER\_NBR,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.CIRC\_PATH\_INST\_ID

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.CIRC\_PATH\_INST\_ID

ELSE NULL

END CIRC\_PATH\_INST\_ID,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.CIRC\_PATH\_HUM\_ID

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.CIRC\_PATH\_HUM\_ID

ELSE NULL

END CIRC\_PATH\_HUM\_ID,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.PATH\_STATUS

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.PATH\_STATUS

ELSE NULL

END PATH\_STATUS,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.BANDWIDTH

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.BANDWIDTH

ELSE NULL

END BANDWIDTH

FROM

NETSMART\_SNCNE NE

JOIN

NETSMART\_XCONNECT XC ON XC.SNCNE\_ID = NE.INST\_ID

LEFT OUTER JOIN

NETSMART\_XNG\_PORTS XP1 ON XP1.TID = NE.TID AND (((XP1.PORT\_HUM\_ID = XC.XCONNECT\_ASIDE) OR (XP1.SLOT || '-' || XP1.PORT\_HUM\_ID = XC.XCONNECT\_ASIDE)) OR

(XP1.PORT\_ACCESS\_ID = XC.XCONNECT\_ASIDE))

LEFT OUTER JOIN

NETSMART\_XNG\_PORTS XP2 ON XP2.TID = NE.TID AND (((XP2.PORT\_HUM\_ID = XC.XCONNECT\_ZSIDE) OR (XP2.SLOT || '-' || XP2.PORT\_HUM\_ID = XC.XCONNECT\_ZSIDE)) OR

(XP2.PORT\_ACCESS\_ID = XC.XCONNECT\_ZSIDE))

WHERE

REGEXP\_LIKE(NE.TID,'^[[:alnum:]]{8}MS')

-- WHERE

-- XP1.A\_SIDE\_SITE\_ID = XP2.Z\_SIDE\_SITE\_ID

-- REGEXP\_LIKE(NE.TID\_TYPE, '9500')

ORDER BY

NE.TID,

XCONNECT\_ASIDE,

XCONNECT\_ZSIDE;

COMMIT;

-- Mark the audit rows where neither side was found

UPDATE

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT\_WK

SET

MATCH\_CODE = 'DISCREPANCY',

MATCH\_STATUS = 'Neither of the two cross connected ports have an associated path in Granite.'

WHERE

A\_SIDE\_LEG\_INST\_ID IS NULL AND Z\_SIDE\_LEG\_INST\_ID IS NULL;

COMMIT;

-- Mark the audit rows where A side was found but the Z side wasn't

UPDATE

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'One of the cross connected ports is missing from the path in Granite. ' || XC.XCONNECT\_ZSIDE || ' not found in the circuit path. Remove cross connect from the MSPP or correct circuit path elements in Granite.'

WHERE

XC.MATCH\_STATUS IS NULL AND

A\_SIDE\_LEG\_INST\_ID IS NOT NULL AND Z\_SIDE\_LEG\_INST\_ID IS NULL;

COMMIT;

-- Mark the audit rows where Z side was found but the A side wasn't

UPDATE

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'One of the cross connected ports is missing from the path in Granite. ' || XC.XCONNECT\_ASIDE || ' not found in the circuit path. Remove cross connect from the MSPP or correct circuit path elements in Granite.'

WHERE

XC.MATCH\_STATUS IS NULL AND

A\_SIDE\_LEG\_INST\_ID IS NULL AND Z\_SIDE\_LEG\_INST\_ID IS NOT NULL;

COMMIT;

-- Mark the rows where the path\_inst ids do not match

UPDATE

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The A side port and Z side port of the cross connect do not belong to the same Circuit Path.'

WHERE

XC.MATCH\_CODE IS NULL

AND A\_SIDE\_PATH\_INST\_ID != Z\_SIDE\_PATH\_INST\_ID;

COMMIT;

-- Mark the rows where the leg\_inst ids do not match

UPDATE

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The A side port and Z side port are not endpoints of a single circuit leg.'

WHERE

XC.MATCH\_CODE IS NULL

AND A\_SIDE\_LEG\_INST\_ID != Z\_SIDE\_LEG\_INST\_ID;

COMMIT;

-- Mark the rows where the leg\_inst match but the member numbers do not match

UPDATE

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The member number of the A side port does not match the member number of the Z side port.'

WHERE

A\_SIDE\_LEG\_INST\_ID = Z\_SIDE\_LEG\_INST\_ID

AND ((A\_SIDE\_MEMBER\_NBR != Z\_SIDE\_MEMBER\_NBR

OR (A\_SIDE\_MEMBER\_NBR IS NULL AND Z\_SIDE\_MEMBER\_NBR IS NOT NULL)

or (A\_SIDE\_MEMBER\_NBR IS NOT NULL AND Z\_SIDE\_MEMBER\_NBR IS NULL)));

COMMIT;

-- Mark the rest of the Audit records as passed audit

UPDATE

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'MATCH',

XC.MATCH\_STATUS = ''

WHERE

A\_SIDE\_LEG\_INST\_ID = Z\_SIDE\_LEG\_INST\_ID

AND (A\_SIDE\_MEMBER\_NBR = Z\_SIDE\_MEMBER\_NBR or (A\_SIDE\_MEMBER\_NBR IS NULL AND Z\_SIDE\_MEMBER\_NBR IS NULL));

COMMIT;

END AUDIT\_XCONNECTS;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Audit\_ADM\_XConnects \*

\* \*

\* Purpose: Audit FlashWave cross connects against XNG \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE AUDIT\_ADM\_XCONNECTS IS

BEGIN

--DELETE FROM NETSMART\_FUJITSU\_EQUIPMENT;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_FUJITSU\_EQUIPMENT';

--

-- Don't need to regenerate NetSmart\_XConnect\_Equipment table since it's

-- not based on any ADM tables.

--

-- INSERT INTO NETSMART\_XCONNECT\_EQUIPMENT

-- SELECT

-- LEVEL NE\_LEVEL,

-- TO\_NUMBER

-- (TRIM

-- (BOTH '/' FROM SYS\_CONNECT\_BY\_PATH(

-- DECODE (LEVEL, 1, EI.EQUIP\_INST\_ID, '' ),

-- '/'

-- )

-- )

-- ) NE\_INST\_ID,

-- EI.EQUIP\_INST\_ID,

-- EI.SITE\_INST\_ID,

-- EI.PARENT\_EQ\_INST\_ID,

-- DECODE (LEVEL,

-- 1, 'NE',

-- DECODE (EI.EQ\_CLASS, 'S', 'SHELF', 'N/A')

-- ) EQ\_CLASS\_TYPE,

-- EI.DESCR,

-- EI.MODEL,

-- EI.TYPE,

-- EI.STATUS,

-- EI.VENDOR,

-- EI.EQ\_CLASS

-- FROM

-- VZWNET.EQUIP\_INST EI

-- WHERE

-- (LEVEL = 1 OR EI.EQ\_CLASS = 'S')

-- START WITH

-- EI.VENDOR = 'FUJITSU'

-- AND REGEXP\_LIKE (MODEL, 'FLASHWAVE|FW')

-- CONNECT BY

-- PRIOR EI.EQUIP\_INST\_ID = EI.PARENT\_EQ\_INST\_ID;

--

-- COMMIT;

INSERT INTO NETSMART\_ADM\_FUJITSU\_EQUIPMENT

SELECT

NE.TID,

NE.TID\_TYPE,

NXE.NE\_LEVEL,

NXE.NE\_INST\_ID,

NXE.EQUIP\_INST\_ID,

NXE.SITE\_INST\_ID,

NXE.PARENT\_EQ\_INST\_ID,

NXE.EQ\_CLASS\_TYPE,

NXE.DESCR,

NXE.MODEL,

NXE.TYPE,

NXE.STATUS,

NXE.VENDOR,

NXE.EQ\_CLASS

FROM

NETSMART\_XCONNECT\_EQUIPMENT NXE

JOIN

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK NE ON (NE.NE\_INST\_ID = NXE.NE\_INST\_ID)

WHERE

NE.TID IS NOT NULL;

COMMIT;

UPDATE NETSMART\_ADM\_FUJITSU\_EQUIPMENT

SET PARENT\_EQ\_INST\_ID = NULL

WHERE EQUIP\_INST\_ID IN

(SELECT

NE2.EQUIP\_INST\_ID

FROM

NETSMART\_ADM\_FUJITSU\_EQUIPMENT NE2

WHERE

NE2.PARENT\_EQ\_INST\_ID IS NOT NULL

AND NOT EXISTS (SELECT

NE.EQUIP\_INST\_ID

FROM

NETSMART\_ADM\_FUJITSU\_EQUIPMENT NE

WHERE

NE.EQUIP\_INST\_ID = NE2.PARENT\_EQ\_INST\_ID

)

);

COMMIT;

--DELETE FROM NETSMART\_XNG\_PORTS;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_XNG\_PORTS';

INSERT INTO NETSMART\_ADM\_XNG\_PORTS

SELECT DISTINCT

NE.TID,

NE.TID\_TYPE,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-') SLOT,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '') PORT\_HUM\_ID,

EP.PORT\_ACCESS\_ID,

CPE.CIRC\_PATH\_INST\_ID,

LI.LEG\_INST\_ID,

CPE.MEMBER\_NBR,

CPI.A\_SIDE\_SITE\_ID,

S1.SITE\_HUM\_ID A\_SIDE\_SITE\_HUM\_ID,

CPI.Z\_SIDE\_SITE\_ID,

S2.SITE\_HUM\_ID Z\_SIDE\_SITE\_HUM\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.BANDWIDTH,

CPI.STATUS PATH\_STATUS,

CPI.NEXT\_PATH\_INST\_ID,

CPI.PREV\_PATH\_INST\_ID,

NE.NE\_INST\_ID,

NE.EQUIP\_INST\_ID,

NE.PARENT\_EQ\_INST\_ID,

NE.SITE\_INST\_ID,

NE.VENDOR,

NE.MODEL,

NE.DESCR,

EP.PORT\_INST\_ID,

EP.CARD\_INST\_ID

FROM

NETSMART\_ADM\_FUJITSU\_EQUIPMENT NE

JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.EQUIP\_INST\_ID)

JOIN

VZWNET.EPA EP ON (EP.EQUIP\_INST\_ID = EQ.EQUIP\_INST\_ID)

JOIN

VZWNET.CIRC\_PATH\_ELEMENT CPE ON CPE.PORT\_INST\_ID = EP.PORT\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_INST LI ON LI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_MEMBER PLM ON PLM.LEG\_INST\_ID = LI.LEG\_INST\_ID AND PLM.ELEMENT\_INST\_ID = CPE.ELEMENT\_INST\_ID

JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.SITE\_INST S1 ON S1.SITE\_INST\_ID = CPI.A\_SIDE\_SITE\_ID

JOIN

VZWNET.SITE\_INST S2 ON S2.SITE\_INST\_ID = CPI.Z\_SIDE\_SITE\_ID

JOIN

VZWNET.CARD\_INST CRD ON (CRD.CARD\_INST\_ID = EP.CARD\_INST\_ID)

WHERE

NE.TID IS NOT NULL

AND REGEXP\_LIKE(EP.PORT\_HUM\_ID, '-')

AND UPPER(CPI.STATUS) = 'LIVE'

ORDER BY

TID,

PORT\_HUM\_ID;

COMMIT;

---------------------------------------------------------------

-- Insert ports where a "live" port does not already exist --

---------------------------------------------------------------

INSERT INTO NETSMART\_ADM\_XNG\_PORTS

SELECT DISTINCT

NE.TID,

NE.TID\_TYPE,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-') SLOT,

REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '') PORT\_HUM\_ID,

EP.PORT\_ACCESS\_ID,

CPE.CIRC\_PATH\_INST\_ID,

LI.LEG\_INST\_ID,

CPE.MEMBER\_NBR,

CPI.A\_SIDE\_SITE\_ID,

S1.SITE\_HUM\_ID A\_SIDE\_SITE\_HUM\_ID,

CPI.Z\_SIDE\_SITE\_ID,

S2.SITE\_HUM\_ID Z\_SIDE\_SITE\_HUM\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.BANDWIDTH,

CPI.STATUS PATH\_STATUS,

CPI.NEXT\_PATH\_INST\_ID,

CPI.PREV\_PATH\_INST\_ID,

NE.NE\_INST\_ID,

NE.EQUIP\_INST\_ID,

NE.PARENT\_EQ\_INST\_ID,

NE.SITE\_INST\_ID,

NE.VENDOR,

NE.MODEL,

NE.DESCR,

EP.PORT\_INST\_ID,

EP.CARD\_INST\_ID

FROM

NETSMART\_ADM\_FUJITSU\_EQUIPMENT NE

JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.EQUIP\_INST\_ID OR EQ.PARENT\_EQ\_INST\_ID = NE.EQUIP\_INST\_ID)

JOIN

VZWNET.EPA EP ON (EP.EQUIP\_INST\_ID = EQ.EQUIP\_INST\_ID)

JOIN

VZWNET.CIRC\_PATH\_ELEMENT CPE ON CPE.PORT\_INST\_ID = EP.PORT\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_INST LI ON LI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.PATH\_LEG\_MEMBER PLM ON PLM.LEG\_INST\_ID = LI.LEG\_INST\_ID AND PLM.ELEMENT\_INST\_ID = CPE.ELEMENT\_INST\_ID

JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

JOIN

VZWNET.SITE\_INST S1 ON S1.SITE\_INST\_ID = CPI.A\_SIDE\_SITE\_ID

JOIN

VZWNET.SITE\_INST S2 ON S2.SITE\_INST\_ID = CPI.Z\_SIDE\_SITE\_ID

JOIN

VZWNET.CARD\_INST CRD ON (CRD.CARD\_INST\_ID = EP.CARD\_INST\_ID)

WHERE

NE.TID IS NOT NULL

AND REGEXP\_LIKE(EP.PORT\_HUM\_ID, '-')

AND UPPER(CPI.STATUS) != 'LIVE'

AND NOT EXISTS (SELECT 1

FROM

XNG\_REPORTS.NETSMART\_ADM\_XNG\_PORTS NXP

WHERE

NXP.TID = NE.TID

--AND NXP.SLOT = REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(CRD.SLOT, '-0+', '-'), '^0+', ''), '^-', '0-')

AND NXP.PORT\_HUM\_ID = REGEXP\_REPLACE(REGEXP\_REPLACE(REGEXP\_REPLACE(EP.PORT\_HUM\_ID, ':', '-'), '-0+', '-'), '^0+', '')

)

ORDER BY

TID,

PORT\_HUM\_ID;

COMMIT;

--DELETE FROM NETSMART\_XCONNECT\_AUDIT\_WK;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_XCONNECT\_AUDIT\_WK';

INSERT INTO NETSMART\_ADM\_XCONNECT\_AUDIT\_WK

SELECT

NE.TID,

NE.TID\_TYPE,

NULL MATCH\_CODE,

NULL MATCH\_STATUS,

XC.XCONNECT\_ASIDE,

XP1.NE\_INST\_ID A\_SIDE\_EQUIP\_ID,

XP1.CIRC\_PATH\_INST\_ID A\_SIDE\_PATH\_INST\_ID,

XP1.LEG\_INST\_ID A\_SIDE\_LEG\_INST\_ID,

XP1.MEMBER\_NBR A\_SIDE\_MEMBER\_NBR,

XC.XCONNECT\_ZSIDE,

XP2.NE\_INST\_ID Z\_SIDE\_EQUIP\_ID,

XP1.CIRC\_PATH\_INST\_ID Z\_SIDE\_PATH\_INST\_ID,

XP2.LEG\_INST\_ID A\_SIDE\_LEG\_INST\_ID,

XP2.MEMBER\_NBR Z\_SIDE\_MEMBER\_NBR,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.CIRC\_PATH\_INST\_ID

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.CIRC\_PATH\_INST\_ID

ELSE NULL

END CIRC\_PATH\_INST\_ID,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.CIRC\_PATH\_HUM\_ID

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.CIRC\_PATH\_HUM\_ID

ELSE NULL

END CIRC\_PATH\_HUM\_ID,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.PATH\_STATUS

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.PATH\_STATUS

ELSE NULL

END PATH\_STATUS,

CASE

WHEN XP1.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP1.BANDWIDTH

WHEN XP2.CIRC\_PATH\_INST\_ID IS NOT NULL

THEN XP2.BANDWIDTH

ELSE NULL

END BANDWIDTH

FROM

NETSMART\_SNCNE NE

JOIN

NETSMART\_XCONNECT XC ON XC.SNCNE\_ID = NE.INST\_ID

LEFT OUTER JOIN

NETSMART\_ADM\_XNG\_PORTS XP1 ON XP1.TID = NE.TID AND (((XP1.PORT\_HUM\_ID = XC.XCONNECT\_ASIDE) OR (XP1.SLOT || '-' || XP1.PORT\_HUM\_ID = XC.XCONNECT\_ASIDE)) OR

(XP1.PORT\_ACCESS\_ID = XC.XCONNECT\_ASIDE))

LEFT OUTER JOIN

NETSMART\_ADM\_XNG\_PORTS XP2 ON XP2.TID = NE.TID AND (((XP2.PORT\_HUM\_ID = XC.XCONNECT\_ZSIDE) OR (XP2.SLOT || '-' || XP2.PORT\_HUM\_ID = XC.XCONNECT\_ZSIDE)) OR

(XP2.PORT\_ACCESS\_ID = XC.XCONNECT\_ZSIDE))

WHERE

NOT REGEXP\_LIKE(NE.TID,'^[[:alnum:]]{8}MS')

-- WHERE

-- XP1.A\_SIDE\_SITE\_ID = XP2.Z\_SIDE\_SITE\_ID

-- REGEXP\_LIKE(NE.TID\_TYPE, '9500')

ORDER BY

NE.TID,

XCONNECT\_ASIDE,

XCONNECT\_ZSIDE;

COMMIT;

-- Mark the audit rows where neither side was found

UPDATE

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT\_WK

SET

MATCH\_CODE = 'DISCREPANCY',

MATCH\_STATUS = 'Neither of the two cross connected ports have an associated path in Granite.'

WHERE

A\_SIDE\_LEG\_INST\_ID IS NULL AND Z\_SIDE\_LEG\_INST\_ID IS NULL;

COMMIT;

-- Mark the audit rows where A side was found but the Z side wasn't

UPDATE

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'One of the cross connected ports is missing from the path in Granite. ' || XC.XCONNECT\_ZSIDE || ' not found in the circuit path. Remove cross connect from the MSPP or correct circuit path elements in Granite.'

WHERE

XC.MATCH\_STATUS IS NULL AND

A\_SIDE\_LEG\_INST\_ID IS NOT NULL AND Z\_SIDE\_LEG\_INST\_ID IS NULL;

COMMIT;

-- Mark the audit rows where Z side was found but the A side wasn't

UPDATE

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'One of the cross connected ports is missing from the path in Granite. ' || XC.XCONNECT\_ASIDE || ' not found in the circuit path. Remove cross connect from the MSPP or correct circuit path elements in Granite.'

WHERE

XC.MATCH\_STATUS IS NULL AND

A\_SIDE\_LEG\_INST\_ID IS NULL AND Z\_SIDE\_LEG\_INST\_ID IS NOT NULL;

COMMIT;

-- Mark the rows where the path\_inst ids do not match

UPDATE

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The A side port and Z side port of the cross connect do not belong to the same Circuit Path.'

WHERE

XC.MATCH\_CODE IS NULL

AND A\_SIDE\_PATH\_INST\_ID != Z\_SIDE\_PATH\_INST\_ID;

COMMIT;

-- Mark the rows where the leg\_inst ids do not match

UPDATE

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The A side port and Z side port are not endpoints of a single circuit leg.'

WHERE

XC.MATCH\_CODE IS NULL

AND A\_SIDE\_LEG\_INST\_ID != Z\_SIDE\_LEG\_INST\_ID;

COMMIT;

-- Mark the rows where the leg\_inst match but the member numbers do not match

UPDATE

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'DISCREPANCY',

XC.MATCH\_STATUS = 'The member number of the A side port does not match the member number of the Z side port.'

WHERE

A\_SIDE\_LEG\_INST\_ID = Z\_SIDE\_LEG\_INST\_ID

AND ((A\_SIDE\_MEMBER\_NBR != Z\_SIDE\_MEMBER\_NBR

OR (A\_SIDE\_MEMBER\_NBR IS NULL AND Z\_SIDE\_MEMBER\_NBR IS NOT NULL)

or (A\_SIDE\_MEMBER\_NBR IS NOT NULL AND Z\_SIDE\_MEMBER\_NBR IS NULL)));

COMMIT;

-- Mark the rest of the Audit records as passed audit

UPDATE

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT\_WK XC

SET

XC.MATCH\_CODE = 'MATCH',

XC.MATCH\_STATUS = ''

WHERE

A\_SIDE\_LEG\_INST\_ID = Z\_SIDE\_LEG\_INST\_ID

AND (A\_SIDE\_MEMBER\_NBR = Z\_SIDE\_MEMBER\_NBR or (A\_SIDE\_MEMBER\_NBR IS NULL AND Z\_SIDE\_MEMBER\_NBR IS NULL));

COMMIT;

END AUDIT\_ADM\_XCONNECTS;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Function NS\_Summary\_Tbl \*

\* \*

\* Purpose: Outputs NetSmart MSPP Summary Report Table \*

\* \*

\* Inputs: None \*

\* Outputs: NetSmart\_Summary\_Tbl \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_SUMMARY\_TBL RETURN NETSMART\_SUMMARY\_TBL PIPELINED IS

OUT\_REC NETSMART\_SUMMARY\_REC := NETSMART\_SUMMARY\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR NETSMART\_CUR IS

SELECT

AREA,

REGION,

TOTAL\_NE,

DUPLICATE\_NE,

MATCHED\_NE,

PERCENT\_MATCHED,

PASSED\_AUDIT,

NAMING\_COMPLIANCE,

DISCOVERED\_XCONNECTS,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

TOTAL\_COMPLIANCE

FROM

NETSMART\_EQ\_SUMM\_REGION;

BEGIN

OPEN NETSMART\_CUR;

LOOP

FETCH NETSMART\_CUR INTO OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.TOTAL\_NE, OUT\_REC.DUPLICATE\_NE, OUT\_REC.MATCHED\_NE, OUT\_REC.PERCENT\_MATCHED, OUT\_REC.PASSED\_AUDIT,

OUT\_REC.NAMING\_COMPLIANCE, OUT\_REC.DISCOVERED\_XCONNECTS, OUT\_REC.LIVE\_XCONNECTS, OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.TOTAL\_COMPLIANCE;

EXIT WHEN NETSMART\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE NETSMART\_CUR;

RETURN;

END NS\_SUMMARY\_TBL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Function NS\_Summary\_Tbl \*

\* \*

\* Purpose: Outputs NetSmart ADM Summary Report Table \*

\* \*

\* Inputs: None \*

\* Outputs: NetSmart\_Summary\_Tbl \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_ADM\_SUMMARY\_TBL RETURN NETSMART\_SUMMARY\_TBL PIPELINED IS

OUT\_REC NETSMART\_SUMMARY\_REC := NETSMART\_SUMMARY\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR NETSMART\_CUR IS

SELECT

AREA,

REGION,

TOTAL\_NE,

DUPLICATE\_NE,

MATCHED\_NE,

PERCENT\_MATCHED,

PASSED\_AUDIT,

NAMING\_COMPLIANCE,

DISCOVERED\_XCONNECTS,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

TOTAL\_COMPLIANCE

FROM

NETSMART\_ADM\_EQ\_SUMM\_REGION;

BEGIN

OPEN NETSMART\_CUR;

LOOP

FETCH NETSMART\_CUR INTO OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.TOTAL\_NE, OUT\_REC.DUPLICATE\_NE, OUT\_REC.MATCHED\_NE, OUT\_REC.PERCENT\_MATCHED, OUT\_REC.PASSED\_AUDIT,

OUT\_REC.NAMING\_COMPLIANCE, OUT\_REC.DISCOVERED\_XCONNECTS, OUT\_REC.LIVE\_XCONNECTS, OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.TOTAL\_COMPLIANCE;

EXIT WHEN NETSMART\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE NETSMART\_CUR;

RETURN;

END NS\_ADM\_SUMMARY\_TBL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function NS\_DETAIL\_TBL \*

\* \*

\* Parameters \*

\* REGION\_IN Region Name (from CLLI\_DOMAIN\_MAP) \*

\* use NULL for all regions. \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_DETAIL\_TBL(REGION\_IN IN VARCHAR2, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_DETAIL\_TBL PIPELINED IS

OUT\_REC NETSMART\_DETAIL\_REC := NETSMART\_DETAIL\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR ALL\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY AREA, REGION, LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION IS NOT NULL

GROUP BY

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

UNION

SELECT

'unknown' AREA,

'unknown' REGION,

NULL LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION IS NULL

GROUP BY

'unknown',

'unknown',

NULL,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

)

ORDER BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR UNKNOWN\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (SELECT

'unknown' AREA,

'unknown' REGION,

'UNKNOWN\_DOMAIN' LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION IS NULL

GROUP BY

'unknown',

'unknown',

'UNKNOWN\_DOMAIN',

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

ORDER BY

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR REGION\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION = REGION\_IN

GROUP BY

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

ORDER BY

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

/\* CURSOR NNO\_CUR IS

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (SELECT

'NNO' AREA,

'NNO' REGION,

'NNO\_DOMAIN' LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

XNG\_REPORTS.CLLI\_DOMAIN\_MAP\_V CDM ON ((SUBSTR(CDM.CLLI,1,6) = SUBSTR(NE.TID,1,6)) AND (UPPER(CDM.REGION) != 'NNO'))

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

-- UPPER(REGION) = 'NNO' OR

SUBSTR(RPAD(NE.TID, 16), 12, 2) = '-N'

GROUP BY

'NNO',

'NNO',

'NNO\_DOMAIN',

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

ORDER BY

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER; \*/

BEGIN

IF (REGION\_IN IS NULL) THEN

OPEN ALL\_CUR;

LOOP

FETCH ALL\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN ALL\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE ALL\_CUR;

/\* ELSIF (UPPER(REGION\_IN) = 'NNO') THEN

OPEN NNO\_CUR;

LOOP

FETCH NNO\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN NNO\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE NNO\_CUR; \*/

ELSIF (UPPER(REGION\_IN) = 'UNKNOWN') THEN

OPEN UNKNOWN\_CUR;

LOOP

FETCH UNKNOWN\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN UNKNOWN\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE UNKNOWN\_CUR;

ELSE

OPEN REGION\_CUR;

LOOP

FETCH REGION\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN REGION\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE REGION\_CUR;

END IF ;

RETURN;

END NS\_DETAIL\_TBL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function NS\_ADM\_DETAIL\_TBL \*

\* \*

\* Parameters \*

\* REGION\_IN Region Name (from CLLI\_DOMAIN\_MAP) \*

\* use NULL for all regions. \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_ADM\_DETAIL\_TBL(REGION\_IN IN VARCHAR2, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_DETAIL\_TBL PIPELINED IS

OUT\_REC NETSMART\_DETAIL\_REC := NETSMART\_DETAIL\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR ALL\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY AREA, REGION, LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_ADM\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION IS NOT NULL

GROUP BY

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

UNION

SELECT

'unknown' AREA,

'unknown' REGION,

NULL LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_ADM\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION IS NULL

GROUP BY

'unknown',

'unknown',

NULL,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

)

ORDER BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR UNKNOWN\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (SELECT

'unknown' AREA,

'unknown' REGION,

'UNKNOWN\_DOMAIN' LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_ADM\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION IS NULL

GROUP BY

'unknown',

'unknown',

'UNKNOWN\_DOMAIN',

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

ORDER BY

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR REGION\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT NE

LEFT OUTER JOIN

NETSMART\_ADM\_XCONNECT\_AUDIT XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION = REGION\_IN

GROUP BY

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

ORDER BY

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

IF (REGION\_IN IS NULL) THEN

OPEN ALL\_CUR;

LOOP

FETCH ALL\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN ALL\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE ALL\_CUR;

ELSIF (UPPER(REGION\_IN) = 'UNKNOWN') THEN

OPEN UNKNOWN\_CUR;

LOOP

FETCH UNKNOWN\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN UNKNOWN\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE UNKNOWN\_CUR;

ELSE

OPEN REGION\_CUR;

LOOP

FETCH REGION\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN REGION\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE REGION\_CUR;

END IF ;

RETURN;

END NS\_ADM\_DETAIL\_TBL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function NS\_DETAIL\_TBL\_WRK (Internal) \*

\* \*

\* Parameters \*

\* REGION\_IN Region Name (from CLLI\_DOMAIN\_MAP) \*

\* use NULL for all regions. \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_DETAIL\_TBL\_WRK(START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_DETAIL\_TBL PIPELINED IS

OUT\_REC NETSMART\_DETAIL\_REC := NETSMART\_DETAIL\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR ALL\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY AREA, REGION, LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (

SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NE

LEFT OUTER JOIN

NETSMART\_XCONNECT\_AUDIT\_WK XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

NE.LEAF\_DOMAIN\_NAME IS NOT NULL

/\* AND UPPER(REGION) <> 'NNO' \*/

GROUP BY

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

UNION

SELECT

'unknown' AREA,

'unknown' REGION,

NULL LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_NE\_VS\_XNG\_AUDIT\_WK NE

LEFT OUTER JOIN

NETSMART\_XCONNECT\_AUDIT\_WK XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON ((CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME) AND (UPPER(CDM.REGION) != 'NNO'))

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

/\*REGION IS NULL

AND (NOT SUBSTR(RPAD(NE.TID, 16), 12, 2) = '-N' OR NE.TID IS NULL)\*/

NE.LEAF\_DOMAIN\_NAME IS NULL

GROUP BY

'unknown',

'unknown',

NULL,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

)

ORDER BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

OPEN ALL\_CUR;

LOOP

FETCH ALL\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN ALL\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE ALL\_CUR;

RETURN;

END NS\_DETAIL\_TBL\_WRK;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function NS\_ADM\_DETAIL\_TBL\_WRK (Internal) \*

\* \*

\* Parameters \*

\* REGION\_IN Region Name (from CLLI\_DOMAIN\_MAP) \*

\* use NULL for all regions. \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_ADM\_DETAIL\_TBL\_WRK(START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_DETAIL\_TBL PIPELINED IS

OUT\_REC NETSMART\_DETAIL\_REC := NETSMART\_DETAIL\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR ALL\_CUR IS

WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY AREA, REGION, LEAF\_DOMAIN\_NAME, TID, DESCR)) ROWNUMBER,

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

TID\_TYPE,

LOCATION,

CONTACTS,

IP\_ADDR,

MATCH\_CODE,

MATCH\_STATUS,

NE\_INST\_ID,

DESCR,

SITE\_INST\_ID,

SITE\_HUM\_ID,

TYPE,

VENDOR,

MODEL,

NUM\_XCONNECT,

DISCREPANCY,

LIVE\_XCONNECTS,

OTHER\_XCONNECTS,

COMPLIANCE

FROM (

SELECT

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK NE

LEFT OUTER JOIN

NETSMART\_ADM\_XCONNECT\_AUDIT\_WK XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

NE.LEAF\_DOMAIN\_NAME IS NOT NULL

/\* AND UPPER(REGION) <> 'NNO' \*/

GROUP BY

CDM.AREA,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

UNION

SELECT

'unknown' AREA,

'unknown' REGION,

NULL LEAF\_DOMAIN\_NAME,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) NUM\_XCONNECT,

SUM(DECODE(CASE WHEN XC.XCONNECT\_ASIDE IS NOT NULL THEN XC.XCONNECT\_ASIDE||','||XC.XCONNECT\_ZSIDE ELSE NULL END, NULL, 0, 1)) - SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) DISCREPANCY,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) = 'LIVE'

THEN 1

ELSE 0

END) LIVE\_XCONNECTS,

SUM(CASE WHEN UPPER(TRIM(XC.MATCH\_CODE)) = 'MATCH' AND UPPER(XC.PATH\_STATUS) != 'LIVE'

THEN 1

ELSE 0

END) OTHER\_XCONNECTS,

CASE

WHEN SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1)) != 0

THEN TO\_CHAR(ROUND((SUM(DECODE(UPPER(XC.MATCH\_CODE), 'MATCH', 1, 0)) / SUM(DECODE(XC.XCONNECT\_ASIDE, NULL, 0, 1))) \* 100,2), '990.00')

ELSE

TO\_CHAR(0,'990.00')

END COMPLIANCE

FROM

NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK NE

LEFT OUTER JOIN

NETSMART\_ADM\_XCONNECT\_AUDIT\_WK XC ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON ((CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME) AND (UPPER(CDM.REGION) != 'NNO'))

LEFT OUTER JOIN

VZWNET.EQUIP\_INST EQ ON (EQ.EQUIP\_INST\_ID = NE.NE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.SITE\_INST S ON (S.SITE\_INST\_ID = EQ.SITE\_INST\_ID)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

/\*REGION IS NULL

AND (NOT SUBSTR(RPAD(NE.TID, 16), 12, 2) = '-N' OR NE.TID IS NULL)\*/

NE.LEAF\_DOMAIN\_NAME IS NULL

GROUP BY

'unknown',

'unknown',

NULL,

NE.TID,

NE.TID\_TYPE,

NE.LOCATION,

NE.CONTACTS,

NE.IP\_ADDR,

NE.MATCH\_CODE,

NE.MATCH\_STATUS,

NE.NE\_INST\_ID,

EQ.DESCR,

EQ.SITE\_INST\_ID,

S.SITE\_HUM\_ID,

EQ.TYPE,

EQ.VENDOR,

EQ.MODEL

)

ORDER BY

AREA,

REGION,

LEAF\_DOMAIN\_NAME,

TID,

DESCR

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

OPEN ALL\_CUR;

LOOP

FETCH ALL\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.AREA, OUT\_REC.REGION, OUT\_REC.LEAF\_DOMAIN\_NAME, OUT\_REC.TID, OUT\_REC.TID\_TYPE, OUT\_REC.LOCATION, OUT\_REC.CONTACTS, OUT\_REC.IP\_ADDR, OUT\_REC.MATCH\_CODE, OUT\_REC.MATCH\_STATUS, OUT\_REC.NE\_INST\_ID,

OUT\_REC.DESCR, OUT\_REC.SITE\_INST\_ID, OUT\_REC.SITE\_HUM\_ID, OUT\_REC.TYPE, OUT\_REC.VENDOR, OUT\_REC.MODEL, OUT\_REC.NUM\_XCONNECT, OUT\_REC.DISCREPANCY, OUT\_REC.LIVE\_XCONNECTS,

OUT\_REC.OTHER\_XCONNECTS, OUT\_REC.COMPLIANCE;

EXIT WHEN ALL\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE ALL\_CUR;

RETURN;

END NS\_ADM\_DETAIL\_TBL\_WRK;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function NS\_XConnect\_TBL \*

\* \*

\* Parameters \*

\* TID\_IN Target Id Name (use NULL for all Target Ids) \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_XCONNECT\_TBL(TID\_IN IN VARCHAR2, ONLY\_DISCREPANCIES\_IN IN NUMBER, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_XCONNECT\_TBL PIPELINED IS

OUT\_REC NETSMART\_XCONNECT\_REC := NETSMART\_XCONNECT\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR ALL\_CUR IS

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY TID,XC.XCONNECT\_ASIDE,XC.XCONNECT\_ZSIDE)) ROWNUMBER,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT XC

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID

/\* WHERE

XC.TID = TID\_IN \*/

ORDER BY

TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR DISC\_CUR IS

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY TID,XC.XCONNECT\_ASIDE,XC.XCONNECT\_ZSIDE)) ROWNUMBER,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT XC

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID

WHERE

XC.TID = TID\_IN

AND XC.MATCH\_CODE != 'MATCH'

ORDER BY

TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR TID\_CUR IS

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY TID,XC.XCONNECT\_ASIDE,XC.XCONNECT\_ZSIDE)) ROWNUMBER,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

XNG\_REPORTS.NETSMART\_XCONNECT\_AUDIT XC

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID

WHERE

XC.TID = TID\_IN

ORDER BY

TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

IF (TID\_IN IS NULL) THEN

OPEN ALL\_CUR;

LOOP

FETCH ALL\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN ALL\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE ALL\_CUR;

ELSIF (ONLY\_DISCREPANCIES\_IN = 1) THEN

OPEN DISC\_CUR;

LOOP

FETCH DISC\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN DISC\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE DISC\_CUR;

ELSE

OPEN TID\_CUR;

LOOP

FETCH TID\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN TID\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE TID\_CUR;

END IF ;

RETURN;

END NS\_XCONNECT\_TBL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function NS\_ADM\_XConnect\_TBL \*

\* \*

\* Parameters \*

\* TID\_IN Target Id Name (use NULL for all Target Ids) \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_ADM\_XCONNECT\_TBL(TID\_IN IN VARCHAR2, ONLY\_DISCREPANCIES\_IN IN NUMBER, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_XCONNECT\_TBL PIPELINED IS

OUT\_REC NETSMART\_XCONNECT\_REC := NETSMART\_XCONNECT\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR ALL\_CUR IS

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY TID,XC.XCONNECT\_ASIDE,XC.XCONNECT\_ZSIDE)) ROWNUMBER,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT XC

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID

/\* WHERE

XC.TID = TID\_IN \*/

ORDER BY

TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR DISC\_CUR IS

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY TID,XC.XCONNECT\_ASIDE,XC.XCONNECT\_ZSIDE)) ROWNUMBER,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT XC

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID

WHERE

XC.TID = TID\_IN

AND XC.MATCH\_CODE != 'MATCH'

ORDER BY

TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR TID\_CUR IS

SELECT

\*

FROM (SELECT

(ROW\_NUMBER () OVER (ORDER BY TID,XC.XCONNECT\_ASIDE,XC.XCONNECT\_ZSIDE)) ROWNUMBER,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

XNG\_REPORTS.NETSMART\_ADM\_XCONNECT\_AUDIT XC

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID

WHERE

XC.TID = TID\_IN

ORDER BY

TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

IF (TID\_IN IS NULL) THEN

OPEN ALL\_CUR;

LOOP

FETCH ALL\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN ALL\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE ALL\_CUR;

ELSIF (ONLY\_DISCREPANCIES\_IN = 1) THEN

OPEN DISC\_CUR;

LOOP

FETCH DISC\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN DISC\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE DISC\_CUR;

ELSE

OPEN TID\_CUR;

LOOP

FETCH TID\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN TID\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE TID\_CUR;

END IF ;

RETURN;

END NS\_ADM\_XCONNECT\_TBL;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function NS\_XCONNECT\_SUMMARY \*

\* \*

\* Parameters \*

\* REGION\_IN Region Name \*

\* ONLY\_DISCREPANCIES\_IN (1 = true, 0 = false) \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_XCONNECT\_SUMMARY(REGION\_IN IN VARCHAR2, ONLY\_DISCREPANCIES\_IN IN NUMBER, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_XCONNECT\_TBL PIPELINED IS

OUT\_REC NETSMART\_XCONNECT\_REC := NETSMART\_XCONNECT\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR REGION\_CUR IS

SELECT

ROWNUMBER,

TID,

DEVICE\_XCONNECTS,

AUDIT\_STATUS,

CIRC\_PATH\_INST\_ID,

CIRC\_PATH\_HUM\_ID,

PATH\_STATUS,

BANDWIDTH,

COMMENTS

FROM (WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

(ROW\_NUMBER () OVER (ORDER BY CDM.REGION, NE.LEAF\_DOMAIN\_NAME, XC.TID, XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE)) ROWNUMBER,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

NETSMART\_XCONNECT\_AUDIT XC

JOIN

NETSMART\_NE\_VS\_XNG\_AUDIT NE ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION = REGION\_IN

/\*AND XC.MATCH\_CODE <> 'MATCH'\*/

ORDER BY

REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR DISC\_CUR IS

SELECT

ROWNUMBER,

TID,

DEVICE\_XCONNECTS,

AUDIT\_STATUS,

CIRC\_PATH\_INST\_ID,

CIRC\_PATH\_HUM\_ID,

PATH\_STATUS,

BANDWIDTH,

COMMENTS

FROM (WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

(ROW\_NUMBER () OVER (ORDER BY CDM.REGION, NE.LEAF\_DOMAIN\_NAME, XC.TID, XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE)) ROWNUMBER,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

NETSMART\_XCONNECT\_AUDIT XC

JOIN

NETSMART\_NE\_VS\_XNG\_AUDIT NE ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION = REGION\_IN

AND XC.MATCH\_CODE <> 'MATCH'

ORDER BY

REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR UNKNOWN\_CUR IS

SELECT

ROWNUMBER,

TID,

DEVICE\_XCONNECTS,

AUDIT\_STATUS,

CIRC\_PATH\_INST\_ID,

CIRC\_PATH\_HUM\_ID,

PATH\_STATUS,

BANDWIDTH,

COMMENTS

FROM (

SELECT

(ROW\_NUMBER () OVER (ORDER BY 'unknown', 'unknown', XC.TID, XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE)) ROWNUMBER,

'unknown' REGION,

'UNKNOWN\_DOMAIN' LEAF\_DOMAIN\_NAME,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

NETSMART\_XCONNECT\_AUDIT XC

JOIN

NETSMART\_NE\_VS\_XNG\_AUDIT NE ON XC.TID = NE.TID

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

NE.LEAF\_DOMAIN\_NAME is NULL

/\*AND XC.MATCH\_CODE <> 'MATCH'\*/

ORDER BY

REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR UNKNOWN\_DISC\_CUR IS

SELECT

ROWNUMBER,

TID,

DEVICE\_XCONNECTS,

AUDIT\_STATUS,

CIRC\_PATH\_INST\_ID,

CIRC\_PATH\_HUM\_ID,

PATH\_STATUS,

BANDWIDTH,

COMMENTS

FROM (

SELECT

(ROW\_NUMBER () OVER (ORDER BY 'unknown', 'UNKNOWN\_DOMAIN', XC.TID, XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE)) ROWNUMBER,

'unknown' REGION,

'UNKNOWN\_DOMAIN' LEAF\_DOMAIN\_NAME,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

NETSMART\_XCONNECT\_AUDIT XC

JOIN

NETSMART\_NE\_VS\_XNG\_AUDIT NE ON XC.TID = NE.TID

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

NE.LEAF\_DOMAIN\_NAME is NULL

AND XC.MATCH\_CODE <> 'MATCH'

ORDER BY

REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

IF (ONLY\_DISCREPANCIES\_IN = 1) THEN

IF (REGION\_IN = 'Unknown') THEN

OPEN DISC\_CUR;

LOOP

FETCH UNKNOWN\_DISC\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN UNKNOWN\_DISC\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE UNKNOWN\_DISC\_CUR;

ELSE

OPEN DISC\_CUR;

LOOP

FETCH DISC\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN DISC\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE DISC\_CUR;

END IF;

ELSE

IF (REGION\_IN = 'Unknown') THEN

OPEN UNKNOWN\_CUR;

LOOP

FETCH UNKNOWN\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN UNKNOWN\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE UNKNOWN\_CUR;

ELSE

OPEN REGION\_CUR;

LOOP

FETCH REGION\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN REGION\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE REGION\_CUR;

END IF ;

END IF ;

RETURN;

END NS\_XCONNECT\_SUMMARY;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Table Function NS\_ADM\_XCONNECT\_SUMMARY \*

\* \*

\* Parameters \*

\* REGION\_IN Region Name \*

\* ONLY\_DISCREPANCIES\_IN (1 = true, 0 = false) \*

\* START\_IN Starting Row Number \*

\* STOP\_IN Ending Row Number \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FUNCTION NS\_ADM\_XCONNECT\_SUMMARY(REGION\_IN IN VARCHAR2, ONLY\_DISCREPANCIES\_IN IN NUMBER, START\_IN IN NUMBER, STOP\_IN IN NUMBER) RETURN NETSMART\_XCONNECT\_TBL PIPELINED IS

OUT\_REC NETSMART\_XCONNECT\_REC := NETSMART\_XCONNECT\_REC(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);

CURSOR REGION\_CUR IS

SELECT

ROWNUMBER,

TID,

DEVICE\_XCONNECTS,

AUDIT\_STATUS,

CIRC\_PATH\_INST\_ID,

CIRC\_PATH\_HUM\_ID,

PATH\_STATUS,

BANDWIDTH,

COMMENTS

FROM (WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

(ROW\_NUMBER () OVER (ORDER BY CDM.REGION, NE.LEAF\_DOMAIN\_NAME, XC.TID, XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE)) ROWNUMBER,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

NETSMART\_ADM\_XCONNECT\_AUDIT XC

JOIN

NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT NE ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION = REGION\_IN

/\*AND XC.MATCH\_CODE <> 'MATCH'\*/

ORDER BY

REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR DISC\_CUR IS

SELECT

ROWNUMBER,

TID,

DEVICE\_XCONNECTS,

AUDIT\_STATUS,

CIRC\_PATH\_INST\_ID,

CIRC\_PATH\_HUM\_ID,

PATH\_STATUS,

BANDWIDTH,

COMMENTS

FROM (WITH DOMAINS AS

(

SELECT DISTINCT

AREA,

REGION,

LEAF\_DOMAIN\_NAME

FROM

CLLI\_DOMAIN\_MAP\_V

)

SELECT

(ROW\_NUMBER () OVER (ORDER BY CDM.REGION, NE.LEAF\_DOMAIN\_NAME, XC.TID, XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE)) ROWNUMBER,

CDM.REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

NETSMART\_ADM\_XCONNECT\_AUDIT XC

JOIN

NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT NE ON XC.TID = NE.TID

LEFT OUTER JOIN

DOMAINS CDM ON (CDM.LEAF\_DOMAIN\_NAME = NE.LEAF\_DOMAIN\_NAME)

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

REGION = REGION\_IN

AND XC.MATCH\_CODE <> 'MATCH'

ORDER BY

REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR UNKNOWN\_CUR IS

SELECT

ROWNUMBER,

TID,

DEVICE\_XCONNECTS,

AUDIT\_STATUS,

CIRC\_PATH\_INST\_ID,

CIRC\_PATH\_HUM\_ID,

PATH\_STATUS,

BANDWIDTH,

COMMENTS

FROM (

SELECT

(ROW\_NUMBER () OVER (ORDER BY 'unknown', 'unknown', XC.TID, XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE)) ROWNUMBER,

'unknown' REGION,

'UNKNOWN\_DOMAIN' LEAF\_DOMAIN\_NAME,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

NETSMART\_ADM\_XCONNECT\_AUDIT XC

JOIN

NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT NE ON XC.TID = NE.TID

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

NE.LEAF\_DOMAIN\_NAME is NULL

/\*AND XC.MATCH\_CODE <> 'MATCH'\*/

ORDER BY

REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

CURSOR UNKNOWN\_DISC\_CUR IS

SELECT

ROWNUMBER,

TID,

DEVICE\_XCONNECTS,

AUDIT\_STATUS,

CIRC\_PATH\_INST\_ID,

CIRC\_PATH\_HUM\_ID,

PATH\_STATUS,

BANDWIDTH,

COMMENTS

FROM (

SELECT

(ROW\_NUMBER () OVER (ORDER BY 'unknown', 'UNKNOWN\_DOMAIN', XC.TID, XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE)) ROWNUMBER,

'unknown' REGION,

'UNKNOWN\_DOMAIN' LEAF\_DOMAIN\_NAME,

XC.TID,

XC.XCONNECT\_ASIDE || ',' || XC.XCONNECT\_ZSIDE DEVICE\_XCONNECTS,

XC.MATCH\_CODE AUDIT\_STATUS,

XC.CIRC\_PATH\_INST\_ID,

CPI.CIRC\_PATH\_HUM\_ID,

CPI.STATUS PATH\_STATUS,

CPI.BANDWIDTH,

XC.MATCH\_STATUS COMMENTS

FROM

NETSMART\_ADM\_XCONNECT\_AUDIT XC

JOIN

NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT NE ON XC.TID = NE.TID

LEFT OUTER JOIN

VZWNET.CIRC\_PATH\_INST CPI ON CPI.CIRC\_PATH\_INST\_ID = XC.CIRC\_PATH\_INST\_ID AND (XC.A\_SIDE\_EQUIP\_ID IS NOT NULL AND XC.A\_SIDE\_EQUIP\_ID = XC.Z\_SIDE\_EQUIP\_ID)

WHERE

NE.LEAF\_DOMAIN\_NAME is NULL

AND XC.MATCH\_CODE <> 'MATCH'

ORDER BY

REGION,

NE.LEAF\_DOMAIN\_NAME,

XC.TID,

DEVICE\_XCONNECTS

)

WHERE

ROWNUMBER BETWEEN START\_IN AND STOP\_IN

ORDER BY

ROWNUMBER;

BEGIN

IF (ONLY\_DISCREPANCIES\_IN = 1) THEN

IF (REGION\_IN = 'Unknown') THEN

OPEN DISC\_CUR;

LOOP

FETCH UNKNOWN\_DISC\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN UNKNOWN\_DISC\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE UNKNOWN\_DISC\_CUR;

ELSE

OPEN DISC\_CUR;

LOOP

FETCH DISC\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN DISC\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE DISC\_CUR;

END IF;

ELSE

IF (REGION\_IN = 'Unknown') THEN

OPEN UNKNOWN\_CUR;

LOOP

FETCH UNKNOWN\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN UNKNOWN\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE UNKNOWN\_CUR;

ELSE

OPEN REGION\_CUR;

LOOP

FETCH REGION\_CUR INTO OUT\_REC.ROWNUMBER, OUT\_REC.TID, OUT\_REC.DEVICE\_XCONNECTS, OUT\_REC.AUDIT\_STATUS, OUT\_REC.CIRC\_PATH\_INST\_ID, OUT\_REC.CIRC\_PATH\_HUM\_ID, OUT\_REC.PATH\_STATUS,

OUT\_REC.BANDWIDTH, OUT\_REC.COMMENTS;

EXIT WHEN REGION\_CUR%NOTFOUND;

PIPE ROW(OUT\_REC);

END LOOP;

CLOSE REGION\_CUR;

END IF ;

END IF ;

RETURN;

END NS\_ADM\_XCONNECT\_SUMMARY;

--

-- Both MSPP and ADM Functions

--

PROCEDURE RUNAUDIT IS

PROCESSNAME VARCHAR2(100);

BEGIN

BEGIN

PROCESSNAME := 'NETSMART\_AUDIT';

-- Tell watchDog the process has started

watchdog.updateprocessstart (processName);

BEGIN

-- Start NetSmart MSPP Audit

-- find the top level Fujitsu equipment (or containers) in Granite Inventory

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'NETSMART\_AUDIT.GETGRANITEEQUIP();', 'N');

GETGRANITEEQUIP ();

-- Audit the loaded device files against the Fujitsu FlashWave devices in Granite

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'NETSMART\_AUDIT.AUDITFWEQUIP();', 'N');

AUDITFWEQUIP ();

-- Audit the NetSmart XConnects against Granite

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'NETSMART\_AUDIT.AUDIT\_XCONNECTS();', 'N');

AUDIT\_XCONNECTS ();

-- Generate the overall summary table

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'NETSMART\_AUDIT.GENSUMMARY();', 'N');

GENSUMMARY ();

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'Completed NetSmart MSPP Audit', 'N');

-- Completed MSPP Audit

-- Start NetSmart ADM Audit

-- find the top level NetSmart equipment (or containers) in Granite Inventory

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'NETSMART\_AUDIT.GETADMGRANITEEQUIP();', 'N');

GETADMGRANITEEQUIP ();

-- Audit the loaded device files against the Fujitsu FlashWave devices in Granite

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'NETSMART\_AUDIT.AUDITADMFWEQUIP();', 'N');

AUDITADMFWEQUIP ();

-- Audit the NetSmart XConnects against Granite

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'NETSMART\_AUDIT.AUDIT\_ADM\_XCONNECTS();', 'N');

AUDIT\_ADM\_XCONNECTS ();

-- Generate the overall summary table

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'NETSMART\_AUDIT.GENADMSUMMARY();', 'N');

GENADMSUMMARY ();

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_TRANSIENT', 'Completed NetSmart ADM Audit', 'N');

-- Completed NetSmart ADM Audit

-- Move the data from work to the display table for testing

--MOVE\_DATA\_FROM\_WK ();

-- mark successful completion of the task

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_SUCCESS', PROCESSNAME || ' ran successfully!', 'N');

WATCHDOG.UPDATEPROCESSEND (PROCESSNAME, 'STATUS\_SUCCESS', 'Y');

WATCHDOG.POPULATE\_LAST\_SUCCESS\_DT\_PROC(PROCESSNAME);

DBMS\_OUTPUT.PUT\_LINE('NetSmart\_Audit.RunAudit() ran successfully!');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SUBSTR('Error: NetSmart\_Audit.runAudit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SUBSTR('Error: NetSmart\_Audit.RunAudit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

-- mark failed completion of the task

WATCHDOG.LOGERROR(PROCESSNAME, 'STATUS\_FAILURE',' Error: NetSmart\_Audit.RunAudit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM||'!', 'N');

WATCHDOG.UPDATEPROCESSEND (PROCESSNAME, 'STATUS\_FAILURE', 'N');

END;

END RUNAUDIT;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Truncate\_WK \*

\* \*

\* Purpose: Truncate work tables. \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE TRUNCATE\_WK IS

BEGIN

-- changed all truncates to deletes

--delete from netsmart\_eq\_summ\_region\_wk;

--

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_EQ\_SUMM\_REGION\_WK';

DBMS\_OUTPUT.PUT\_LINE('Deleted from netsmart\_eq\_summ\_region\_wk');

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_EQ\_SUMM\_REGION\_WK';

DBMS\_OUTPUT.PUT\_LINE('Deleted from netsmart\_adm\_eq\_summ\_region\_wk');

-- -- changed all truncates to deletes

-- delete from netsmart\_eq\_detail\_region\_wk;

--

-- COMMIT;

--

-- DBMS\_OUTPUT.PUT\_LINE('Deleted from netsmart\_eq\_detail\_region\_wk');

-- changed all truncates to deletes

--delete from netsmart\_ne\_vs\_xng\_audit\_wk;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_NE\_VS\_XNG\_AUDIT\_WK';

DBMS\_OUTPUT.PUT\_LINE('Deleted from netsmart\_ne\_vs\_xng\_audit\_wk ');

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK';

DBMS\_OUTPUT.PUT\_LINE('Deleted from netsmart\_adm\_ne\_vs\_xng\_adt\_wk ');

END TRUNCATE\_WK;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Move\_Data\_From\_WK \*

\* \*

\* Purpose: Copy work tables to production tables. \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE MOVE\_DATA\_FROM\_WK IS

BEGIN

--delete from netsmart\_eq\_summ\_region;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_EQ\_SUMM\_REGION';

insert into netsmart\_eq\_summ\_region select \* from netsmart\_eq\_summ\_region\_wk;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_eq\_summ\_region');

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_EQ\_SUMM\_REGION';

insert into netsmart\_adm\_eq\_summ\_region select \* from netsmart\_adm\_eq\_summ\_region\_wk;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_adm\_eq\_summ\_region');

-- delete from netsmart\_eq\_detail\_region;

--

-- insert into netsmart\_eq\_detail\_region select \* from netsmart\_eq\_detail\_region\_wk;

--

-- COMMIT;

--

-- DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_eq\_detail\_region');

--delete from netsmart\_ne\_vs\_xng\_audit;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_NE\_VS\_XNG\_AUDIT';

insert into netsmart\_ne\_vs\_xng\_audit select \* from netsmart\_ne\_vs\_xng\_audit\_wk;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_ne\_vs\_xng\_audit');

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT';

insert into netsmart\_adm\_ne\_vs\_xng\_audit select \* from netsmart\_adm\_ne\_vs\_xng\_adt\_wk;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_ne\_vs\_xng\_audit');

--delete from netsmart\_xconnect\_audit;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_XCONNECT\_AUDIT';

insert into netsmart\_xconnect\_audit select \* from netsmart\_xconnect\_audit\_wk;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_xconnect\_audit');

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_XCONNECT\_AUDIT';

insert into netsmart\_adm\_xconnect\_audit select \* from netsmart\_adm\_xconnect\_audit\_wk;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_adm\_xconnect\_audit');

END MOVE\_DATA\_FROM\_WK;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure Move\_Data\_To\_WK \*

\* \*

\* Purpose: Copy production tables to work tables. \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE MOVE\_DATA\_TO\_WK IS

BEGIN

--DELETE FROM NETSMART\_EQ\_SUMM\_REGION\_WK;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_EQ\_SUMM\_REGION\_WK';

insert into netsmart\_eq\_summ\_region\_wk select \* from netsmart\_eq\_summ\_region;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_eq\_summ\_region\_wk');

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_EQ\_SUMM\_REGION\_WK';

insert into netsmart\_adm\_eq\_summ\_region\_wk select \* from netsmart\_adm\_eq\_summ\_region;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_adm\_eq\_summ\_region\_wk');

-- delete from netsmart\_eq\_detail\_region\_wk;

--

-- insert into netsmart\_eq\_detail\_region\_wk select \* from netsmart\_eq\_detail\_region;

--

-- COMMIT;

--

-- DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_eq\_detail\_region\_wk');

--DELETE FROM NETSMART\_NE\_VS\_XNG\_AUDIT\_WK;

--COMMIT;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_NE\_VS\_XNG\_AUDIT\_WK';

INSERT INTO NETSMART\_NE\_VS\_XNG\_AUDIT\_WK SELECT \* FROM NETSMART\_NE\_VS\_XNG\_AUDIT;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_ne\_vs\_xng\_audit\_wk');

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK';

INSERT INTO NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK SELECT \* FROM NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_adm\_ne\_vs\_xng\_adt\_wk');

--delete from netsmart\_xconnect\_audit\_wk;

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_XCONNECT\_AUDIT\_WK';

insert into netsmart\_xconnect\_audit\_wk select \* from netsmart\_xconnect\_audit;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_xconnect\_audit\_wk');

EXECUTE IMMEDIATE 'TRUNCATE TABLE NETSMART\_ADM\_XCONNECT\_AUDIT\_WK';

insert into netsmart\_adm\_xconnect\_audit\_wk select \* from netsmart\_adm\_xconnect\_audit;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Done populating netsmart\_adm\_xconnect\_audit\_wk');

END MOVE\_DATA\_TO\_WK;

END NETSMART\_AUDIT;

/

--------------------------------------------------------

-- DDL for Package Body NL\_OBJS\_IN\_LIVE\_PATHS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: NL\_OBJS\_IN\_LIVE\_PATHS

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 11/13/2014 Gautam Dev 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE insert\_nl\_objs IS

sqlStmt VARCHAR2(32267);

message varchar2(1000);

methodName VARCHAR2(32267) := 'insert\_nl\_objs';

BEGIN

BEGIN

EXECUTE IMMEDIATE 'truncate table ' || table\_name || ' reuse storage';

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t truncate table: ' || table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

sqlStmt := ' insert into ' || table\_name ||

' (select cpe.circ\_path\_inst\_id, cp.CIRC\_PATH\_HUM\_ID, cpe.SEQUENCE,

cpe.element\_type, ci.status, ci.circ\_inst\_id elem\_inst\_id, cp.TYPE path\_type,

ci.TYPE, ci.circ\_hum\_id element\_name, pm.domain\_inst\_id

FROM vzwnet.circ\_path\_element cpe, vzwnet.circ\_path\_inst cp, vzwnet.PATH\_DOMAIN\_MAP pm, vzwnet.circ\_inst ci, DOMAINS\_REGIONAL\_REPORTING dr

WHERE cpe.circ\_path\_inst\_id = cp.circ\_path\_inst\_id

and cp.CIRC\_PATH\_INST\_ID = cp.circ\_path\_inst\_id

AND pm.circ\_path\_inst\_id = cp.circ\_path\_inst\_id

AND ci.circ\_inst\_id = cpe.segment\_inst\_id

AND cpe.element\_type = ''S'' AND pm.domain\_inst\_id = dr.domain\_inst\_id

AND cp.status = ''Live'' AND ci.status <> ''Live''

)';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into table: ' || table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

sqlStmt := ' insert into ' || table\_name ||

' (select cpe.circ\_path\_inst\_id, cp.CIRC\_PATH\_HUM\_ID, cpe.SEQUENCE,

cpe.element\_type, ci.status, ci.cable\_inst\_id elem\_inst\_id, cp.TYPE path\_type,

ci.TYPE, ci.cable\_name element\_name, pm.domain\_inst\_id

FROM vzwnet.circ\_path\_element cpe, vzwnet.circ\_path\_inst cp, vzwnet.PATH\_DOMAIN\_MAP pm, vzwnet.cable\_inst ci, DOMAINS\_REGIONAL\_REPORTING dr

WHERE cpe.circ\_path\_inst\_id = cp.circ\_path\_inst\_id

and cp.CIRC\_PATH\_INST\_ID = cp.circ\_path\_inst\_id

AND pm.circ\_path\_inst\_id = cp.circ\_path\_inst\_id

AND ci.cable\_inst\_id = cpe.cable\_inst\_id

AND cpe.element\_type = ''B'' AND pm.domain\_inst\_id = dr.domain\_inst\_id

AND cp.status = ''Live'' AND ci.status <> ''Live''

)';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into table: ' || table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

sqlStmt := ' insert into ' || table\_name ||

' (select cpe.circ\_path\_inst\_id, cp.CIRC\_PATH\_HUM\_ID, cpe.SEQUENCE,

cpe.element\_type, ci.status, ci.circ\_path\_inst\_id elem\_inst\_id, cp.TYPE path\_type,

ci.TYPE, ci.circ\_path\_hum\_id element\_name, pm.domain\_inst\_id

FROM vzwnet.circ\_path\_element cpe, vzwnet.circ\_path\_inst cp, vzwnet.PATH\_DOMAIN\_MAP pm, vzwnet.circ\_path\_inst ci, DOMAINS\_REGIONAL\_REPORTING dr

WHERE cpe.circ\_path\_inst\_id = cp.circ\_path\_inst\_id

and cp.CIRC\_PATH\_INST\_ID = cp.circ\_path\_inst\_id

AND pm.circ\_path\_inst\_id = cp.circ\_path\_inst\_id

AND ci.circ\_path\_inst\_id = cpe.path\_inst\_id

AND cpe.element\_type = ''P'' AND pm.domain\_inst\_id = dr.domain\_inst\_id

AND cp.status = ''Live'' AND ci.status <> ''Live''

)';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into table: ' || table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

sqlStmt := ' insert into ' || table\_name ||

' (select cpe.circ\_path\_inst\_id, cp.CIRC\_PATH\_HUM\_ID, cpe.SEQUENCE,

cpe.element\_type, ei.status, ei.equip\_inst\_id elem\_inst\_id, cp.TYPE path\_type,

ei.TYPE, ei.descr element\_name, pm.domain\_inst\_id

FROM vzwnet.circ\_path\_element cpe, vzwnet.circ\_path\_inst cp, vzwnet.PATH\_DOMAIN\_MAP pm, vzwnet.EPA ci, vzwnet.equip\_inst ei, DOMAINS\_REGIONAL\_REPORTING dr

WHERE cpe.circ\_path\_inst\_id = cp.circ\_path\_inst\_id

AND pm.circ\_path\_inst\_id = cp.circ\_path\_inst\_id

AND ei.TYPE not in (''AC DIST PANEL'',''DSX-0'', ''DSX-1'', ''DSX-3'', ''FIBER PANEL'', ''PATCH PANEL'',''LGX'')

AND ci.port\_inst\_id = cpe.port\_inst\_id

AND ei.equip\_inst\_id = ci.equip\_inst\_id

AND cpe.element\_type = ''E'' AND pm.domain\_inst\_id = dr.domain\_inst\_id

AND cp.status = ''Live'' AND ei.status <> ''Live'')';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into table: ' || table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END insert\_nl\_objs;

PROCEDURE insert\_nl\_objs\_summary IS

sqlStmt VARCHAR2(32267);

message varchar2(1000);

methodName VARCHAR2(100) := 'insert\_nl\_objs\_summary';

BEGIN

BEGIN

EXECUTE IMMEDIATE 'truncate table ' || summary\_table\_name || ' reuse storage';

commit;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t truncate table: ' || summary\_table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

sqlStmt := ' insert into ' || summary\_table\_name || ' SELECT

(SELECT dr.AREA from DOMAINS\_REGIONAL\_REPORTING dr where dr.DOMAIN\_INST\_ID = n.DOMAIN\_INST\_ID) AREA,

(SELECT dr.REGION from DOMAINS\_REGIONAL\_REPORTING dr where dr.DOMAIN\_INST\_ID = n.DOMAIN\_INST\_ID) REGION,

count( case when (n.PATH\_TYPE = ''EBH'' and n.ELEMENT\_TYPE = ''S'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) SEG\_EBH\_COUNT,

count(case when (n.PATH\_TYPE <> ''EBH'' and n.ELEMENT\_TYPE = ''S'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) SEG\_NON\_EBH\_COUNT,

count(case when (n.ELEMENT\_TYPE = ''S'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) SEG\_ALL\_COUNT,

count(case when (n.PATH\_TYPE = ''EBH'' and n.ELEMENT\_TYPE = ''E'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) EQUIP\_EBH\_COUNT,

count(case when (n.PATH\_TYPE <> ''EBH'' and n.ELEMENT\_TYPE = ''E'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) EQUIP\_NON\_EBH\_COUNT,

count(case when (n.ELEMENT\_TYPE = ''E'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) EQUIP\_ALL\_COUNT,

count(case when (n.PATH\_TYPE = ''EBH'' and n.ELEMENT\_TYPE = ''P'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) PATH\_EBH\_COUNT,

count(case when (n.PATH\_TYPE <> ''EBH'' and n.ELEMENT\_TYPE = ''P'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) PATH\_NON\_EBH\_COUNT,

count(case when (n.ELEMENT\_TYPE = ''P'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) PATH\_ALL\_COUNT,

count(case when (n.PATH\_TYPE = ''EBH'' and n.ELEMENT\_TYPE = ''B'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) CABLE\_EBH\_COUNT,

count(case when (n.PATH\_TYPE <> ''EBH'' and n.ELEMENT\_TYPE = ''B'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) CABLE\_NON\_EBH\_COUNT,

count(case when (n.ELEMENT\_TYPE = ''B'') Then n.CIRC\_PATH\_INST\_ID ELSE null end) CABLE\_ALL\_COUNT

FROM ' || table\_name || ' N group by N.DOMAIN\_INST\_ID';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into: ' || summary\_table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

sqlStmt := 'insert into ' || summary\_table\_name ||

' SELECT

N.AREA, null, SUM(N.SEG\_EBH\_COUNT),

SUM(N.SEG\_NON\_EBH\_COUNT), SUM(N.SEG\_ALL\_COUNT), SUM(N.EQUIP\_EBH\_COUNT),

SUM(N.EQUIP\_NON\_EBH\_COUNT), SUM(N.EQUIP\_ALL\_COUNT), SUM(N.PATH\_EBH\_COUNT),

SUM(N.PATH\_NON\_EBH\_COUNT), SUM(N.PATH\_ALL\_COUNT), SUM(N.CABLE\_EBH\_COUNT),

SUM(N.CABLE\_NON\_EBH\_COUNT), SUM(N.CABLE\_ALL\_COUNT)

FROM ' || summary\_table\_name || ' N group by rollup(N.area)';

BEGIN

EXECUTE IMMEDIATE sqlStmt;

EXCEPTION

WHEN OTHERS THEN

message := 'Error: '|| methodName ||'(): Can''t insert into: ' || summary\_table\_name ;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END;

PROCEDURE do\_all IS

methodName VARCHAR2(32267) := 'do\_all';

message varchar2(1000);

BEGIN

BEGIN

watchdog.updateprocessstart(processName);

insert\_nl\_objs();

insert\_nl\_objs\_summary();

watchdog.updateprocessend(processName, 'STATUS\_SUCCESS', 'Y');

watchdog.populate\_last\_success\_dt\_proc(processName);

EXCEPTION

WHEN OTHERS THEN

message := 'Error: in '|| methodName;

dbms\_output.put\_line(SubStr(message

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

xng\_reports.watchdog.logerror (processName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

xng\_reports.watchdog.updateprocessend(processName,'STATUS\_FAILURE','N');

RAISE;

END;

END;

END NL\_OBJS\_IN\_LIVE\_PATHS;

/

--------------------------------------------------------

-- DDL for Package Body NONPRINTABLE\_CLEANUP\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."NONPRINTABLE\_CLEANUP\_PKG" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Purpose: to clear nonprintable characters from fields

\* Process: Enter information on column to clean in xng\_reports.nonprintable\_cols\_to\_clean:

\* table\_name - Required - table that column is in (this must be in vzwnet)

\* column\_name - Required - the column to clean

\* clean\_type - Required - Enter ALL removes all display characters; anything other value removes all

\* except for tab and linefeed

\* primary\_key - Required - primary key column name of table that column is in

\* type\_key - Required for all but attr\_value - column in table that has type (ie. type in site\_inst,

\* descr in equip\_inst, num in site\_inst). Type\_key should be null if column is attr\_value

\* in an attr\_setting table.

\*

\* Records with cleanup to be done are put in NONPRINTABLE\_UPDATES - upd\_ts is populated when update

\* complete.

\*

\* The process looks for violations of unique constraints in indexes. Those errors are put in

\* NONPRINTABLE\_ERRORS and removed from NONPRINTABLE\_UPDATES. NONPRINTABLE\_ERRORS contains all bad

\* records that could not be cleaned automatically and should be handled manually.

\*

\* replace(upd\_row.new\_val,'''','''''') is used to insert records that contain single quotes.

\*

\* if column\_name is ATTR\_VALUE meaning a UDA in one of the attr\_setting tables, val\_attr\_inst\_id

\* is populated in NONPRINTABLE\_UPDATES and used when updates are done. Otherwise it is left blank.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* procedure that runs process

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure runCleanup is

BEGIN

processName := 'NONPRINTABLE\_CLEANUP';

-- Tell watchDdog the process has started

BEGIN

watchdog.updateprocessstart (processName);

BEGIN

nonprintableCleanup();

-- tell watchDog process completed successfully

watchdog.updateprocessend (processName, 'STATUS\_SUCCESS', 'Y');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: nonprintableCleanup(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: nonprintableCleanup(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.updateprocessend (processName, 'STATUS\_FAILURE', 'N');

END;

END runCleanup;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Nonprintable Cleanup process

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure nonprintableCleanup is

sql\_stmt VARCHAR2(32767);

upd\_stmt VARCHAR2(32767);

v\_cnt number;

v\_process\_name xng\_reports.all\_processes.process\_name%type;

cursor detail\_cursor is

select table\_name, column\_name, clean\_type, primary\_key, nvl(type\_key,'null') type\_key

from xng\_reports.nonprintable\_cols\_to\_clean

order by table\_name, column\_name;

detail\_row detail\_cursor%ROWTYPE;

cursor upd\_cursor is

select table\_name, column\_name, primary\_key, inst\_id, val\_attr\_inst\_id, new\_val

from xng\_reports.nonprintable\_updates

where upd\_ts is null

order by table\_name, column\_name, inst\_id;

upd\_row upd\_cursor%ROWTYPE;

BEGIN

v\_cnt := 0;

XNG\_REPORTS.TRUNCATE\_XNG\_REPORTS\_TABLE('nonprintable\_errors');

delete from xng\_reports.nonprintable\_updates where upd\_ts is null; commit;

--pull records with bad values in column being cleaned

OPEN detail\_cursor;

LOOP

FETCH detail\_cursor

INTO detail\_row;

EXIT WHEN detail\_cursor%NOTFOUND;

if detail\_row.column\_name <> 'ATTR\_VALUE' then

--not udas/attr\_value

sql\_stmt := 'insert into xng\_reports.nonprintable\_updates

(select table\_name, column\_name, primary\_key, inst\_id, type\_val, field, field\_fix, null as val\_attr\_inst\_id, null as upd\_ts

from (select '''||detail\_row.table\_name||''' as table\_name,

'''||detail\_row.column\_name||''' as column\_name,

'''||detail\_row.primary\_key||''' as primary\_key,

'||detail\_row.primary\_key||' as inst\_id,

'||detail\_row.type\_key||' as type\_val,

'||detail\_row.column\_name||' as field,

xng\_apps.remove\_nonprintable('||detail\_row.column\_name||','''||detail\_row.clean\_type||''') as field\_fix

from vzwnet.'||detail\_row.table\_name||')

where field <> field\_fix)';

else

--udas/attr\_value

sql\_stmt := 'insert into xng\_reports.nonprintable\_updates

(select table\_name, column\_name, primary\_key, inst\_id, van.group\_name||'' / ''||van.attr\_name as type\_val,

field, field\_fix, np.val\_attr\_inst\_id, null as upd\_ts

from (select table\_name, column\_name, primary\_key, inst\_id, type\_val, field, field\_fix, val\_attr\_inst\_id

from (select '''||detail\_row.table\_name||''' as table\_name,

'''||detail\_row.column\_name||''' as column\_name,

'''||detail\_row.primary\_key||''' as primary\_key,

'||detail\_row.primary\_key||' as inst\_id,

'||detail\_row.type\_key||' as type\_val,

'||detail\_row.column\_name||' as field,

xng\_apps.remove\_nonprintable('||detail\_row.column\_name||','''||detail\_row.clean\_type||''') as field\_fix,

val\_attr\_inst\_id

from vzwnet.'||detail\_row.table\_name||')

where field <> field\_fix) np,

vzwnet.val\_attr\_name van

where np.val\_attr\_inst\_id = van.val\_attr\_inst\_id(+))';

end if;

EXECUTE IMMEDIATE sql\_stmt;

COMMIT;

END LOOP;

CLOSE detail\_cursor;

COMMIT WORK;

--function can fail with really bad values

if detail\_row.column\_name <> 'ATTR\_VALUE' then

--not udas/attr\_value

insert into xng\_reports.nonprintable\_errors

(select table\_name, column\_name, primary\_key, inst\_id, null, new\_val, sysdate

from xng\_reports.nonprintable\_updates

where new\_val like 'ERROR - Could not correct string%');

commit;

else

--udas/attr\_value

insert into xng\_reports.nonprintable\_errors

(select table\_name, column\_name, primary\_key, inst\_id, val\_attr\_inst\_id, new\_val, sysdate

from xng\_reports.nonprintable\_updates

where new\_val like 'ERROR - Could not correct string%');

commit;

end if;

delete from xng\_reports.nonprintable\_updates where new\_val like 'ERROR - Could not correct string%';

commit;

--do updates

v\_cnt := 0;

OPEN upd\_cursor;

LOOP

FETCH upd\_cursor

INTO upd\_row;

EXIT WHEN upd\_cursor%NOTFOUND;

if upd\_row.column\_name <> 'ATTR\_VALUE' then

BEGIN

upd\_stmt := 'update vzwnet.'||upd\_row.table\_name||

' set '||upd\_row.column\_name||' = '''||replace(upd\_row.new\_val,'''','''''')||''' where '||upd\_row.primary\_key||

' = '||upd\_row.inst\_id||'';

EXECUTE IMMEDIATE upd\_stmt;

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

insert into xng\_reports.nonprintable\_errors (table\_name, column\_name, primary\_key, inst\_id, error\_msg, upd\_ts)

values (upd\_row.table\_name, upd\_row.column\_name, upd\_row.primary\_key, upd\_row.inst\_id, 'Duplicate exists for '||replace(upd\_row.new\_val,'''',''''''), sysdate);

delete from xng\_reports.nonprintable\_updates

where table\_name = upd\_row.table\_name

and column\_name = upd\_row.column\_name

and primary\_key = upd\_row.primary\_key

and new\_val = upd\_row.new\_val;

commit;

END;

else

BEGIN

upd\_stmt := 'update vzwnet.'||upd\_row.table\_name||

' set '||upd\_row.column\_name||' = '''||replace(upd\_row.new\_val,'''','''''')||''' where '||upd\_row.primary\_key||

' = '||upd\_row.inst\_id||' and val\_attr\_inst\_id = '||upd\_row.val\_attr\_inst\_id||'';

EXECUTE IMMEDIATE upd\_stmt;

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

insert into xng\_reports.nonprintable\_errors (table\_name, column\_name, primary\_key, inst\_id, val\_attr\_inst\_id, error\_msg, upd\_ts)

values (upd\_row.table\_name, upd\_row.column\_name, upd\_row.primary\_key, upd\_row.inst\_id, upd\_row.val\_attr\_inst\_id, 'Duplicate exists for '||replace(upd\_row.new\_val,'''',''''''), sysdate);

delete from xng\_reports.nonprintable\_updates

where table\_name = upd\_row.table\_name

and column\_name = upd\_row.column\_name

and primary\_key = upd\_row.primary\_key

and val\_attr\_inst\_id = upd\_row.val\_attr\_inst\_id

and new\_val = upd\_row.new\_val;

commit;

END;

end if;

update xng\_reports.nonprintable\_updates

set upd\_ts = sysdate

where table\_name = upd\_row.table\_name

and column\_name = upd\_row.column\_name

and inst\_id = upd\_row.inst\_id

and upd\_ts is null;

COMMIT;

END LOOP;

CLOSE upd\_cursor;

COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

xng\_reports.WATCHDOG.logerror(v\_process\_name,4000,SubStr('Error in xng\_apps.NONPRINTABLE\_CLEANUP stored procedure: '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

xng\_reports.WATCHDOG.updateprocessend('NONPRINTABLE\_CLEANUP','STATUS\_FAILURE','N');

--raise;

END nonprintableCleanup;

END;

/

--------------------------------------------------------

-- DDL for Package Body NORTEL\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."NORTEL\_AUDIT"

IS

-- Purpose: To cretae nortel audit tables for voice spans

PROCEDURE nt\_cdma\_audit

IS

-- Two BSM's with same MTX,DCG,SPAN is wrong

CURSOR t1\_dup IS

WITH t1\_audit AS

(SELECT vsm\_device\_name\_mtx,vsm\_device\_name\_bsm,ne\_dcg\_number,ne\_span\_number

FROM nortel\_ne\_vs\_xng\_audit\_wrk

WHERE ne\_span\_type = 'VOICE\_SPAN' AND termination\_type = 'T1'

)

SELECT tgt.vsm\_device\_name\_bsm, tgt.ne\_dcg\_number,tgt.ne\_span\_number

FROM t1\_audit tgt,t1\_audit src

WHERE tgt.vsm\_device\_name\_mtx = src.vsm\_device\_name\_mtx

AND tgt.ne\_dcg\_number = src.ne\_dcg\_number

AND tgt.ne\_span\_number = src.ne\_span\_number

AND tgt.vsm\_device\_name\_bsm <> src.vsm\_device\_name\_bsm;

-- Two BSM's with same MTX,DCG is wrong for EBH

CURSOR ebh\_dup IS

WITH ebh\_audit AS

(SELECT vsm\_device\_name\_mtx,vsm\_device\_name\_bsm,ne\_dcg\_number

FROM nortel\_ne\_vs\_xng\_audit\_wrk

WHERE ne\_span\_type = 'VOICE\_SPAN' AND termination\_type = 'Ethernet'

)

SELECT tgt.vsm\_device\_name\_bsm, tgt.ne\_dcg\_number

FROM ebh\_audit tgt,ebh\_audit src

WHERE tgt.vsm\_device\_name\_mtx = src.vsm\_device\_name\_mtx

AND tgt.ne\_dcg\_number = src.ne\_dcg\_number

AND tgt.vsm\_device\_name\_bsm <> src.vsm\_device\_name\_bsm;

-- One Live and N non Live match

CURSOR t1\_nl\_match IS

SELECT vsm\_device\_name\_bsm,ne\_dcg\_number,ne\_span\_number

FROM nortel\_ne\_vs\_xng\_audit\_wrk

WHERE match\_code='BOTH' AND xng\_path\_status='Live'

AND ne\_span\_type = 'VOICE\_SPAN'

AND termination\_type = 'T1';

-- One Live and N non Live match

CURSOR ebh\_nl\_match IS

SELECT vsm\_device\_name\_bsm,ne\_dcg\_number

FROM nortel\_ne\_vs\_xng\_audit\_wrk

WHERE match\_code='BOTH' AND xng\_path\_status='Live'

AND ne\_span\_type = 'VOICE\_SPAN'

AND termination\_type = 'Ethernet';

CURSOR t1\_cdma\_multiples\_xng IS

WITH

DUPS AS

(

select distinct vsm\_device\_name\_bsm,ne\_dcg\_number,ne\_span\_number,

count(case when xng\_path\_status = 'Live' then 1 else null end) over (Partition by vsm\_device\_name\_bsm,ne\_dcg\_number,ne\_span\_number) as live\_count,

count(case when xng\_path\_status != 'Live' then 1 else null end) over (Partition by vsm\_device\_name\_bsm,ne\_dcg\_number,ne\_span\_number) as non\_live\_count

from nortel\_ne\_vs\_xng\_audit\_wrk src

where src.match\_code='BOTH'

and src.termination\_type='T1'

and src.ne\_span\_type='VOICE\_SPAN'

)

select tgt.rowid rd

from nortel\_ne\_vs\_xng\_audit\_wrk tgt

join DUPS dup

on dup.vsm\_device\_name\_bsm = tgt.vsm\_device\_name\_bsm

AND dup.ne\_dcg\_number = tgt.ne\_dcg\_number

AND dup.ne\_span\_number = tgt.ne\_span\_number

AND (dup.live\_count > 1 OR (dup.live\_count = 0 AND dup.non\_live\_count > 1)

AND tgt.termination\_type='T1'

AND tgt.match\_code='BOTH'

AND tgt.ne\_span\_type='VOICE\_SPAN'

);

cursor ethernet\_cdma\_multiples\_xng is

WITH

DUPS as

(

select distinct vsm\_device\_name\_bsm,ne\_dcg\_number,

count(case when xng\_path\_status = 'Live' then 1 else null end) over (Partition by vsm\_device\_name\_bsm,ne\_dcg\_number) as live\_count,

count(case when xng\_path\_status != 'Live' then 1 else null end) over (Partition by vsm\_device\_name\_bsm,ne\_dcg\_number) as non\_live\_count

from nortel\_ne\_vs\_xng\_audit\_wrk src

where src.match\_code='BOTH'

and src.termination\_type='Ethernet'

and src.ne\_span\_type='VOICE\_SPAN'

)

select tgt.rowid rd

from nortel\_ne\_vs\_xng\_audit\_wrk tgt

join DUPS dup

on dup.vsm\_device\_name\_bsm= tgt.vsm\_device\_name\_bsm

AND dup.ne\_dcg\_number = tgt.ne\_dcg\_number

AND (dup.live\_count > 1 OR (dup.live\_count = 0 AND dup.non\_live\_count > 1)

AND tgt.termination\_type='Ethernet'

AND tgt.match\_code='BOTH'

AND tgt.ne\_span\_type='VOICE\_SPAN'

);

cursor update\_bsm\_with\_unq\_mtx is

with

mtx\_cnt as

(select vsm\_device\_name\_mtx, count(vsm\_device\_name\_bsm)

from bsm\_mtx\_map bmm

group by vsm\_device\_name\_mtx

having count(vsm\_device\_name\_bsm) = 1

)

select B.VSM\_DEVICE\_NAME\_BSM, B.VSM\_DEVICE\_NAME\_MTX

from bsm\_mtx\_map b, mtx\_cnt

where mtx\_cnt.vsm\_device\_name\_mtx = B.VSM\_DEVICE\_NAME\_MTX

;

--JC

cursor update\_nt\_cdma\_wrk\_no\_csr\_port is

WITH

NXA\_PATHS as

(

select nxa.CIRC\_PATH\_INST\_ID

from nortel\_ne\_vs\_xng\_audit\_wrk nxa

where nxa.TERMINATION\_TYPE = 'Ethernet'

AND nxa.XNG\_BANDWIDTH LIKE '%Mbps'

and nxa.match\_code = 'BOTH'

and nxa.XNG\_PATH\_STATUS = 'Live'

)

,

PATHS\_W\_CSR as

(

select cpi.circ\_path\_inst\_id

from NXA\_PATHS cpi

join VZWNET.circ\_path\_element cpe

on cpi.CIRC\_PATH\_INST\_ID = cpe.CIRC\_PATH\_INST\_ID

join vzwnet.epa p

on CPE.PORT\_INST\_ID = P.PORT\_INST\_ID

join VZWNET.equip\_inst ei

on p.equip\_INST\_ID = ei.equip\_INST\_ID

and ei.type = 'CSR'

)

select NXAP.CIRC\_PATH\_INST\_ID

from NXA\_PATHS nxap

MINUS

select pc.CIRC\_PATH\_INST\_ID

from PATHS\_W\_CSR pc;

sql\_stmt varchar2(32000);

BEGIN

sql\_stmt:='DELETE FROM NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK WHERE ne\_span\_type = ''VOICE\_SPAN''';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in XngVsNortel();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end;

commit;

sql\_stmt:='

INSERT INTO NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK

(vsm\_device\_name\_bsm, ne\_dcg\_number, termination\_type,

ne\_span\_number, switch\_id, vsm\_device\_name\_mtx, circ\_path\_hum\_id,

circ\_path\_inst\_id, xng\_dcg\_number, xng\_span\_number,

xng\_path\_status, circ\_type,xng\_bandwidth, match\_code, match\_status,

ne\_span\_type, extract\_date, audit\_date)

WITH bsm\_ether\_terminations AS

(SELECT bei.\*,

SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2) switch\_id,

''Ethernet'' termination\_type, bmm.vsm\_device\_name\_mtx,

bei.ROWID ne\_rowid

FROM bsm\_ethernet\_inv bei JOIN bsm\_mtx\_map bmm

ON bei.vsm\_device\_name\_bsm = bmm.vsm\_device\_name\_bsm

),

bsm\_t1\_terminations AS

(SELECT bdsm.\*,

SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2) switch\_id,

''T1'' termination\_type, bmm.vsm\_device\_name\_mtx,

bdsm.ROWID ne\_rowid

FROM bsm\_dcg\_span\_voice\_map bdsm JOIN bsm\_mtx\_map bmm

ON bdsm.vsm\_device\_name\_bsm = bmm.vsm\_device\_name\_bsm

),

bsm\_ether\_audit AS

(SELECT bet.vsm\_device\_name\_bsm, bet.dcg\_number ne\_dcg\_number,

bet.termination\_type, NULL ne\_span\_number, bet.switch\_id,

bet.vsm\_device\_name\_mtx, cpi.path\_name circ\_path\_hum\_id,

cpi.circ\_path\_inst\_id, cpi.dcg\_number xng\_dcg\_number,

cpi.span\_number xng\_span\_number, cpi.xng\_path\_status,

cpi.xng\_path\_type circ\_type,cpi.bandwidth,

CASE

WHEN cpi.switch\_id IS NULL

THEN ''NE Only''

ELSE ''BOTH''

END match\_code,

CASE

WHEN cpi.span\_number IS NOT NULL AND UPPER(cpi.bandwidth) LIKE ''%DS1%''

THEN ''Pathname Invalid Format, Path Invalid Bandwidth''

WHEN cpi.span\_number IS NOT NULL AND UPPER(cpi.bandwidth) NOT LIKE ''%DS1%''

THEN ''Pathname Invalid Format''

WHEN cpi.span\_number IS NULL AND UPPER(cpi.bandwidth) LIKE ''%DS1%''

THEN ''Path Invalid Bandwidth''

ELSE parse\_status

END match\_status, ''VOICE\_SPAN'' ne\_span\_type,

bet.extract\_date, cpi.ROWID cpi\_rowid, ne\_rowid

FROM bsm\_ether\_terminations bet

LEFT OUTER JOIN

(SELECT \*

FROM vzw\_nt\_voice\_cell\_paths cp

WHERE cp.dcg\_number IS NOT NULL) cpi

ON bet.switch\_id = cpi.switch\_id

AND bet.dcg\_number = cpi.dcg\_number

),

bsm\_t1\_audit AS

(SELECT btt.vsm\_device\_name\_bsm, btt.dcg\_number ne\_dcg\_number,

btt.termination\_type, btt.span\_number ne\_span\_number,

btt.switch\_id, btt.vsm\_device\_name\_mtx,

cpi.path\_name circ\_path\_hum\_id, cpi.circ\_path\_inst\_id,

cpi.dcg\_number xng\_dcg\_number,

cpi.span\_number xng\_span\_number, cpi.xng\_path\_status,

cpi.xng\_path\_type circ\_type,cpi.bandwidth,

CASE

WHEN cpi.switch\_id IS NULL

THEN ''NE Only''

ELSE ''BOTH''

END match\_code,

CASE

WHEN UPPER(cpi.bandwidth) NOT LIKE ''%DS1%''

THEN ''Path Invalid Bandwidth''

ELSE parse\_status

END match\_status, ''VOICE\_SPAN'' ne\_span\_type,

btt.extract\_date, cpi.ROWID cpi\_rowid, ne\_rowid

FROM bsm\_t1\_terminations btt

LEFT OUTER JOIN

(SELECT \*

FROM vzw\_nt\_voice\_cell\_paths cp

WHERE cp.dcg\_number IS NOT NULL

AND cp.span\_number IS NOT NULL) cpi

ON btt.switch\_id = cpi.switch\_id

AND btt.dcg\_number = cpi.dcg\_number

AND btt.span\_number = cpi.span\_number

),

bsm\_ne\_only\_and\_xng AS

(SELECT \*

FROM bsm\_ether\_audit ea

UNION

SELECT \*

FROM bsm\_t1\_audit t1a),

xng\_only\_path\_id AS

(SELECT ROWID cpi\_rowid

FROM vzw\_nt\_voice\_cell\_paths cpi

MINUS

SELECT cpi\_rowid

FROM bsm\_ne\_only\_and\_xng),

bsm\_xng\_only AS

(SELECT NULL vsm\_device\_name\_bsm, NULL ne\_dcg\_number,

CASE

WHEN cpi.bandwidth = ''DS1''

THEN ''T1''

ELSE ''Ethernet''

END termination\_type,

NULL ne\_span\_number, cpi.switch\_id, bmm.vsm\_device\_name\_mtx,

cpi.path\_name circ\_path\_hum\_id, cpi.circ\_path\_inst\_id,

cpi.dcg\_number xng\_dcg\_number,

cpi.span\_number xng\_span\_number, cpi.xng\_path\_status,

cpi.xng\_path\_type circ\_type, cpi.bandwidth,''Xng Only'' match\_code,

parse\_status match\_status, ''VOICE\_SPAN'' ne\_span\_type,

NULL extract\_date, cpi.ROWID cpi\_row\_id, NULL ne\_rowid

FROM vzw\_nt\_voice\_cell\_paths cpi,

xng\_reports.bsm\_mtx\_map bmm,

xng\_only\_path\_id xopi

WHERE cpi.switch\_id =

( SUBSTR (bmm.vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (bmm.vsm\_device\_name\_mtx, 11, 2)

)

AND cpi.ROWID = xopi.cpi\_rowid),

full\_audit AS

(SELECT \*

FROM bsm\_ne\_only\_and\_xng

UNION

SELECT \*

FROM bsm\_xng\_only)

SELECT fa.vsm\_device\_name\_bsm, fa.ne\_dcg\_number, fa.termination\_type,

fa.ne\_span\_number, fa.switch\_id, fa.vsm\_device\_name\_mtx, fa.circ\_path\_hum\_id,

fa.circ\_path\_inst\_id, fa.xng\_dcg\_number, fa.xng\_span\_number,

fa.xng\_path\_status, fa.circ\_type,fa.bandwidth, fa.match\_code, fa.match\_status,

fa.ne\_span\_type, fa.extract\_date, TRUNC (SYSDATE) AS audit\_date

FROM full\_audit fa

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Nortel CDMA audit table created');

FOR cursorRec IN t1\_cdma\_multiples\_xng LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk

SET match\_status = nvl2(match\_status,match\_status||',Multiple spans found in Xng same tokens','Multiple spans found in Xng for same tokens')

WHERE rowid = cursorRec.rd;

END LOOP;

COMMIT;

FOR cursorRec IN ethernet\_cdma\_multiples\_xng LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk

SET match\_status =nvl2(match\_status,match\_status||',Multiple spans found in Xng for same tokens','Multiple spans found in Xng for same tokens')

WHERE rowid = cursorRec.rd

AND termination\_type='Ethernet' ;

END LOOP;

COMMIT;

FOR cursorRec IN t1\_dup LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

SET match\_status = nvl2(match\_status,match\_status||',Multiple BSM with the Same Tokens','Multiple BSM with the Same Tokens'), match\_code='NE Only'

WHERE nnvxa.vsm\_device\_name\_bsm=cursorRec.vsm\_device\_name\_bsm

AND nnvxa.ne\_dcg\_number=cursorRec.ne\_dcg\_number

AND nnvxa.ne\_span\_number=cursorRec.ne\_span\_number

AND (nnvxa.match\_code ='BOTH' OR nnvxa.xng\_dcg\_number IS NULL)

AND nnvxa.termination\_type='T1'

AND nnvxa.ne\_span\_type='VOICE\_SPAN';

END LOOP;

COMMIT;

FOR cursorRec IN ebh\_dup LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

SET match\_status = nvl2(match\_status,match\_status||',Multiple BSM with the Same Tokens','Multiple BSM with the Same Tokens'),match\_code='NE Only'

WHERE nnvxa.vsm\_device\_name\_bsm=cursorRec.vsm\_device\_name\_bsm

AND nnvxa.ne\_dcg\_number=cursorRec.ne\_dcg\_number

AND nnvxa.termination\_type='Ethernet'

AND (nnvxa.match\_code ='BOTH' OR nnvxa.xng\_dcg\_number IS NULL)

AND nnvxa.ne\_span\_type='VOICE\_SPAN';

END LOOP;

COMMIT;

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk

SET match\_code = 'NE Only'

WHERE match\_code = 'BOTH' AND match\_status IS NOT NULL and ne\_span\_type='VOICE\_SPAN';

COMMIT;

FOR cursorRec IN t1\_nl\_match LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

SET match\_status = nvl2(match\_status, 'Live Path Exists,'||match\_status,'Live Path Exists')

WHERE nnvxa.vsm\_device\_name\_bsm=cursorRec.vsm\_device\_name\_bsm

AND nnvxa.ne\_dcg\_number=cursorRec.ne\_dcg\_number

AND nnvxa.ne\_span\_number=cursorRec.ne\_span\_number

AND nnvxa.xng\_path\_status <> 'Live'

AND nnvxa.termination\_type='T1'

AND nnvxa.ne\_span\_type='VOICE\_SPAN';

END LOOP;

COMMIT;

FOR cursorRec IN ebh\_nl\_match LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

SET match\_status = nvl2(match\_status, 'Live Path Exists,'||match\_status,'Live Path Exists')

WHERE nnvxa.vsm\_device\_name\_bsm=cursorRec.vsm\_device\_name\_bsm

AND nnvxa.ne\_dcg\_number=cursorRec.ne\_dcg\_number

AND nnvxa.termination\_type='Ethernet'

AND nnvxa.xng\_path\_status <> 'Live'

AND nnvxa.ne\_span\_type='VOICE\_SPAN';

END LOOP;

COMMIT;

FOR cursorRec IN update\_bsm\_with\_unq\_mtx LOOP

update nortel\_ne\_vs\_xng\_audit\_wrk nxa set

vsm\_device\_name\_bsm = cursorRec.vsm\_device\_name\_bsm

where vsm\_device\_name\_bsm is null

and vsm\_device\_name\_mtx = cursorRec.vsm\_device\_name\_mtx

and nxa.MATCH\_CODE = 'Xng Only';

END LOOP;

COMMIT;

--JC

FOR cursorRec IN update\_nt\_cdma\_wrk\_no\_csr\_port LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nxa

SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS =

CASE

when MATCH\_STATUS = 'BOTH/IP Match'

then 'No CSR port on the e-pipe'

else

nvl2(match\_status, 'No CSR port on the e-pipe, '||match\_status, 'No CSR port on the e-pipe')

end

WHERE

NXA.CIRC\_PATH\_INST\_ID = cursorRec.CIRC\_PATH\_INST\_ID

;

END LOOP;

commit;

--JC

dbms\_output.put\_line('Nortel audit completd successfully');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in XngVsNortel(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end nt\_cdma\_audit;

PROCEDURE nt\_evdo\_audit

IS

-- One Live and N non Live match

CURSOR t1\_nl\_match IS

SELECT ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number,ne\_span\_number

FROM nortel\_ne\_vs\_xng\_audit\_wrk

WHERE match\_code='BOTH' AND xng\_path\_status='Live'

AND ne\_status='Up'

AND ne\_span\_type = 'DATA\_SPAN'

AND termination\_type = 'T1';

-- One Live and N non Live match

CURSOR ebh\_nl\_match IS

SELECT ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number

FROM nortel\_ne\_vs\_xng\_audit\_wrk

WHERE match\_code='BOTH' AND xng\_path\_status='Live' AND ne\_status='Up'

AND ne\_span\_type = 'DATA\_SPAN'

AND termination\_type = 'Ethernet';

-- if both and t1 and ebh are turned up, then flag as NE Only and change the match\_status

cursor t1\_ebh\_both\_up is

WITH

DUPS as

(

select distinct ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number, ne\_status

from nortel\_ne\_vs\_xng\_audit\_wrk src

where

src.termination\_type='Ethernet'

and src.ne\_span\_type='DATA\_SPAN'

and SRC.NE\_STATUS='Up'

)

select distinct tgt.ne\_dom\_ip,tgt.ne\_dcg\_number,tgt.ne\_slot\_number -- tgt.rowid rd

from nortel\_ne\_vs\_xng\_audit\_wrk tgt

join DUPS dup

on dup.ne\_dom\_ip = tgt.ne\_dom\_ip

AND dup.ne\_dcg\_number = tgt.ne\_dcg\_number

AND dup.ne\_slot\_number = tgt.ne\_slot\_number

and dup.ne\_status=tgt.ne\_status

AND tgt.termination\_type='T1'

AND tgt.ne\_span\_type='DATA\_SPAN'

and tgt.ne\_status='Up'

;

cursor t1\_evdo\_multiples is

WITH

DUPS as

(

select distinct ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number,ne\_span\_number,

count(case when xng\_path\_status = 'Live' then 1 else null end) over (Partition by ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number,ne\_span\_number) as live\_count,

count(case when xng\_path\_status != 'Live' then 1 else null end) over (Partition by ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number,ne\_span\_number) as non\_live\_count

from nortel\_ne\_vs\_xng\_audit\_wrk src

where src.match\_code='BOTH'

and src.termination\_type='T1'

and src.ne\_span\_type='DATA\_SPAN'

)

select tgt.rowid rd

from nortel\_ne\_vs\_xng\_audit\_wrk tgt

join DUPS dup

on dup.ne\_dom\_ip = tgt.ne\_dom\_ip

AND dup.ne\_dcg\_number = tgt.ne\_dcg\_number

AND dup.ne\_slot\_number = tgt.ne\_slot\_number

AND dup.ne\_span\_number = tgt.ne\_span\_number

AND (dup.live\_count > 1 OR (dup.live\_count = 0 AND dup.non\_live\_count > 1)

AND tgt.termination\_type='T1'

AND tgt.match\_code='BOTH'

AND tgt.ne\_span\_type='DATA\_SPAN'

);

cursor ethernet\_evdo\_multiples is

WITH

DUPS as

(

select distinct ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number,

count(case when xng\_path\_status = 'Live' then 1 else null end) over (Partition by ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number) as live\_count,

count(case when xng\_path\_status != 'Live' then 1 else null end) over (Partition by ne\_dom\_ip,ne\_dcg\_number,ne\_slot\_number) as non\_live\_count

from nortel\_ne\_vs\_xng\_audit\_wrk src

where src.match\_code='BOTH'

and src.termination\_type='Ethernet'

and src.ne\_span\_type='DATA\_SPAN'

)

select tgt.rowid rd

from nortel\_ne\_vs\_xng\_audit\_wrk tgt

join DUPS dup

on dup.ne\_dom\_ip = tgt.ne\_dom\_ip

AND dup.ne\_dcg\_number = tgt.ne\_dcg\_number

AND dup.ne\_slot\_number = tgt.ne\_slot\_number

AND (dup.live\_count > 1 OR (dup.live\_count = 0 AND dup.non\_live\_count > 1)

AND tgt.termination\_type='Ethernet'

AND tgt.match\_code='BOTH'

AND tgt.ne\_span\_type='DATA\_SPAN'

);

--JC

cursor update\_nt\_evdo\_wrk\_no\_csr\_port is

WITH

NXA\_PATHS as

(

select nxa.CIRC\_PATH\_INST\_ID

from nortel\_ne\_vs\_xng\_audit\_wrk nxa

where nxa.TERMINATION\_TYPE = 'Ethernet'

AND nxa.XNG\_BANDWIDTH LIKE '%Mbps'

and nxa.match\_code = 'BOTH'

and nxa.XNG\_PATH\_STATUS = 'Live'

)

,

PATHS\_W\_CSR as

(

select cpi.circ\_path\_inst\_id

from NXA\_PATHS cpi

join VZWNET.circ\_path\_element cpe

on cpi.CIRC\_PATH\_INST\_ID = cpe.CIRC\_PATH\_INST\_ID

join vzwnet.epa p

on CPE.PORT\_INST\_ID = P.PORT\_INST\_ID

join VZWNET.equip\_inst ei

on p.equip\_INST\_ID = ei.equip\_INST\_ID

and ei.type = 'CSR'

)

select NXAP.CIRC\_PATH\_INST\_ID

from NXA\_PATHS nxap

MINUS

select pc.CIRC\_PATH\_INST\_ID

from PATHS\_W\_CSR pc;

sql\_stmt varchar2(32000);

BEGIN

sql\_stmt:='DELETE FROM NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK WHERE ne\_span\_type = ''DATA\_SPAN''';

begin

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sql\_stmt);

dbms\_output.put\_line(SubStr('Error in nt\_evdo\_audit();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

end;

commit;

sql\_stmt:= 'INSERT INTO NORTEL\_NE\_VS\_XNG\_AUDIT\_WRK

(vsm\_device\_name\_bsm, vsm\_device\_name\_ems, ne\_dcg\_number,

ne\_slot\_number, ne\_span\_number, ne\_dom\_ip, ne\_status,

termination\_type, switch\_id, circ\_path\_hum\_id, circ\_path\_inst\_id,

xng\_dcg\_number, xng\_slot\_number, xng\_span\_number, xng\_dom\_ip,

xng\_path\_status, circ\_type,xng\_bandwidth, match\_code, match\_status,

ne\_span\_type,extract\_date,audit\_date)

WITH t1\_ne\_spans AS

(SELECT bdi.ems\_name vsm\_device\_name\_ems, bdi.vsm\_device\_name\_bsm,

bdi.dcg\_number, bdi.slot\_number, epi.port\_num span\_number,

bdi.dom\_ip, epi.status, ''T1'' port\_type , epi.extract\_date,

epi.ROWID ne\_rowid

FROM nortel\_dom\_inventory bdi JOIN nortel\_port\_inventory epi

ON bdi.seq\_num = epi.dom\_seq\_num AND epi.port\_type = ''T1E1''

),

ether\_ne\_spans AS

(

-- if any of the ethernet ports are hot then it is an ebh site

SELECT DISTINCT bdi.ems\_name vsm\_device\_name\_ems,

bdi.vsm\_device\_name\_bsm, bdi.dcg\_number,

bdi.slot\_number, NULL span\_number, bdi.dom\_ip,

epi.status, port\_type, epi.extract\_date,

bdi.ROWID ne\_rowid

FROM nortel\_dom\_inventory bdi JOIN nortel\_port\_inventory epi

ON bdi.seq\_num = epi.dom\_seq\_num

AND epi.port\_type = ''Ethernet''

AND primary\_ip <> ''0.0.0.0''

AND epi.status = ''Up''

),

ether\_audit AS

(SELECT ne.vsm\_device\_name\_bsm, ne.vsm\_device\_name\_ems,

ne.dcg\_number ne\_dcg\_number, ne.slot\_number ne\_slot\_number,

ne.span\_number ne\_span\_number, ne.dom\_ip ne\_dom\_ip,

ne.status ne\_status, ne.port\_type termination\_type,

NVL (vcpi.switch\_id, ''UNKNOWN'') switch\_id, vcpi.path\_name,

vcpi.circ\_path\_inst\_id, vcpi.dcg\_number xng\_dcg\_number,

vcpi.slot\_number xng\_slot\_number,

vcpi.span\_number xng\_span\_number, vcpi.dom\_node\_ip xng\_dom\_ip,

vcpi.xng\_path\_status, vcpi.xng\_path\_type,vcpi.bandwidth,

CASE

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

and vcpi.SPAN\_NUMBER is null

and (upper(vcpi.BANDWIDTH) like ''%MBPS''

or upper(vcpi.BANDWIDTH) = ''AGGREGATE'')

THEN ''BOTH''

ELSE ''NE Only''

END match\_code,

CASE

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

and vcpi.SPAN\_NUMBER is null

and (upper(vcpi.BANDWIDTH) like ''%MBPS''

or upper(vcpi.BANDWIDTH) = ''AGGREGATE'')

THEN ''BOTH/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

and vcpi.SPAN\_NUMBER is null

and upper(vcpi.BANDWIDTH) not like ''%MBPS''

and upper(vcpi.BANDWIDTH) <> ''AGGREGATE''

THEN ''Path Invalid Bandwidth/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

and vcpi.SPAN\_NUMBER is not null

and upper(vcpi.BANDWIDTH) not like ''%MBPS''

and upper(vcpi.BANDWIDTH) <> ''AGGREGATE''

THEN ''Pathname Invalid Format, Path Invalid Bandwidth/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

and vcpi.SPAN\_NUMBER is not null

and (upper(vcpi.BANDWIDTH) like ''%MBPS''

or upper(vcpi.BANDWIDTH) = ''AGGREGATE'')

THEN ''Pathname Invalid Format/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND (vcpi.dcg\_number <> ne.dcg\_number

or vcpi.slot\_number <> ne.slot\_number)

and vcpi.SPAN\_NUMBER is not null

and (upper(vcpi.BANDWIDTH) like ''%MBPS''

or upper(vcpi.BANDWIDTH) = ''AGGREGATE'')

THEN ''Pathname Token Mismatch, Pathname Invalid Format/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND (vcpi.dcg\_number <> ne.dcg\_number

or vcpi.slot\_number <> ne.slot\_number)

and vcpi.SPAN\_NUMBER is not null

and upper(vcpi.BANDWIDTH) not like ''%MBPS''

and upper(vcpi.BANDWIDTH) <> ''AGGREGATE''

THEN ''Pathname Token Mismatch, Pathname Invalid Format, Path Invalid Bandwidth/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND (vcpi.dcg\_number <> ne.dcg\_number

or vcpi.slot\_number <> ne.slot\_number)

and vcpi.SPAN\_NUMBER is null

and upper(vcpi.BANDWIDTH) not like ''%MBPS''

and upper(vcpi.BANDWIDTH) <> ''AGGREGATE''

THEN ''Pathname Token Mismatch, Path Invalid Bandwidth/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND (vcpi.dcg\_number <> ne.dcg\_number

or vcpi.slot\_number <> ne.slot\_number)

and vcpi.SPAN\_NUMBER is null

and (upper(vcpi.BANDWIDTH) like ''%MBPS''

or upper(vcpi.BANDWIDTH) = ''AGGREGATE'')

THEN ''Pathname Token Mismatch/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number is null

and (upper(vcpi.BANDWIDTH) like ''%MBPS''

or upper(vcpi.BANDWIDTH) = ''AGGREGATE'')

THEN ''Pathname Invalid Format/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number is null

and upper(vcpi.BANDWIDTH) not like ''%MBPS''

and upper(vcpi.BANDWIDTH) <> ''AGGREGATE''

THEN ''Pathname Invalid Format, Path Invalid Bandwidth/IP Match''

WHEN parse\_status IS NOT NULL

THEN ''??????'' || parse\_status || ''/IP Match''

END match\_status,

''DATA\_SPAN'' ne\_span\_type, ne.extract\_date,

vcpi.ROWID cpi\_rowid, ne\_rowid

FROM ether\_ne\_spans ne LEFT OUTER JOIN vzw\_nt\_evdo\_cell\_paths vcpi

ON ne.dom\_ip = vcpi.dom\_node\_ip

),

t1\_audit AS

(SELECT ne.vsm\_device\_name\_bsm, ne.vsm\_device\_name\_ems,

ne.dcg\_number ne\_dcg\_number, ne.slot\_number ne\_slot\_number,

ne.span\_number ne\_span\_number, ne.dom\_ip ne\_dom\_ip,

ne.status ne\_status, ne.port\_type termination\_type,

NVL (vcpi.switch\_id, ''UNKNOWN'') switch\_id, vcpi.path\_name,

vcpi.circ\_path\_inst\_id, vcpi.dcg\_number xng\_dcg\_number,

vcpi.slot\_number xng\_slot\_number,

vcpi.span\_number xng\_span\_number, vcpi.dom\_node\_ip xng\_dom\_ip,

vcpi.xng\_path\_status, vcpi.xng\_path\_type,vcpi.bandwidth,

CASE

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

AND vcpi.span\_number = ne.span\_number

AND vcpi.BANDWIDTH = ''DS1''

THEN ''BOTH''

ELSE ''NE Only''

END match\_code,

CASE

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

AND vcpi.span\_number = ne.span\_number

AND vcpi.BANDWIDTH = ''DS1''

THEN ''BOTH/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

AND vcpi.span\_number = ne.span\_number

AND vcpi.BANDWIDTH <> ''DS1''

THEN ''Path Invalid Bandwidth/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

AND vcpi.span\_number is null

AND vcpi.BANDWIDTH <> ''DS1''

THEN ''Pathname Invalid Format, Path Invalid Bandwidth/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND vcpi.dcg\_number = ne.dcg\_number

AND vcpi.slot\_number = ne.slot\_number

AND vcpi.span\_number is null

AND vcpi.BANDWIDTH = ''DS1''

THEN ''Pathname Invalid Format/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND (vcpi.dcg\_number <> ne.dcg\_number

or vcpi.slot\_number <> ne.slot\_number

or vcpi.span\_number <> ne.span\_number)

AND vcpi.BANDWIDTH = ''DS1''

THEN ''Pathname Token Mismatch/IP Match''

WHEN vcpi.dom\_node\_ip IS NOT NULL

AND ne.dom\_ip IS NOT NULL

AND (vcpi.dcg\_number <> ne.dcg\_number

or vcpi.slot\_number <> ne.slot\_number

or vcpi.span\_number <> ne.span\_number)

AND vcpi.BANDWIDTH <> ''DS1''

THEN ''Pathname Token Mismatch, Path Invalid Bandwidth/IP Match''

WHEN parse\_status IS NOT NULL

THEN ''?????'' || parse\_status ||''/IP Match''

END match\_status,

''DATA\_SPAN'' ne\_span\_type, ne.extract\_date,

vcpi.ROWID cpi\_rowid, ne\_rowid

FROM t1\_ne\_spans ne LEFT OUTER JOIN vzw\_nt\_evdo\_cell\_paths vcpi

ON ne.dom\_ip = vcpi.dom\_node\_ip

AND ne.span\_number = vcpi.span\_number

),

t1\_ether\_audit AS

(SELECT ne.vsm\_device\_name\_bsm, ne.vsm\_device\_name\_ems, ne\_dcg\_number,

ne\_slot\_number, ne\_span\_number, ne\_dom\_ip, ne\_status,

termination\_type, switch\_id, path\_name, circ\_path\_inst\_id,

xng\_dcg\_number, xng\_slot\_number, xng\_span\_number, xng\_dom\_ip,

xng\_path\_status, xng\_path\_type,bandwidth,

match\_code, match\_status,

ne\_span\_type, ne.extract\_date, cpi\_rowid, ne\_rowid

FROM t1\_audit ne

UNION

SELECT ne.vsm\_device\_name\_bsm, ne.vsm\_device\_name\_ems, ne\_dcg\_number,

ne\_slot\_number, ne\_span\_number, ne\_dom\_ip, ne\_status,

termination\_type, switch\_id, path\_name, circ\_path\_inst\_id,

xng\_dcg\_number, xng\_slot\_number, xng\_span\_number, xng\_dom\_ip,

xng\_path\_status, xng\_path\_type, bandwidth,match\_code, match\_status,

ne\_span\_type, ne.extract\_date, cpi\_rowid, ne\_rowid

FROM ether\_audit ne),

xng\_only\_paths\_id AS

(SELECT ROWID cpi\_rowid

FROM vzw\_nt\_evdo\_cell\_paths cpi

WHERE cpi.dom\_node\_ip IS NOT NULL

MINUS

SELECT cpi\_rowid

FROM t1\_ether\_audit ea),

xng\_only AS

(SELECT NULL vsm\_device\_name\_bsm, NULL vsm\_device\_name\_ems,

NULL ne\_dcg\_number, NULL ne\_slot\_number, NULL ne\_span\_number,

NULL ne\_dom\_ip, NULL ne\_status,

CASE

WHEN cpi.bandwidth = ''DS1''

THEN ''T1''

ELSE ''Ethernet''

END termination\_type,

NVL (cpi.switch\_id, ''UNKNOWN'') switch\_id, cpi.path\_name,

cpi.circ\_path\_inst\_id, cpi.dcg\_number xng\_dcg\_number,

cpi.slot\_number xng\_slot\_number,

cpi.span\_number xng\_span\_number, cpi.dom\_node\_ip xng\_dom\_ip,

cpi.xng\_path\_status, cpi.xng\_path\_type,cpi.bandwidth, ''Xng Only'' match\_code,

CASE

WHEN parse\_status IS NULL

THEN ''Invalid IP''

ELSE parse\_status || ''/Invalid IP''

END match\_status,

''DATA\_SPAN'' ne\_span\_type, NULL extract\_date,

cpi.ROWID cpi\_rowid, NULL ne\_rowid

FROM vzw\_nt\_evdo\_cell\_paths cpi JOIN xng\_only\_paths\_id xngo

ON xngo.cpi\_rowid = cpi.ROWID

),

full\_audit AS

(SELECT \*

FROM t1\_ether\_audit

UNION

SELECT \*

FROM xng\_only)

SELECT fa.vsm\_device\_name\_bsm, fa.vsm\_device\_name\_ems,fa.ne\_dcg\_number,

fa.ne\_slot\_number, fa.ne\_span\_number, fa.ne\_dom\_ip, fa.ne\_status,

fa.termination\_type, fa.switch\_id, fa.path\_name, fa.circ\_path\_inst\_id,

fa.xng\_dcg\_number, fa.xng\_slot\_number, fa.xng\_span\_number, fa.xng\_dom\_ip,

fa.xng\_path\_status, fa.xng\_path\_type,fa.bandwidth,fa.match\_code, fa.match\_status,

fa.ne\_span\_type, fa.extract\_date,TRUNC (SYSDATE) AS audit\_date

FROM full\_audit fa';

EXECUTE IMMEDIATE sql\_stmt;

commit;

dbms\_output.put\_line('Nortel EVDO audit completed');

FOR cursorRec IN t1\_evdo\_multiples LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk

set match\_status = nvl2(match\_status,match\_status||',Multiple spans found in Xng for same tokens','Multiple spans found in Xng for same tokens') , match\_code='NE Only'

WHERE rowid = cursorRec.rd;

END LOOP;

commit;

FOR cursorRec IN ethernet\_evdo\_multiples LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk

set match\_status = nvl2(match\_status,match\_status||',Multiple spans found in Xng for same tokens','Multiple spans found in Xng for same tokens') , match\_code='NE Only'

WHERE rowid = cursorRec.rd;

END LOOP;

commit;

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk

SET match\_code = 'NE Only'

WHERE match\_code = 'BOTH' AND match\_status <>'BOTH/IP Match' and ne\_span\_type='DATA\_SPAN';

COMMIT;

FOR cursorRec IN t1\_nl\_match LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

SET match\_status = nvl2(match\_status, 'Live Path Exists,'||match\_status,'Live Path Exists')

WHERE nnvxa.ne\_dom\_ip=cursorRec.ne\_dom\_ip

AND nnvxa.ne\_dcg\_number=cursorRec.ne\_dcg\_number

AND nnvxa.ne\_slot\_number=cursorRec.ne\_slot\_number

AND nnvxa.ne\_span\_number=cursorRec.ne\_span\_number

AND nnvxa.xng\_path\_status <> 'Live' AND ne\_status='Up'

AND nnvxa.termination\_type='T1'

AND nnvxa.ne\_span\_type='DATA\_SPAN';

END LOOP;

COMMIT;

FOR cursorRec IN ebh\_nl\_match LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

SET match\_status = nvl2(match\_status, 'Live Path Exists,'||match\_status,'Live Path Exists')

WHERE nnvxa.ne\_dom\_ip=cursorRec.ne\_dom\_ip

AND nnvxa.ne\_dcg\_number=cursorRec.ne\_dcg\_number

AND nnvxa.ne\_slot\_number=cursorRec.ne\_slot\_number

AND nnvxa.termination\_type='Ethernet'

AND nnvxa.xng\_path\_status <> 'Live' AND ne\_status='Up'

AND nnvxa.ne\_span\_type='DATA\_SPAN';

END LOOP;

COMMIT;

--JC

FOR cursorRec IN update\_nt\_evdo\_wrk\_no\_csr\_port LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nxa

SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS =

CASE

when MATCH\_STATUS = 'BOTH/IP Match'

then 'No CSR port on the e-pipe'

else

nvl2(match\_status, 'No CSR port on the e-pipe, '||match\_status, 'No CSR port on the e-pipe')

end

WHERE

nxa.CIRC\_PATH\_INST\_ID = cursorRec.CIRC\_PATH\_INST\_ID

;

END LOOP;

commit;

--JC

FOR cursorRec IN t1\_ebh\_both\_up LOOP

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

SET MATCH\_CODE = 'NE Only',

MATCH\_STATUS =

CASE

when MATCH\_STATUS = 'BOTH/IP Match'

then 'T1 and EBH are both Up'

else

nvl2(match\_status, 'T1 and EBH are both Up, '||match\_status, 'T1 and EBH are both Up')

end

WHERE nnvxa.ne\_dom\_ip=cursorRec.ne\_dom\_ip

AND nnvxa.ne\_dcg\_number=cursorRec.ne\_dcg\_number

AND nnvxa.ne\_slot\_number=cursorRec.ne\_slot\_number

AND nnvxa.ne\_status='Up'

;

END LOOP;

commit;

-- LR 1/25/2012 - update vsm\_device\_name\_mtx when mtx is null

UPDATE nortel\_ne\_vs\_xng\_audit\_wrk w

set w.vsm\_device\_name\_mtx=(select m.vsm\_device\_name\_mtx from bsm\_mtx\_map m where m.vsm\_device\_name\_bsm=w.vsm\_device\_name\_bsm)

where w.vsm\_device\_name\_mtx is null

and w.vsm\_device\_name\_ems is not null

;

commit;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END nt\_evdo\_audit;

procedure populate\_cdma\_regional\_summary is

cursor updateCompliance is

select summ.rowid rd,summ.\* from NORTEL\_CDMA\_AUDIT\_REG\_SUMM\_WRK summ;

sqlStmt VARCHAR2(32767);

sqlStmtT1Comp VARCHAR2(32767);

sqlStmtEbhComp VARCHAR2(32767);

sqlStmtOvlComp VARCHAR2(32767);

t1Comp NUMBER;

ebhComp NUMBER;

overallComp NUMBER;

t1Reported NUMBER;

ebhReported NUMBER;

BEGIN

sqlStmt := 'truncate table NORTEL\_CDMA\_AUDIT\_REG\_SUMM\_WRK';

execute immediate sqlStmt;

sqlStmt := 'insert into NORTEL\_CDMA\_AUDIT\_REG\_SUMM\_WRK

(area,region,t1\_discovered,t1\_matched\_live,t1\_matched\_nonlive,ebh\_discovered,ebh\_matched\_live,ebh\_matched\_nonlive,ebh\_csr\_matched\_live,ebh\_csr\_matched\_non\_live

)

WITH mtx\_region\_map AS

(SELECT DISTINCT cdmp.area, cdmp.region, m.vsm\_device\_name\_mtx,

( SUBSTR (vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (vsm\_device\_name\_mtx, 11, 2)

) switch\_id

FROM xng\_reports.clli\_domain\_map\_v cdmp

LEFT OUTER JOIN

(SELECT \*

FROM xng\_reports.bsm\_mtx\_map bmm

WHERE bmm.vsm\_device\_name\_bsm LIKE ''%BSM%''

AND bmm.mtx\_status = ''Live'') m

ON SUBSTR (m.vsm\_device\_name\_mtx, 1, 6) =

SUBSTR (cdmp.clli, 1, 6)

WHERE cdmp.area <> ''NNO''

),

t1\_discovered AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) t1\_discovered

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,nnvxa.vsm\_device\_name\_bsm,

nnvxa.ne\_dcg\_number, nnvxa.ne\_span\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

WHERE nnvxa.termination\_type = ''T1''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.match\_code IN (''BOTH'', ''NE Only''))

GROUP BY vsm\_device\_name\_mtx),

ethernet\_discovered AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_discovered

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,nnvxa.vsm\_device\_name\_bsm,

nnvxa.ne\_dcg\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.match\_code IN (''BOTH'', ''NE Only''))

GROUP BY vsm\_device\_name\_mtx),

t1\_matched\_live AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) t1\_matched\_live

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number, nnvxa.ne\_span\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

WHERE nnvxa.termination\_type = ''T1''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH'')

GROUP BY vsm\_device\_name\_mtx),

t1\_matched\_nonlive AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) t1\_matched\_nonlive

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number, nnvxa.ne\_span\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

WHERE nnvxa.termination\_type = ''T1''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status <> ''Live''

AND nnvxa.match\_code = ''BOTH''

MINUS

SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number, nnvxa.ne\_span\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

WHERE nnvxa.termination\_type = ''T1''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH'')

GROUP BY vsm\_device\_name\_mtx),

ethernet\_matched\_live AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_matched\_live

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH''

)

GROUP BY vsm\_device\_name\_mtx),

ethernet\_matched\_nonlive AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_matched\_nonlive

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status <> ''Live''

AND nnvxa.match\_code = ''BOTH''

MINUS

SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH''

)

GROUP BY vsm\_device\_name\_mtx)

--JC

,

ethernet\_csr\_matched\_live as

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_csr\_matched\_live

FROM ( SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx, nnvxa.ne\_dcg\_number, NNVXA.XNG\_BANDWIDTH

FROM mtx\_region\_map mrp, xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

join VZWNET.CIRC\_PATH\_INST cpi on NNVXA.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH''

AND nnvxa.vsm\_device\_name\_mtx = mrp.vsm\_device\_name\_mtx

and NNVXA.XNG\_BANDWIDTH like ''%Mbps''

and EI.TYPE = ''CSR'' )

GROUP BY vsm\_device\_name\_mtx)

,

ethernet\_csr\_matched\_non\_live as

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_csr\_matched\_non\_live

FROM ( SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx, nnvxa.ne\_dcg\_number, NNVXA.XNG\_BANDWIDTH

FROM mtx\_region\_map mrp, xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

join VZWNET.CIRC\_PATH\_INST cpi on NNVXA.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status <> ''Live''

AND nnvxa.match\_code = ''BOTH''

AND nnvxa.vsm\_device\_name\_mtx = mrp.vsm\_device\_name\_mtx

and NNVXA.XNG\_BANDWIDTH like ''%Mbps''

and EI.TYPE = ''CSR'' )

GROUP BY vsm\_device\_name\_mtx)

--JC

SELECT mrm.area, mrm.region,

SUM (NVL (t1\_discovered, 0)) AS t1\_discovered,

SUM (NVL (t1\_matched\_live, 0)) AS t1\_matched\_live,

SUM (NVL (t1\_matched\_nonlive, 0)) AS t1\_matched\_nonlive,

SUM (NVL (ethernet\_discovered, 0)) AS ethernet\_discovered,

SUM (NVL (ethernet\_matched\_live, 0)) AS ethernet\_matched\_live,

SUM (NVL (ethernet\_matched\_nonlive, 0)) AS ethernet\_matched\_nonlive,

SUM (NVL (ethernet\_csr\_matched\_live, 0)) AS ethernet\_csr\_matched\_live,

SUM (NVL (ethernet\_csr\_matched\_non\_live, 0)) AS ethernet\_csr\_matched\_non\_live

FROM mtx\_region\_map mrm LEFT OUTER JOIN t1\_discovered td

ON mrm.vsm\_device\_name\_mtx = td.vsm\_device\_name\_mtx

LEFT OUTER JOIN t1\_matched\_live tm

ON mrm.vsm\_device\_name\_mtx = tm.vsm\_device\_name\_mtx

LEFT OUTER JOIN t1\_matched\_nonlive tum

ON mrm.vsm\_device\_name\_mtx = tum.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_discovered ed

ON mrm.vsm\_device\_name\_mtx = ed.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_matched\_live em

ON mrm.vsm\_device\_name\_mtx = em.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_matched\_nonlive eum

ON mrm.vsm\_device\_name\_mtx = eum.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_csr\_matched\_live emcsr

ON mrm.vsm\_device\_name\_mtx = emcsr.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_csr\_matched\_non\_live eumcsr

ON mrm.vsm\_device\_name\_mtx = eumcsr.vsm\_device\_name\_mtx

GROUP BY ROLLUP (mrm.area, mrm.region)

ORDER BY mrm.area, mrm.region

';

execute immediate sqlStmt;

commit;

sqlStmtT1Comp := 'update NORTEL\_CDMA\_AUDIT\_REG\_SUMM\_WRK set t1\_comp =:x ,t1\_reported =:y where rowid = :a';

sqlStmtEbhComp := 'update NORTEL\_CDMA\_AUDIT\_REG\_SUMM\_WRK set ebh\_comp =:x,ebh\_reported =:y where rowid =:a';

sqlStmtOvlComp := 'update NORTEL\_CDMA\_AUDIT\_REG\_SUMM\_WRK set overall\_comp =:x where rowid =:a';

for cur in updateCompliance

loop

t1Reported := nvl(cur.t1\_discovered,0) - nvl(cur.t1\_matched\_nonlive,0);

ebhReported := nvl(cur.ebh\_discovered,0) - nvl(cur.ebh\_csr\_matched\_non\_live,0);

overallComp :=nvl(cur.t1\_discovered,0) + nvl(cur.ebh\_discovered,0) ;

overallComp := overallComp - nvl(cur.t1\_matched\_nonlive,0) - nvl(cur.ebh\_csr\_matched\_non\_live,0);

if(t1Reported >0) then

t1Comp :=(nvl(cur.t1\_matched\_live,0)/t1Reported) \*100;

t1Comp := round(t1Comp,2);

execute immediate sqlStmtT1Comp using t1Comp,t1Reported,cur.rd;

--dbms\_output.put\_line('Non null '|| t1Comp);

end if;

--JC

if(ebhReported>0) then

ebhComp :=(nvl(cur.ebh\_csr\_matched\_live,0)/ebhReported) \*100;

ebhComp := round(ebhComp,2);

execute immediate sqlStmtEbhComp using ebhComp,ebhReported,cur.rd;

--dbms\_output.put\_line('good EBH '||ebhComp);

end if;

--JC

if(overallComp>0) then

overallComp :=((nvl(cur.t1\_matched\_live,0)+nvl(cur.ebh\_csr\_matched\_live,0))/overallComp) \*100;

overallComp := round(overallComp,2);

execute immediate sqlStmtOvlComp using overallComp,cur.rd;

--dbms\_output.put\_line('good EBH '||overallComp);

end if;

end loop;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlStmt);

dbms\_output.put\_line(SubStr('Error in populate\_cdma\_regional\_summary();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

procedure populate\_cdma\_device\_summary is

cursor updateCompliance is

select summ.rowid rd,summ.\* from NORTEL\_CDMA\_AUDIT\_DEV\_SUMM\_WRK summ;

sqlStmt VARCHAR2(32767);

sqlStmtT1Comp VARCHAR2(32767);

sqlStmtEbhComp VARCHAR2(32767);

sqlStmtOvlComp VARCHAR2(32767);

t1Comp NUMBER;

ebhComp NUMBER;

overallComp NUMBER;

BEGIN

sqlStmt := 'truncate table NORTEL\_CDMA\_AUDIT\_DEV\_SUMM\_WRK';

execute immediate sqlStmt;

sqlStmt := 'insert into NORTEL\_CDMA\_AUDIT\_DEV\_SUMM\_WRK

(area,region,leaf\_domain,vsm\_device\_name\_mtx,switch\_id,t1\_discovered,t1\_matched\_live,t1\_matched\_nonlive,ebh\_discovered,ebh\_matched\_live,ebh\_matched\_nonlive, ebh\_csr\_matched\_live, ebh\_csr\_matched\_non\_live

)

with mtx\_region\_map AS

(SELECT DISTINCT cdmp.area,cdmp.region,cdmp.leaf\_domain\_name, m.vsm\_device\_name\_mtx ,

( SUBSTR (vsm\_device\_name\_mtx, 1, 6)

|| SUBSTR (vsm\_device\_name\_mtx, 11, 2)

) switch\_id

FROM clli\_domain\_map\_v cdmp

RIGHT OUTER JOIN (SELECT \* FROM xng\_reports.bsm\_mtx\_map bmm

WHERE bmm.vsm\_device\_name\_bsm LIKE ''%BSM%''

AND bmm.mtx\_status =''Live'') m

ON substr(m.vsm\_device\_name\_mtx,1,6)=substr(cdmp.clli,1,6)

WHERE cdmp.area <> ''NNO''

),

t1\_discovered AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) t1\_discovered

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,nnvxa.vsm\_device\_name\_bsm,

nnvxa.ne\_dcg\_number, nnvxa.ne\_span\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa,

mtx\_region\_map mrp

WHERE nnvxa.termination\_type = ''T1''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.match\_code IN (''BOTH'', ''NE Only'')

AND nnvxa.vsm\_device\_name\_mtx IN

(mrp.vsm\_device\_name\_mtx))

GROUP BY vsm\_device\_name\_mtx),

ethernet\_discovered AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_discovered

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,nnvxa.vsm\_device\_name\_bsm,

nnvxa.ne\_dcg\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa,

mtx\_region\_map mrp

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.match\_code IN (''BOTH'', ''NE Only'')

AND nnvxa.vsm\_device\_name\_mtx IN

(mrp.vsm\_device\_name\_mtx))

GROUP BY vsm\_device\_name\_mtx),

t1\_matched\_live AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) t1\_matched\_live

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number, nnvxa.ne\_span\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa,

mtx\_region\_map mrp

WHERE nnvxa.termination\_type = ''T1''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH''

AND nnvxa.vsm\_device\_name\_mtx =

mrp.vsm\_device\_name\_mtx

)

GROUP BY vsm\_device\_name\_mtx),

ethernet\_matched\_live AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_matched\_live

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa,

mtx\_region\_map mrp

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH''

AND nnvxa.vsm\_device\_name\_mtx =

mrp.vsm\_device\_name\_mtx)

GROUP BY vsm\_device\_name\_mtx),

t1\_matched\_nonlive AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) t1\_matched\_nonlive

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number, nnvxa.ne\_span\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa,

mtx\_region\_map mrp

WHERE nnvxa.termination\_type = ''T1''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status <> ''Live''

AND nnvxa.match\_code =''BOTH''

AND nnvxa.vsm\_device\_name\_mtx =

mrp.vsm\_device\_name\_mtx

MINUS

SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number, nnvxa.ne\_span\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa,

mtx\_region\_map mrp

WHERE nnvxa.termination\_type = ''T1''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code =''BOTH''

AND nnvxa.vsm\_device\_name\_mtx =

mrp.vsm\_device\_name\_mtx)

GROUP BY vsm\_device\_name\_mtx),

ethernet\_matched\_nonlive AS

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_matched\_nonlive

FROM (SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa,

mtx\_region\_map mrp

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status <> ''Live''

AND nnvxa.match\_code = ''BOTH''

AND nnvxa.vsm\_device\_name\_mtx = mrp.vsm\_device\_name\_mtx

MINUS

SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx,

nnvxa.ne\_dcg\_number

FROM xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa,

mtx\_region\_map mrp

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH''

AND nnvxa.vsm\_device\_name\_mtx = mrp.vsm\_device\_name\_mtx)

GROUP BY vsm\_device\_name\_mtx)

--JC

,

ethernet\_csr\_matched\_live as

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_csr\_matched\_live

FROM ( SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx, nnvxa.ne\_dcg\_number, NNVXA.XNG\_BANDWIDTH

FROM mtx\_region\_map mrp, xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

join VZWNET.CIRC\_PATH\_INST cpi on NNVXA.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status = ''Live''

AND nnvxa.match\_code = ''BOTH''

AND nnvxa.vsm\_device\_name\_mtx = mrp.vsm\_device\_name\_mtx

and NNVXA.XNG\_BANDWIDTH like ''%Mbps''

and EI.TYPE = ''CSR'' )

GROUP BY vsm\_device\_name\_mtx)

,

ethernet\_csr\_matched\_non\_live as

(SELECT vsm\_device\_name\_mtx, COUNT (1) ethernet\_csr\_matched\_non\_live

FROM ( SELECT DISTINCT nnvxa.vsm\_device\_name\_mtx, nnvxa.ne\_dcg\_number, NNVXA.XNG\_BANDWIDTH

FROM mtx\_region\_map mrp, xng\_reports.nortel\_ne\_vs\_xng\_audit\_wrk nnvxa

join VZWNET.CIRC\_PATH\_INST cpi on NNVXA.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

WHERE nnvxa.termination\_type = ''Ethernet''

AND nnvxa.ne\_span\_type = ''VOICE\_SPAN''

AND nnvxa.xng\_path\_status <> ''Live''

AND nnvxa.match\_code = ''BOTH''

AND nnvxa.vsm\_device\_name\_mtx = mrp.vsm\_device\_name\_mtx

and NNVXA.XNG\_BANDWIDTH like ''%Mbps''

and EI.TYPE = ''CSR'' )

GROUP BY vsm\_device\_name\_mtx)

--JC

SELECT mrm.area,mrm.region,mrm.leaf\_domain\_name,mrm.vsm\_device\_name\_mtx, mrm.switch\_id,

NVL (t1\_discovered, 0) AS t1\_discovered,

NVL (t1\_matched\_live, 0) AS t1\_matched\_live,

NVL (t1\_matched\_nonlive, 0) AS t1\_matched\_nonlive,

NVL (ethernet\_discovered, 0) AS ethernet\_discovered,

NVL (ethernet\_matched\_live, 0) AS ethernet\_matched\_live,

NVL (ethernet\_matched\_nonlive, 0) AS ethernet\_matched\_nonlive,

NVL (ethernet\_csr\_matched\_live, 0) AS ethernet\_csr\_matched\_live,

NVL (ethernet\_csr\_matched\_non\_live, 0) AS ethernet\_csr\_matched\_non\_live

FROM mtx\_region\_map mrm LEFT OUTER JOIN t1\_discovered td

ON mrm.vsm\_device\_name\_mtx = td.vsm\_device\_name\_mtx

LEFT OUTER JOIN t1\_matched\_live tm

ON mrm.vsm\_device\_name\_mtx = tm.vsm\_device\_name\_mtx

LEFT OUTER JOIN t1\_matched\_nonlive tum

ON mrm.vsm\_device\_name\_mtx = tum.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_discovered ed

ON mrm.vsm\_device\_name\_mtx = ed.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_matched\_live em

ON mrm.vsm\_device\_name\_mtx = em.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_matched\_nonlive eum

ON mrm.vsm\_device\_name\_mtx = eum.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_csr\_matched\_live emcsr

ON mrm.vsm\_device\_name\_mtx = emcsr.vsm\_device\_name\_mtx

LEFT OUTER JOIN ethernet\_csr\_matched\_non\_live eumcsr

ON mrm.vsm\_device\_name\_mtx = eumcsr.vsm\_device\_name\_mtx

ORDER BY mrm.vsm\_device\_name\_mtx

';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Device Summary created');

sqlStmtT1Comp := 'update NORTEL\_CDMA\_AUDIT\_DEV\_SUMM\_WRK set t1\_comp =:x where rowid = :a';

sqlStmtEbhComp := 'update NORTEL\_CDMA\_AUDIT\_DEV\_SUMM\_WRK set ebh\_comp =:x where rowid = :a';

sqlStmtOvlComp := 'update NORTEL\_CDMA\_AUDIT\_DEV\_SUMM\_WRK set overall\_comp =:x where rowid = :a';

for cur in updateCompliance

loop

t1Comp := nvl(cur.t1\_discovered,0) - nvl(cur.t1\_matched\_nonlive,0);

--ebhComp := nvl(cur.ebh\_discovered,0) - nvl(cur.ebh\_matched\_nonlive,0);

ebhComp := nvl(cur.ebh\_discovered,0) - nvl(cur.ebh\_csr\_matched\_non\_live,0);

overallComp :=nvl(cur.t1\_discovered,0) + nvl(cur.ebh\_discovered,0) ;

--overallComp := overallComp - nvl(cur.t1\_matched\_nonlive,0) - nvl(cur.ebh\_matched\_nonlive,0);

overallComp := overallComp - nvl(cur.t1\_matched\_nonlive,0) - nvl(cur.ebh\_csr\_matched\_non\_live,0);

if(t1Comp >0) then

t1Comp :=(nvl(cur.t1\_matched\_live,0)/t1Comp) \*100;

t1Comp := round(t1Comp,2);

execute immediate sqlStmtT1Comp using t1Comp,cur.rd;

--dbms\_output.put\_line('Non null '|| t1Comp);

end if;

if(ebhComp>0) then

ebhComp :=(nvl(cur.ebh\_csr\_matched\_live,0)/ebhComp) \*100;

ebhComp := round(ebhComp,2);

execute immediate sqlStmtEbhComp using ebhComp,cur.rd;

--dbms\_output.put\_line('good EBH '||ebhComp);

end if;

if(overallComp>0) then

overallComp :=((nvl(cur.t1\_matched\_live,0)+nvl(cur.ebh\_csr\_matched\_live,0))/overallComp) \*100;

overallComp := round(overallComp,2);

execute immediate sqlStmtOvlComp using overallComp,cur.rd;

--dbms\_output.put\_line('good EBH '||overallComp);

end if;

end loop;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlStmt);

dbms\_output.put\_line(SubStr('Error in populate\_cdma\_device\_summary();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

procedure populate\_evdo\_regional\_summary is

cursor updateCompliance is

select summ.rowid rd,summ.\* from NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK summ;

cursor totUpEbhSpans is

WITH

-- Total EBH spans

LIVE\_EBH\_PORTS as

(

select distinct npi.DOM\_SEQ\_NUM

from NORTEL\_PORT\_INVENTORY npi

where npi.PORT\_TYPE = 'Ethernet'

and npi.STATUS = 'Up'

)

,

TOTAL\_LIVE\_EBH\_SPANS as

(

select cdm.leaf\_domain\_inst\_id,

ndi.\*

from NORTEL\_DOM\_INVENTORY ndi

join LIVE\_EBH\_PORTS npi

on ndi.SEQ\_NUM = npi.DOM\_SEQ\_NUM

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.clli = substr(ndi.EMS\_NAME, 1, 6)

)

select area, region,

count(ebh.dcg\_number) tot\_up\_ebh\_spans

from domains\_leaf\_reporting dlr

join TOTAL\_LIVE\_EBH\_SPANS ebh

on ebh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region)

;

cursor MatchedUpLiveT1Spans is

WITH

MATCHED\_LIVE\_T1\_SPANS as

(

select cdm.leaf\_domain\_inst\_id

, aud.NE\_DOM\_IP

, aud.ne\_dcg\_number

, aud.ne\_slot\_number

, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.clli = substr(aud.VSM\_device\_name\_EMS, 1, 6)

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'T1'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

)

select area, region

, count(lt1.ne\_dcg\_number ) tot\_up\_t1\_spans\_w\_l\_paths

from domains\_leaf\_reporting dlr

join MATCHED\_LIVE\_T1\_SPANS lt1

on lt1.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region)

;

cursor MatchedUpLiveEbhSpans is

WITH

MATCHED\_LIVE\_EBH\_SPANS as

(

select cdm.leaf\_domain\_inst\_id

, aud.NE\_DOM\_IP

, aud.ne\_dcg\_number

, aud.ne\_slot\_number

, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.CLLI = substr(aud.VSM\_device\_name\_EMS, 1, 6)

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

)

select area, region

, count(lebh.ne\_dcg\_number) tot\_up\_ebh\_spans\_w\_l\_paths

from domains\_leaf\_reporting dlr

join MATCHED\_LIVE\_EBH\_SPANS lebh

on lebh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region)

;

cursor MatchedUpLiveEbhSpansCSR is

WITH

cdm\_map as

(

select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map

),

MATCHED\_LIVE\_EBH\_CSR\_SPANS as

(

select cdm.leaf\_domain\_inst\_id

, aud.NE\_DOM\_IP

, aud.ne\_dcg\_number

, aud.ne\_slot\_number

, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join cdm\_map cdm on cdm.clli = substr(aud.VSM\_device\_name\_EMS, 1, 6)

join VZWNET.CIRC\_PATH\_INST cpi on aud.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

and aud.XNG\_BANDWIDTH like '%Mbps'

and EI.TYPE = 'CSR'

)

select area, region

, count(lebh.ne\_dcg\_number) tot\_up\_ebh\_spans\_w\_l\_pathscsr

from domains\_leaf\_reporting dlr

join MATCHED\_LIVE\_EBH\_CSR\_SPANS lebh

on lebh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region)

;

cursor MatchedUpNonLiveT1Spans is

-- NE Up spans matching non-Live T1 spans

WITH

MATCHED\_LIVE\_T1\_SPANS as

(

select aud.NE\_DOM\_IP, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'T1'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

)

,

MATCHED\_NLIVE\_T1\_SPANS as

(

-- all Up spans that have non-live paths on them

-- minus Up spans that have live paths

select aud.NE\_DOM\_IP, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'T1'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS <> 'Live'

minus

select aud.NE\_DOM\_IP, aud.NE\_SPAN\_NUMBER

from MATCHED\_LIVE\_T1\_SPANS aud

)

,

MATCHED\_NL\_T1\_SPANS\_W\_DOMAIN AS

(

select cdm.leaf\_domain\_inst\_id

, aud.NE\_DOM\_IP

, aud.ne\_dcg\_number

, aud.ne\_slot\_number

, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

JOIN MATCHED\_NLIVE\_T1\_SPANS nl

on aud.NE\_DOM\_IP = nl.NE\_DOM\_IP

and aud.NE\_SPAN\_NUMBER = nl.NE\_SPAN\_NUMBER

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.CLLI = substr(aud.VSM\_device\_name\_EMS, 1, 6)

)

select area, region

, count(nlt1.ne\_dcg\_number) tot\_up\_t1\_spans\_w\_nl\_paths

from domains\_leaf\_reporting dlr

join MATCHED\_NL\_T1\_SPANS\_W\_DOMAIN nlt1

on nlt1.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region)

;

cursor MatchedUpNonLiveEbhSpans is

-- NE Up spans matching non-Live EBH spans

WITH

MATCHED\_LIVE\_EBH\_SPANS as

(

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from nortel\_ne\_vs\_xng\_audit\_wrk aud

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

)

,

MATCHED\_NLIVE\_EBH\_SPANS as

(

-- all Up spans that have non-live paths on them

-- minus Up spans that have live paths

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from nortel\_ne\_vs\_xng\_audit\_wrk aud

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS <> 'Live'

minus

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from MATCHED\_LIVE\_EBH\_SPANS aud

)

,

MATCHED\_NL\_EBH\_SPANS\_W\_DOMAIN AS

(

select cdm.leaf\_domain\_inst\_id

--, nl.VSM\_DEVICE\_NAME\_EMS

, nl.NE\_DOM\_IP

, nl.ne\_dcg\_number

, nl.ne\_slot\_number

, null NE\_SPAN\_NUMBER

from MATCHED\_NLIVE\_EBH\_SPANS nl

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.clli = substr(nl.VSM\_device\_name\_EMS, 1, 6)

)

select area, region

, count(nlebh.ne\_dcg\_number) tot\_up\_ebh\_spans\_w\_nl\_paths

from domains\_leaf\_reporting dlr

join MATCHED\_NL\_EBH\_SPANS\_W\_DOMAIN nlEbh

on nlEbh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region)

;

cursor MatchedUpNonLiveEbhSpansCSR is

-- NE Up spans matching non-Live EBH spans with CSR Ports

WITH

MATCHED\_LIVE\_EBH\_SPANS\_CSR as

(

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join VZWNET.CIRC\_PATH\_INST cpi on aud.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

and aud.XNG\_BANDWIDTH like '%Mbps'

and EI.TYPE = 'CSR'

)

,

MATCHED\_NLIVE\_EBH\_SPANS\_CSR as

(

-- all Up spans that have non-live paths on them

-- minus Up spans that have live paths

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join VZWNET.CIRC\_PATH\_INST cpi on aud.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS <> 'Live'

and aud.XNG\_BANDWIDTH like '%Mbps'

and EI.TYPE = 'CSR'

minus

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from MATCHED\_LIVE\_EBH\_SPANS\_CSR aud

)

,

MATCHED\_NL\_EBH\_SPANS\_W\_DOM\_CSR AS

(

select cdm.leaf\_domain\_inst\_id

--, nl.VSM\_DEVICE\_NAME\_EMS

, nl.NE\_DOM\_IP

, nl.ne\_dcg\_number

, nl.ne\_slot\_number

, null NE\_SPAN\_NUMBER

from MATCHED\_NLIVE\_EBH\_SPANS\_CSR nl

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.clli = substr(nl.VSM\_device\_name\_EMS, 1, 6)

)

select area, region

, count(nlebh.ne\_dcg\_number) tot\_up\_ebh\_spans\_w\_nl\_pathscsr

from domains\_leaf\_reporting dlr

join MATCHED\_NL\_EBH\_SPANS\_W\_DOM\_CSR nlEbh

on nlEbh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region)

;

-- NT\_EVDO\_BTS\_TERM\_REG\_SUMM varchar2(30) := 'Nortel\_EVDO\_BTS\_TERM\_REG\_SUMM';

sqlStmt VARCHAR2(32767);

sqlStmtVZW VARCHAR2(32767);

sqlStmtArea VARCHAR2(32767);

sqlStmtRegion VARCHAR2(32767);

sqlStmtT1Comp VARCHAR2(32767);

sqlStmtEbhComp VARCHAR2(32767);

sqlStmtOvlComp VARCHAR2(32767);

t1Comp NUMBER;

ebhComp NUMBER;

overallComp NUMBER;

t1Reported NUMBER;

ebhReported NUMBER;

BEGIN

sqlStmt := 'truncate table NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK';

execute immediate sqlStmt;

sqlStmt := 'insert into NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK

(area,region,tot\_up\_t1\_spans,tot\_up\_t1\_spans\_w\_l\_paths,tot\_up\_t1\_spans\_w\_nl\_paths,

tot\_up\_ebh\_spans,tot\_up\_ebh\_spans\_w\_l\_paths,tot\_up\_ebh\_spans\_w\_nl\_paths,tot\_up\_ebh\_spans\_w\_l\_pathscsr,tot\_up\_ebh\_spans\_w\_nl\_pathscsr)

-- Total Up T1 spans

WITH

TOTAL\_LIVE\_T1\_SPANS as

(

select cdm.leaf\_domain\_inst\_id,

ndi.\*

from NORTEL\_DOM\_INVENTORY ndi

join NORTEL\_PORT\_INVENTORY npi

on ndi.SEQ\_NUM = npi.DOM\_SEQ\_NUM

and npi.PORT\_TYPE = ''T1E1''

and npi.STATUS = ''Up''

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.clli = substr(ndi.EMS\_NAME, 1, 6)

)

select area, region, count(t1.dcg\_number) tot\_up\_t1\_spans, null, null, null, null, null, null, null

from domains\_leaf\_reporting dlr

left outer join TOTAL\_LIVE\_T1\_SPANS t1

on t1.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

where dlr.DOMAIN\_INST\_ID not in(1012, 1016) -- NNO and OSS

group by rollup(dlr.area, dlr.region)

order by dlr.area, dlr.region

';

execute immediate sqlStmt;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans = :x where area is null and region is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans = :x where area = :a and region is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans = :x where area = :a and region = :b';

for cur in totUpEbhSpans

loop

if (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_l\_paths = :x where area is null and region is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_l\_paths = :x where area = :a and region is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_l\_paths = :x where area = :a and region = :b ';

for cur in matchedUpLiveT1Spans

loop

if (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_t1\_spans\_w\_l\_paths, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_t1\_spans\_w\_l\_paths, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_t1\_spans\_w\_l\_paths;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_paths = :x where area is null and region is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_paths = :x where area = :a and region is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_paths = :x where area = :a and region = :b ';

for cur in matchedUpLiveEbhSpans

loop

if (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans\_w\_l\_paths, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans\_w\_l\_paths, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans\_w\_l\_paths;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_pathscsr = :x where area is null and region is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_pathscsr = :x where area = :a and region is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_pathscsr = :x where area = :a and region = :b ';

for cur in matchedUpLiveEbhSpansCSR

loop

if (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_nl\_paths = :x where area is null and region is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_nl\_paths = :x where area = :a and region is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_nl\_paths = :x where area = :a and region = :b ';

for cur in matchedUpNonLiveT1Spans

loop

if (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_t1\_spans\_w\_nl\_paths, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_t1\_spans\_w\_nl\_paths, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_t1\_spans\_w\_nl\_paths;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_paths = :x where area is null and region is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_paths = :x where area = :a and region is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_paths = :x where area = :a and region = :b ';

for cur in matchedUpNonLiveEbhSpans

loop

if (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans\_w\_nl\_paths, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans\_w\_nl\_paths, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans\_w\_nl\_paths;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_pathscsr = :x where area is null and region is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_pathscsr = :x where area = :a and region is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_pathscsr = :x where area = :a and region = :b ';

for cur in matchedUpNonLiveEbhSpansCSR

loop

if (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr;

end if;

end loop;

commit;

sqlStmtT1Comp := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set t1\_comp =:x,t1\_reported =:y where rowid = :a';

sqlStmtEbhComp := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set ebh\_comp =:x,ebh\_reported =:y where rowid = :a';

sqlStmtOvlComp := 'update NT\_EVDO\_BTS\_TERM\_REG\_SUMM\_WRK set overall\_comp =:x where rowid = :a';

for cur in updateCompliance

loop

t1Reported := nvl(cur.tot\_up\_t1\_spans,0) - nvl(cur.tot\_up\_t1\_spans\_w\_nl\_paths,0);

--ebhReported := nvl(cur.tot\_up\_ebh\_spans,0) - nvl(cur.tot\_up\_ebh\_spans\_w\_nl\_paths,0);

ebhReported := nvl(cur.tot\_up\_ebh\_spans,0) - nvl(cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr,0);

overallComp :=nvl(cur.tot\_up\_t1\_spans,0) + nvl(cur.tot\_up\_ebh\_spans,0) ;

--overallComp := overallComp - nvl(cur.tot\_up\_t1\_spans\_w\_nl\_paths,0) - nvl(cur.tot\_up\_ebh\_spans\_w\_nl\_paths,0);

overallComp := overallComp - nvl(cur.tot\_up\_t1\_spans\_w\_nl\_paths,0) - nvl(cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr,0);

if(t1Reported >0) then

t1Comp :=(nvl(cur.tot\_up\_t1\_spans\_w\_l\_paths,0)/t1Reported) \*100;

t1Comp := round(t1Comp,2);

execute immediate sqlStmtT1Comp using t1Comp,t1Reported,cur.rd;

--dbms\_output.put\_line('Non null '|| t1Comp);

end if;

if(ebhReported>0) then

--ebhComp :=(nvl(cur.tot\_up\_ebh\_spans\_w\_l\_paths,0)/ebhReported) \*100;

ebhComp :=(nvl(cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr,0)/ebhReported) \*100;

ebhComp := round(ebhComp,2);

execute immediate sqlStmtEbhComp using ebhComp,ebhReported,cur.rd;

--dbms\_output.put\_line('good EBH '||ebhComp);

end if;

if(overallComp>0) then

--overallComp :=((nvl(cur.tot\_up\_t1\_spans\_w\_l\_paths,0)+nvl(cur.tot\_up\_ebh\_spans\_w\_l\_paths,0))/overallComp) \*100;

overallComp :=((nvl(cur.tot\_up\_t1\_spans\_w\_l\_paths,0)+nvl(cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr,0))/overallComp) \*100;

overallComp := round(overallComp,2);

execute immediate sqlStmtOvlComp using overallComp,cur.rd;

--dbms\_output.put\_line('good EBH '||overallComp);

end if;

end loop;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlStmt);

dbms\_output.put\_line(SubStr('Error in populate\_evdo\_regional\_summary();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

Procedure populate\_evdo\_device\_summary is

cursor updateCompliance is

select summ.rowid rd,summ.\* from NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK summ;

cursor totUpEbhSpans is

WITH

-- Total EBH spans

LIVE\_EBH\_PORTS as

(

select distinct npi.DOM\_SEQ\_NUM

from NORTEL\_PORT\_INVENTORY npi

where npi.PORT\_TYPE = 'Ethernet'

and npi.STATUS = 'Up'

)

,

TOTAL\_LIVE\_EBH\_SPANS as

(

select cdm.leaf\_domain\_inst\_id,

ndi.\*

from NORTEL\_DOM\_INVENTORY ndi

join LIVE\_EBH\_PORTS npi

on ndi.SEQ\_NUM = npi.DOM\_SEQ\_NUM

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.CLLI = substr(ndi.EMS\_NAME, 1, 6)

)

select area, region, ebh.ems\_name vsm\_device\_name\_ems

, count(ebh.dcg\_number) tot\_up\_ebh\_spans

from domains\_leaf\_reporting dlr

join TOTAL\_LIVE\_EBH\_SPANS ebh

on ebh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region, ems\_name)

;

cursor MatchedUpLiveT1Spans is

WITH

MATCHED\_LIVE\_T1\_SPANS as

(

select cdm.leaf\_domain\_inst\_id

, aud.vsm\_device\_name\_ems

, aud.NE\_DOM\_IP

, aud.ne\_dcg\_number

, aud.ne\_slot\_number

, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.CLLI = substr(aud.VSM\_device\_name\_EMS, 1, 6)

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'T1'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

)

select area, region, vsm\_device\_name\_ems

, count(lt1.ne\_dcg\_number ) tot\_up\_t1\_spans\_w\_l\_paths

from domains\_leaf\_reporting dlr

join MATCHED\_LIVE\_T1\_SPANS lt1

on lt1.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region, vsm\_device\_name\_ems)

;

cursor MatchedUpLiveEbhSpans is

WITH

MATCHED\_LIVE\_EBH\_SPANS as

(

select cdm.leaf\_domain\_inst\_id

, vsm\_device\_name\_ems

, aud.NE\_DOM\_IP

, aud.ne\_dcg\_number

, aud.ne\_slot\_number

, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.CLLI = substr(aud.VSM\_device\_name\_EMS, 1, 6)

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

)

select area, region, vsm\_device\_name\_ems

, count(lebh.ne\_dcg\_number) tot\_up\_ebh\_spans\_w\_l\_paths

from domains\_leaf\_reporting dlr

join MATCHED\_LIVE\_EBH\_SPANS lebh

on lebh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region, vsm\_device\_name\_ems)

;

cursor MatchedUpLiveEbhSpansCSR is

WITH

cdm\_map as

(

select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map

),

MATCHED\_LIVE\_EBH\_CSR\_SPANS as

(

select cdm.leaf\_domain\_inst\_id

, vsm\_device\_name\_ems

, aud.NE\_DOM\_IP

, aud.ne\_dcg\_number

, aud.ne\_slot\_number

, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join cdm\_map cdm on cdm.clli = substr(aud.VSM\_device\_name\_EMS, 1, 6)

join VZWNET.CIRC\_PATH\_INST cpi on aud.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

and aud.XNG\_BANDWIDTH like '%Mbps'

and EI.TYPE = 'CSR'

)

select area, region, vsm\_device\_name\_ems

, count(lebh.ne\_dcg\_number) tot\_up\_ebh\_spans\_w\_l\_pathscsr

from domains\_leaf\_reporting dlr

join MATCHED\_LIVE\_EBH\_CSR\_SPANS lebh

on lebh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region, vsm\_device\_name\_ems)

;

cursor MatchedUpNonLiveT1Spans is

-- NE Up spans matching non-Live T1 spans

WITH

MATCHED\_LIVE\_T1\_SPANS as

(

select aud.NE\_DOM\_IP, aud.NE\_SPAN\_NUMBER, vsm\_device\_name\_ems

from nortel\_ne\_vs\_xng\_audit\_wrk aud

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'T1'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

)

,

MATCHED\_NLIVE\_T1\_SPANS as

(

-- all Up spans that have non-live paths on them

-- minus Up spans that have live paths

select aud.NE\_DOM\_IP, aud.NE\_SPAN\_NUMBER, vsm\_device\_name\_ems

from nortel\_ne\_vs\_xng\_audit\_wrk aud

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'T1'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS <> 'Live'

minus

select aud.NE\_DOM\_IP, aud.NE\_SPAN\_NUMBER, vsm\_device\_name\_ems

from MATCHED\_LIVE\_T1\_SPANS aud

)

,

MATCHED\_NL\_T1\_SPANS\_W\_DOMAIN AS

(

select cdm.leaf\_domain\_inst\_id

, aud.vsm\_device\_name\_ems

, aud.NE\_DOM\_IP

, aud.ne\_dcg\_number

, aud.ne\_slot\_number

, aud.NE\_SPAN\_NUMBER

from nortel\_ne\_vs\_xng\_audit\_wrk aud

JOIN MATCHED\_NLIVE\_T1\_SPANS nl

on aud.NE\_DOM\_IP = nl.NE\_DOM\_IP

and aud.NE\_SPAN\_NUMBER = nl.NE\_SPAN\_NUMBER

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.clli = substr(aud.VSM\_device\_name\_EMS, 1, 6)

)

select area, region, vsm\_device\_name\_ems

, count(nlt1.ne\_dcg\_number) tot\_up\_t1\_spans\_w\_nl\_paths

from domains\_leaf\_reporting dlr

join MATCHED\_NL\_T1\_SPANS\_W\_DOMAIN nlt1

on nlt1.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region, vsm\_device\_name\_ems)

;

cursor MatchedUpNonLiveEbhSpans is

-- NE Up spans matching non-Live EBH spans

WITH

MATCHED\_LIVE\_EBH\_SPANS as

(

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from nortel\_ne\_vs\_xng\_audit\_wrk aud

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

)

,

MATCHED\_NLIVE\_EBH\_SPANS as

(

-- all Up spans that have non-live paths on them

-- minus Up spans that have live paths

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from nortel\_ne\_vs\_xng\_audit\_wrk aud

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS <> 'Live'

minus

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from MATCHED\_LIVE\_EBH\_SPANS aud

)

,

MATCHED\_NL\_EBH\_SPANS\_W\_DOMAIN AS

(

select cdm.leaf\_domain\_inst\_id

, nl.vsm\_device\_name\_ems

, nl.NE\_DOM\_IP

, nl.ne\_dcg\_number

, nl.ne\_slot\_number

, null NE\_SPAN\_NUMBER

from MATCHED\_NLIVE\_EBH\_SPANS nl

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.clli = substr(nl.VSM\_device\_name\_EMS, 1, 6)

)

select area, region, vsm\_device\_name\_ems

, count(nlebh.ne\_dcg\_number) tot\_up\_ebh\_spans\_w\_nl\_paths

from domains\_leaf\_reporting dlr

join MATCHED\_NL\_EBH\_SPANS\_W\_DOMAIN nlEbh

on nlEbh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region, vsm\_device\_name\_ems)

;

cursor MatchedUpNonLiveEbhSpansCSR is

-- NE Up spans matching non-Live EBH spans with CSR ports

WITH

MATCHED\_LIVE\_EBH\_SPANS\_CSR as

(

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join VZWNET.CIRC\_PATH\_INST cpi on aud.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS = 'Live'

and aud.XNG\_BANDWIDTH like '%Mbps'

and EI.TYPE = 'CSR'

)

,

MATCHED\_NLIVE\_EBH\_SPANS\_CSR as

(

-- all Up spans that have non-live paths on them

-- minus Up spans that have live paths

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from nortel\_ne\_vs\_xng\_audit\_wrk aud

join VZWNET.CIRC\_PATH\_INST cpi on aud.CIRC\_PATH\_INST\_ID = CPI.CIRC\_PATH\_INST\_ID

join VZWNET.EPA p on cpi.CIRC\_PATH\_INST\_ID = P.CIRC\_PATH\_INST\_ID

join VZWNET.EQUIP\_INST ei on EI.EQUIP\_INST\_ID = P.EQUIP\_INST\_ID

where aud.MATCH\_CODE = 'BOTH'

and aud.NE\_SPAN\_TYPE = 'DATA\_SPAN'

and aud.TERMINATION\_TYPE = 'Ethernet'

and aud.NE\_STATUS = 'Up'

and aud.XNG\_PATH\_STATUS <> 'Live'

and aud.XNG\_BANDWIDTH like '%Mbps'

and EI.TYPE = 'CSR'

minus

select aud.VSM\_DEVICE\_NAME\_EMS, aud.NE\_DOM\_IP, aud.ne\_dcg\_number, aud.ne\_slot\_number

from MATCHED\_LIVE\_EBH\_SPANS\_CSR aud

)

,

MATCHED\_NL\_EBH\_SPANS\_W\_DOM\_CSR AS

(

select cdm.leaf\_domain\_inst\_id

, nl.VSM\_DEVICE\_NAME\_EMS

, nl.NE\_DOM\_IP

, nl.ne\_dcg\_number

, nl.ne\_slot\_number

, null NE\_SPAN\_NUMBER

from MATCHED\_NLIVE\_EBH\_SPANS\_CSR nl

join (select distinct leaf\_domain\_inst\_id,substr(CLLI, 1, 6) clli from clli\_domain\_map )cdm

on cdm.clli = substr(nl.VSM\_device\_name\_EMS, 1, 6)

)

select area, region, vsm\_device\_name\_ems

, count(nlebh.ne\_dcg\_number) tot\_up\_ebh\_spans\_w\_nl\_pathscsr

from domains\_leaf\_reporting dlr

join MATCHED\_NL\_EBH\_SPANS\_W\_DOM\_CSR nlEbh

on nlEbh.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

group by rollup (dlr.area, dlr.region, vsm\_device\_name\_ems)

;

sqlStmt VARCHAR2(32767);

sqlStmtVZW VARCHAR2(32767);

sqlStmtArea VARCHAR2(32767);

sqlStmtRegion VARCHAR2(32767);

sqlStmtEms VARCHAR2(32767);

sqlStmtT1Comp VARCHAR2(32767);

sqlStmtEbhComp VARCHAR2(32767);

sqlStmtOvlComp VARCHAR2(32767);

t1Comp NUMBER;

ebhComp NUMBER;

overallComp NUMBER;

--NT\_EVDO\_BTS\_TERM\_EMS\_SUMM varchar2(30) := 'Nortel\_EVDO\_BTS\_TERM\_EMS\_SUMM';

BEGIN

dbms\_output.put\_line(' ');

dbms\_output.put\_line(' ');

dbms\_output.put\_line(' ');

dbms\_output.put\_line(' ');

sqlStmt := 'truncate table NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK';

execute immediate sqlStmt;

sqlStmt := 'insert into NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK

(area,region,leaf\_domain\_inst\_id,vsm\_device\_name\_ems,tot\_up\_t1\_spans,tot\_up\_t1\_spans\_w\_l\_paths,tot\_up\_t1\_spans\_w\_nl\_paths,

tot\_up\_ebh\_spans,tot\_up\_ebh\_spans\_w\_l\_paths,tot\_up\_ebh\_spans\_w\_nl\_paths, tot\_up\_ebh\_spans\_w\_l\_pathscsr,tot\_up\_ebh\_spans\_w\_nl\_pathscsr)

-- Total Up T1 spans

WITH

ALL\_EMS as

(

select distinct dlr.area, dlr.region, dlr.cdm.leaf\_domain\_inst\_id, ndi.EMS\_NAME

from domains\_leaf\_reporting dlr

left outer join clli\_domain\_map cdm

on cdm.leaf\_domain\_inst\_id = dlr.DOMAIN\_INST\_ID

left outer join NORTEL\_DOM\_INVENTORY ndi

on substr(CLLI, 1, 6) = substr(ndi.EMS\_NAME, 1, 6)

where dlr.DOMAIN\_INST\_ID not in(1012, 1016) -- NNO and OSS

)

,

TOTAL\_LIVE\_T1\_SPANS as

(

select ndi.\*

from NORTEL\_DOM\_INVENTORY ndi

join NORTEL\_PORT\_INVENTORY npi

on ndi.SEQ\_NUM = npi.DOM\_SEQ\_NUM

and npi.PORT\_TYPE = ''T1E1''

and npi.STATUS = ''Up''

)

select area, region, dlr.leaf\_domain\_inst\_id

, dlr.ems\_name, count(t1.dcg\_number) tot\_up\_t1\_spans, null, null, null, null, null, null, null

from ALL\_EMS dlr

left outer join TOTAL\_LIVE\_T1\_SPANS t1

on t1.ems\_name = dlr.ems\_name

group by rollup (area, region, dlr.leaf\_domain\_inst\_id, dlr.ems\_name)

order by area, region, dlr.leaf\_domain\_inst\_id, dlr.ems\_name

';

execute immediate sqlStmt;

commit;

-- dont need leaf domain rollups. delete those rows

sqlStmt := 'delete NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK where leaf\_domain\_inst\_id is not null and vsm\_device\_name\_ems is null';

execute immediate sqlStmt;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans = :x where area is null and region is null and vsm\_device\_name\_ems is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans = :x where area = :a and region is null and vsm\_device\_name\_ems is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans = :x where area = :a and region = :b and vsm\_device\_name\_ems is null';

sqlStmtEms := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans = :x where area = :a and region = :b and vsm\_device\_name\_ems = :c';

for cur in totUpEbhSpans

loop

--dbms\_output.put\_line('In totUpEbhSpans');

if (cur.vsm\_device\_name\_ems is not null) then

execute immediate sqlStmtEms using cur.tot\_up\_ebh\_spans, cur.area, cur.region, cur.vsm\_device\_name\_ems;

elsif (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_l\_paths = :x where area is null and region is null and vsm\_device\_name\_ems is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_l\_paths = :x where area = :a and region is null and vsm\_device\_name\_ems is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_l\_paths = :x where area = :a and region = :b and vsm\_device\_name\_ems is null';

sqlStmtEms := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_l\_paths = :x where area = :a and region = :b and vsm\_device\_name\_ems = :c';

for cur in matchedUpLiveT1Spans

loop

if (cur.vsm\_device\_name\_ems is not null) then

--dbms\_output.put\_line('In matchedUpLiveT1Spans. '||sqlStmtEms||': '||cur.tot\_up\_t1\_spans\_w\_l\_paths||' , '|| cur.area||' , '|| cur.region ||' , '||cur.vsm\_device\_name\_ems);

execute immediate sqlStmtEms using cur.tot\_up\_t1\_spans\_w\_l\_paths, cur.area, cur.region, cur.vsm\_device\_name\_ems;

elsif (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_t1\_spans\_w\_l\_paths, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_t1\_spans\_w\_l\_paths, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_t1\_spans\_w\_l\_paths;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_paths = :x where area is null and region is null and vsm\_device\_name\_ems is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_paths = :x where area = :a and region is null and vsm\_device\_name\_ems is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_paths = :x where area = :a and region = :b and vsm\_device\_name\_ems is null';

sqlStmtEms := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_paths = :x where area = :a and region = :b and vsm\_device\_name\_ems = :c';

for cur in matchedUpLiveEbhSpans

loop

dbms\_output.put\_line('In matchedUpLiveEbhSpans. '||sqlStmtEms||': '||cur.tot\_up\_ebh\_spans\_w\_l\_paths||' , '|| cur.area||' , '|| cur.region ||' , '||cur.vsm\_device\_name\_ems);

if (cur.vsm\_device\_name\_ems is not null) then

execute immediate sqlStmtEms using cur.tot\_up\_ebh\_spans\_w\_l\_paths, cur.area, cur.region, cur.vsm\_device\_name\_ems;

elsif (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans\_w\_l\_paths, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans\_w\_l\_paths, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans\_w\_l\_paths;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_pathscsr = :x where area is null and region is null and vsm\_device\_name\_ems is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_pathscsr = :x where area = :a and region is null and vsm\_device\_name\_ems is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_pathscsr = :x where area = :a and region = :b and vsm\_device\_name\_ems is null';

sqlStmtEms := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_l\_pathscsr = :x where area = :a and region = :b and vsm\_device\_name\_ems = :c';

for cur in matchedUpLiveEbhSpansCSR

loop

dbms\_output.put\_line('In matchedUpLiveEbhSpansCSR. '||sqlStmtEms||': '||cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr||' , '|| cur.area||' , '|| cur.region ||' , '||cur.vsm\_device\_name\_ems);

if (cur.vsm\_device\_name\_ems is not null) then

execute immediate sqlStmtEms using cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr, cur.area, cur.region, cur.vsm\_device\_name\_ems;

elsif (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans\_w\_l\_pathscsr;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_nl\_paths = :x where area is null and region is null and vsm\_device\_name\_ems is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_nl\_paths = :x where area = :a and region is null and vsm\_device\_name\_ems is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_nl\_paths = :x where area = :a and region = :b and vsm\_device\_name\_ems is null';

sqlStmtEms := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_t1\_spans\_w\_nl\_paths = :x where area = :a and region = :b and vsm\_device\_name\_ems = :c';

for cur in matchedUpNonLiveT1Spans

loop

if (cur.vsm\_device\_name\_ems is not null) then

dbms\_output.put\_line('In matchedUpNonLiveT1Spans. '||sqlStmtEms||': '||cur.tot\_up\_t1\_spans\_w\_nl\_paths||' , '|| cur.area||' , '|| cur.region ||' , '||cur.vsm\_device\_name\_ems);

execute immediate sqlStmtEms using cur.tot\_up\_t1\_spans\_w\_nl\_paths, cur.area, cur.region, cur.vsm\_device\_name\_ems;

elsif (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_t1\_spans\_w\_nl\_paths, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_t1\_spans\_w\_nl\_paths, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_t1\_spans\_w\_nl\_paths;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_paths = :x where area is null and region is null and vsm\_device\_name\_ems is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_paths = :x where area = :a and region is null and vsm\_device\_name\_ems is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_paths = :x where area = :a and region = :b and vsm\_device\_name\_ems is null';

sqlStmtEms := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_paths = :x where area = :a and region = :b and vsm\_device\_name\_ems = :c';

for cur in matchedUpNonLiveEbhSpans

loop

if (cur.vsm\_device\_name\_ems is not null) then

dbms\_output.put\_line('In matchedUpNonLiveEbhSpans. '||sqlStmtEms||': '||cur.tot\_up\_ebh\_spans\_w\_nl\_paths||' , '|| cur.area||' , '|| cur.region ||' , '||cur.vsm\_device\_name\_ems);

execute immediate sqlStmtEms using cur.tot\_up\_ebh\_spans\_w\_nl\_paths, cur.area, cur.region, cur.vsm\_device\_name\_ems;

elsif (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans\_w\_nl\_paths, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans\_w\_nl\_paths, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans\_w\_nl\_paths;

end if;

end loop;

commit;

sqlStmtVZW := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_pathscsr = :x where area is null and region is null and vsm\_device\_name\_ems is null';

sqlStmtArea := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_pathscsr = :x where area = :a and region is null and vsm\_device\_name\_ems is null';

sqlStmtRegion := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_pathscsr = :x where area = :a and region = :b and vsm\_device\_name\_ems is null';

sqlStmtEms := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set tot\_up\_ebh\_spans\_w\_nl\_pathscsr = :x where area = :a and region = :b and vsm\_device\_name\_ems = :c';

for cur in matchedUpNonLiveEbhSpansCSR

loop

if (cur.vsm\_device\_name\_ems is not null) then

dbms\_output.put\_line('In matchedUpNonLiveEbhSpansCSR. '||sqlStmtEms||': '||cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr||' , '|| cur.area||' , '|| cur.region ||' , '||cur.vsm\_device\_name\_ems);

execute immediate sqlStmtEms using cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr, cur.area, cur.region, cur.vsm\_device\_name\_ems;

elsif (cur.region is not null) then

execute immediate sqlStmtRegion using cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr, cur.area, cur.region;

elsif (cur.area is not null) then

execute immediate sqlStmtArea using cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr, cur.area;

else

execute immediate sqlStmtVZW using cur.tot\_up\_ebh\_spans\_w\_nl\_pathscsr;

end if;

end loop;

commit;

sqlStmtT1Comp := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set t1\_comp =:x where rowid = :a';

sqlStmtEbhComp := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set ebh\_comp =:x where rowid = :a';

sqlStmtOvlComp := 'update NT\_EVDO\_BTS\_TERM\_EMS\_SUMM\_WRK set overall\_comp =:x where rowid = :a';

for cur in updateCompliance

loop

t1Comp := nvl(cur.tot\_up\_t1\_spans,0) - nvl(cur.tot\_up\_t1\_spans\_w\_nl\_paths,0);

ebhComp := nvl(cur.tot\_up\_ebh\_spans,0) - nvl(cur.tot\_up\_ebh\_spans\_w\_nl\_paths,0);

overallComp :=nvl(cur.tot\_up\_t1\_spans,0) + nvl(cur.tot\_up\_ebh\_spans,0) ;

overallComp := overallComp - nvl(cur.tot\_up\_t1\_spans\_w\_nl\_paths,0) - nvl(cur.tot\_up\_ebh\_spans\_w\_nl\_paths,0);

if(t1Comp >0) then

t1Comp :=(nvl(cur.tot\_up\_t1\_spans\_w\_l\_paths,0)/t1Comp) \*100;

t1Comp := round(t1Comp,2);

execute immediate sqlStmtT1Comp using t1Comp,cur.rd;

--dbms\_output.put\_line('Non null '|| t1Comp);

end if;

if(ebhComp>0) then

ebhComp :=(nvl(cur.tot\_up\_ebh\_spans\_w\_l\_paths,0)/ebhComp) \*100;

ebhComp := round(ebhComp,2);

execute immediate sqlStmtEbhComp using ebhComp,cur.rd;

--dbms\_output.put\_line('good EBH '||ebhComp);

end if;

if(overallComp>0) then

overallComp :=((nvl(cur.tot\_up\_t1\_spans\_w\_l\_paths,0)+nvl(cur.tot\_up\_ebh\_spans\_w\_l\_paths,0))/overallComp) \*100;

overallComp := round(overallComp,2);

execute immediate sqlStmtOvlComp using overallComp,cur.rd;

--dbms\_output.put\_line('good EBH '||overallComp);

end if;

end loop;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(sqlStmt);

dbms\_output.put\_line(SubStr('Error in populate\_device\_summary();: '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

PROCEDURE copy\_nortel\_cdma\_wrk

IS

sqlStmt varchar2(32000);

v\_ready xng\_reports.all\_processes.is\_ready%type;

BEGIN

select is\_ready

into v\_ready

from xng\_reports.all\_processes

where process\_name = 'NORTEL\_CDMA\_AUDIT';

IF v\_ready = 'Y' then

sqlStmt:='DELETE FROM '||NORTEL\_NE\_VS\_XNG\_AUDIT||' WHERE ne\_span\_type = ''VOICE\_SPAN''';

execute immediate sqlStmt;

sqlStmt := 'insert into '||NORTEL\_NE\_VS\_XNG\_AUDIT||' select \* from nortel\_ne\_vs\_xng\_audit\_wrk WHERE ne\_span\_type = ''VOICE\_SPAN''';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating '||NORTEL\_NE\_VS\_XNG\_AUDIT);

sqlStmt := 'truncate table '||NORTEL\_CDMA\_AUDIT\_REG\_SUMM;

execute immediate sqlStmt;

sqlStmt := 'insert into '||NORTEL\_CDMA\_AUDIT\_REG\_SUMM||' select \* from nortel\_cdma\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating '||NORTEL\_CDMA\_AUDIT\_REG\_SUMM);

sqlStmt := 'truncate table '||NORTEL\_CDMA\_AUDIT\_DEV\_SUMM;

execute immediate sqlStmt;

sqlStmt := 'insert into '||NORTEL\_CDMA\_AUDIT\_DEV\_SUMM||' select \* from nortel\_cdma\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating '||NORTEL\_CDMA\_AUDIT\_DEV\_SUMM);

END IF;

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('NORTEL\_CDMA\_AUDIT',4000,SubStr('Error in copy\_nortel\_cdma\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('NORTEL\_CDMA\_AUDIT','STATUS\_FAILURE','N');

END;

PROCEDURE copy\_nortel\_evdo\_wrk

IS

sqlStmt varchar2(32000);

v\_ready xng\_reports.all\_processes.is\_ready%type;

BEGIN

select is\_ready

into v\_ready

from xng\_reports.all\_processes

where process\_name = 'NORTEL\_EVDO\_AUDIT';

IF v\_ready = 'Y' then

sqlStmt:='DELETE FROM '||NORTEL\_NE\_VS\_XNG\_AUDIT||' WHERE ne\_span\_type = ''DATA\_SPAN''';

execute immediate sqlStmt;

sqlStmt := 'insert into '||NORTEL\_NE\_VS\_XNG\_AUDIT||' select \* from nortel\_ne\_vs\_xng\_audit\_wrk WHERE ne\_span\_type = ''DATA\_SPAN''';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating '||NORTEL\_NE\_VS\_XNG\_AUDIT);

sqlStmt := 'truncate table '||NT\_EVDO\_BTS\_TERM\_REG\_SUMM;

execute immediate sqlStmt;

sqlStmt := 'insert into '||NT\_EVDO\_BTS\_TERM\_REG\_SUMM||' select \* from nt\_evdo\_bts\_term\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating '||NT\_EVDO\_BTS\_TERM\_REG\_SUMM);

sqlStmt := 'truncate table '||NT\_EVDO\_BTS\_TERM\_EMS\_SUMM;

execute immediate sqlStmt;

sqlStmt := 'insert into '||NT\_EVDO\_BTS\_TERM\_EMS\_SUMM||' select \* from nt\_evdo\_bts\_term\_ems\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating '||NT\_EVDO\_BTS\_TERM\_EMS\_SUMM);

END IF;

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('NORTEL\_EVDO\_AUDIT',4000,SubStr('Error in copy\_nortel\_evdo\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('NORTEL\_EVDO\_AUDIT','STATUS\_FAILURE','N');

END;

PROCEDURE run\_cdma\_audit\_wrk

IS

BEGIN

BEGIN

nt\_cdma\_audit();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in nt\_cdma\_audit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

populate\_cdma\_regional\_summary();

populate\_cdma\_device\_summary();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in populate\_cdma\_summary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

END;

PROCEDURE run\_evdo\_audit\_wrk

IS

BEGIN

BEGIN

nt\_evdo\_audit();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in nt\_evdo\_audit(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

BEGIN

populate\_evdo\_regional\_summary();

populate\_evdo\_device\_summary();

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error in nortel\_evdo\_summary(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

END;

PROCEDURE truncate\_nortel\_cdma\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from nortel\_ne\_vs\_xng\_audit\_wrk WHERE ne\_span\_type = ''VOICE\_SPAN''';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting nortel\_ne\_vs\_xng\_audit\_wrk VOICE\_SPANs');

sqlStmt := 'delete from nortel\_cdma\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting nortel\_cdma\_audit\_reg\_summ\_wrk');

sqlStmt := 'delete from nortel\_cdma\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting nortel\_cdma\_audit\_dev\_summ\_wrk');

END;

PROCEDURE truncate\_nortel\_evdo\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt := 'delete from nortel\_ne\_vs\_xng\_audit\_wrk WHERE ne\_span\_type = ''DATA\_SPAN''';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting nortel\_ne\_vs\_xng\_audit\_wrk DATA\_SPANs');

sqlStmt := 'delete from nt\_evdo\_bts\_term\_reg\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting nt\_evdo\_bts\_term\_ems\_summ\_wrk');

sqlStmt := 'delete from nt\_evdo\_bts\_term\_ems\_summ\_wrk';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done deleting nt\_evdo\_bts\_term\_ems\_summ\_wrk');

END;

PROCEDURE restore\_nortel\_cdma\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt:='DELETE FROM nortel\_ne\_vs\_xng\_audit\_wrk WHERE ne\_span\_type = ''VOICE\_SPAN''';

execute immediate sqlStmt;

sqlStmt := 'insert into nortel\_ne\_vs\_xng\_audit\_wrk select \* from '||NORTEL\_NE\_VS\_XNG\_AUDIT||' WHERE ne\_span\_type = ''VOICE\_SPAN''';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating nortel\_ne\_vs\_xng\_audit\_wrk');

sqlStmt := 'truncate table nortel\_cdma\_audit\_reg\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into nortel\_cdma\_audit\_reg\_summ\_wrk select \* from '||NORTEL\_CDMA\_AUDIT\_REG\_SUMM||'';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating nortel\_cdma\_audit\_reg\_summ\_wrk');

sqlStmt := 'truncate table nortel\_cdma\_audit\_dev\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into nortel\_cdma\_audit\_dev\_summ\_wrk select \* from '||NORTEL\_CDMA\_AUDIT\_DEV\_SUMM||'';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating nortel\_cdma\_audit\_dev\_summ\_wrk');

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('NORTEL\_CDMA\_AUDIT',4000,SubStr('Error in restore\_nortel\_cdma\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('NORTEL\_CDMA\_AUDIT','STATUS\_FAILURE','N');

END;

PROCEDURE restore\_nortel\_evdo\_wrk IS

sqlStmt varchar2(32000);

BEGIN

sqlStmt:='DELETE FROM nortel\_ne\_vs\_xng\_audit\_wrk WHERE ne\_span\_type = ''DATA\_SPAN''';

execute immediate sqlStmt;

sqlStmt := 'insert into nortel\_ne\_vs\_xng\_audit\_wrk select \* from '||NORTEL\_NE\_VS\_XNG\_AUDIT||' WHERE ne\_span\_type = ''DATA\_SPAN''';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating nortel\_ne\_vs\_xng\_audit\_wrk');

sqlStmt := 'truncate table nt\_evdo\_bts\_term\_reg\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into nt\_evdo\_bts\_term\_reg\_summ\_wrk select \* from '||NT\_EVDO\_BTS\_TERM\_REG\_SUMM||'';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating nt\_evdo\_bts\_term\_reg\_summ\_wrk');

sqlStmt := 'truncate table nt\_evdo\_bts\_term\_ems\_summ\_wrk';

execute immediate sqlStmt;

sqlStmt := 'insert into nt\_evdo\_bts\_term\_ems\_summ\_wrk select \* from '||NT\_EVDO\_BTS\_TERM\_EMS\_SUMM||'';

execute immediate sqlStmt;

commit;

dbms\_output.put\_line('Done populating nt\_evdo\_bts\_term\_reg\_summ\_wrk');

EXCEPTION

WHEN OTHERS THEN

WATCHDOG.logerror('NORTEL\_EVDO\_AUDIT',4000,SubStr('Error in restore\_nortel\_evdo\_wrk; '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

WATCHDOG.updateprocessend('NORTEL\_EVDO\_AUDIT','STATUS\_FAILURE','N');

END;

END;

/

--------------------------------------------------------

-- DDL for Package Body RPT\_IAN\_NOCC\_SITE\_TEST

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."RPT\_IAN\_NOCC\_SITE\_TEST"

IS

--

-- Purpose: NOCC SITE ALARAMS AGGREAGATE SUMMARY and DETAIL reports.

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- ---------------------- ------------ -----------------------------------------

-- Krishna Gadde Oct 2005 INitial release.

-- Krihsna Gadde March 2006 Tunned...

-- Krishna Gadde Aug 2006 fixed query. Issue with val\_attr\_group

--

PROCEDURE getAllDomains

IS

TYPE l\_cursor\_Type IS REF CURSOR;

cursor\_Domains l\_cursor\_Type;

l\_Area xng\_reports.VZW\_Network\_Org.Area%TYPE;

l\_Region xng\_reports.VZW\_Network\_Org.Region%TYPE;

l\_DomainId vzwnet.Domain\_inst.domain\_inst\_id%TYPE;

l\_DomainName vzwnet.Domain\_inst.domain\_name%TYPE;

sql\_Str VARCHAR2 (555);

BEGIN

sql\_Str := 'SELECT vno.Area, vno.Region, di.Domain\_inst\_id, di.domain\_name

FROM xng\_reports.VZW\_NetWork\_Org vno, vzwnet.Domain\_Inst di

WHERE di.domain\_name = vno.region\_domain\_name

and di.domain\_name = ''NNO\_DOMAIN''

GROUP BY vno.Area, vno.Region, di.Domain\_inst\_id, di.domain\_name

ORDER BY vno.Area, vno.Region';

OPEN cursor\_Domains FOR sql\_Str;

LOOP

FETCH cursor\_Domains INTO l\_Area, l\_Region, l\_DomainId, l\_DomainName;

EXIT WHEN cursor\_Domains%NOTFOUND;

DBMS\_OUTPUT.put\_line('Loading Domain...: ' || l\_DomainName );

-- Remove the old Detail data by domain's...

removeDetails( l\_DomainId );

-- get all Sites for a passed domain...

getAllSites( l\_DomainId, l\_DomainName, l\_Area, l\_Region );

COMMIT;

END LOOP;

CLOSE cursor\_Domains;

-- Not enough rollback segment...Moved up into the loop COMMIT;

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

IF( cursor\_Domains%ISOPEN ) THEN

CLOSE cursor\_Domains;

END IF;

DBMS\_OUTPUT.put\_line( ' Area : ' || l\_Area ||

' Region : ' || l\_Region ||

' DomainId : ' || l\_DomainId ||

' DomainName:' || l\_DomainName ||

' SQLCODE: ' || TO\_CHAR(SQLCODE));

RAISE;

END;

PROCEDURE getAllSites(

p\_DomainId IN vzwnet.Domain\_Inst.domain\_inst\_id%TYPE,

p\_DomainName IN vzwnet.Domain\_Inst.domain\_name%TYPE,

p\_Area IN xng\_reports.VZW\_Network\_Org.Area%TYPE,

p\_Region IN xng\_reports.VZW\_Network\_Org.Region%TYPE)

IS

-- For NOC\_Aggs Summary report....

l\_EquipInstId vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE;

l\_TowerStatus vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE;

l\_AntennaVZWOwned VARCHAR2(255);

l\_HasAntennaTower VARCHAR2(30);

l\_LightingRequired VARCHAR2(30);

l\_SiteInstId NOCC\_LAST\_TEST\_DETAILS.site\_inst\_id%TYPE;

l\_SiteName NOCC\_LAST\_TEST\_DETAILS.site\_name%TYPE;

l\_SiteId NOCC\_LAST\_TEST\_DETAILS.site\_id%TYPE; -- Site\_Inst.Base\_Num

l\_SiteTechName NOCC\_LAST\_TEST\_DETAILS.site\_tech\_name%TYPE;

l\_inventoried\_cells NOCC\_LAST\_TEST\_DETAILS.inventoried\_cells%TYPE;

l\_SiteType NOCC\_LAST\_TEST\_DETAILS.site\_type%TYPE; -- Site\_Inst.Num

l\_NOCC\_Monitored NOCC\_LAST\_TEST\_DETAILS.nocc\_monitored%TYPE;

l\_LightLastTestedDate NOCC\_LAST\_TEST\_DETAILS.light\_last\_tested%TYPE;

l\_LightNextTestDate NOCC\_LAST\_TEST\_DETAILS.light\_next\_test%TYPE;

l\_LightTestOverDue NOCC\_LAST\_TEST\_DETAILS.light\_test\_overdue%TYPE;

l\_EnvLastTestedDate NOCC\_LAST\_TEST\_DETAILS.env\_last\_tested%TYPE;

l\_EnvNextTestDate NOCC\_LAST\_TEST\_DETAILS.env\_next\_test%TYPE;

l\_EnvTestOverDue NOCC\_LAST\_TEST\_DETAILS.env\_test\_overdue%TYPE;

-- For NOC\_Aggs Summary report....

l\_Site\_Count NOCC\_LAST\_TEST\_SUMMARY.sites%TYPE := 0;

l\_HasAntennaTower\_Count NOCC\_LAST\_TEST\_SUMMARY.antennas%TYPE := 0;

l\_AntennaVZWOwned\_Count NOCC\_LAST\_TEST\_SUMMARY.vzw\_owned%TYPE := 0;

l\_LightingRequired\_Count NOCC\_LAST\_TEST\_SUMMARY.lights\_reqd%TYPE := 0;

l\_LightTestedDate\_Count NOCC\_LAST\_TEST\_SUMMARY.lights\_tested%TYPE := 0;

l\_LightTestOverDue\_Count NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_overdue%TYPE := 0;

l\_LightTest\_OK\_Count NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_ok%TYPE := 0;

l\_NOCC\_Monitored\_Count NOCC\_LAST\_TEST\_SUMMARY.NOCC\_monitored\_sites%TYPE := 0;

l\_EnvLastTestedDate\_Count NOCC\_LAST\_TEST\_SUMMARY.env\_tested%TYPE := 0;

l\_EnvTestOverDue\_Count NOCC\_LAST\_TEST\_SUMMARY.env\_test\_overdue%TYPE := 0;

l\_EnvTestOK\_Count NOCC\_LAST\_TEST\_SUMMARY.env\_test\_ok%TYPE := 0;

l\_saved\_at\_equpment\_level number := 1;

--## ADD Include List for SITE\_TYPES

CURSOR csr\_getAllSites( pp\_DomainId IN vzwnet.Domain\_Inst.domain\_inst\_id%Type )

IS

SELECT distinct si.Site\_Inst\_Id, si.Site\_hum\_Id, si.Base\_Num, si.Num

FROM vzwnet.Site\_Domain\_map sdm,

vzwnet.Site\_Inst si

WHERE sdm.Domain\_Inst\_Id = pp\_DomainId

AND si.Num in ('MICROWAVE')

AND si.Site\_Inst\_Id = sdm.Site\_Inst\_Id;

--and si.site\_inst\_id not in (select site\_inst\_id from vzwnet.site\_domain\_map where domain\_inst\_id = 1191); --to keep out ALLTEL sites

CURSOR csr\_getSiteEquipment( pp\_SiteInstId IN vzwnet.Site\_Inst.site\_inst\_id%Type )

IS

SELECT ei.Equip\_Inst\_Id

FROM vzwnet.Equip\_Inst ei

WHERE ei.Site\_Inst\_Id (+) = pp\_SiteInstId

AND UPPER(ei.type (+)) = 'ANTENNA SUPPORT STRUCTURE';

BEGIN

OPEN csr\_getAllSites( p\_DomainId );

LOOP

FETCH csr\_getAllSites INTO l\_SiteInstId, l\_SiteName, l\_SiteId, l\_SiteType;

EXIT WHEN csr\_getAllSites%NOTFOUND;

l\_NOCC\_Monitored := NULL;

l\_SiteTechName := NULL;

l\_inventoried\_cells := NULL;

l\_TowerStatus := NULL;

l\_EnvLastTestedDate := NULL;

l\_EnvNextTestDate := NULL;

l\_EnvTestOverDue := NULL;

l\_saved\_at\_equpment\_level := 1;

l\_Site\_Count := l\_Site\_Count + 1;

l\_SiteTechName := getSiteTechName( l\_SiteInstId );

l\_inventoried\_cells := getCellEquipDesc( l\_SiteInstId );

-- Environmental details...

l\_NOCC\_Monitored := getIsNOCCMonitored( l\_SiteInstId );

IF( upper(l\_NOCC\_Monitored) <> 'NO' ) THEN

l\_NOCC\_Monitored\_Count := l\_NOCC\_Monitored\_Count + 1;

l\_EnvLastTestedDate := varchar2date( getEnvLastTestedDate( l\_SiteInstId, l\_SiteType ));

IF( l\_EnvLastTestedDate IS NOT NULL ) THEN

l\_EnvLastTestedDate\_Count := l\_EnvLastTestedDate\_Count + 1;

-- Changed according to Jim 2/6/2006, --IF( (upper(l\_SiteType) = 'MTSO' ) OR (upper(l\_SiteType) = 'HUB' ) ) THEN

IF ( upper(l\_SiteType) = 'MTSO' ) THEN

l\_EnvNextTestDate := TO\_CHAR( ADD\_MONTHS( TO\_DATE( l\_EnvLastTestedDate, 'DD\_MON\_YY'), 6 ));

ELSIF ( upper(l\_SiteType) = 'CELL' ) THEN

-- Conditional interval based on Failed First Environmental Attempt UDA

l\_EnvNextTestDate := TO\_CHAR( ADD\_MONTHS( TO\_DATE( l\_EnvLastTestedDate, 'DD\_MON\_YY'),

12\* getFailedFirstEnvAttempt( l\_SiteInstId )));

ELSE

l\_EnvNextTestDate := TO\_CHAR( ADD\_MONTHS( TO\_DATE( l\_EnvLastTestedDate, 'DD\_MON\_YY'), 12 ));

END IF;

l\_EnvTestOverDue := TO\_CHAR( TO\_DATE( l\_EnvNextTestDate, 'DD\_MON\_YY') - TRUNC(SYSDATE ));

IF( TO\_NUMBER(l\_EnvTestOverDue ) <= 0 ) THEN

DBMS\_OUTPUT.put\_line('OVERDUE 1 '|| l\_SiteInstId || to\_char(l\_EnvTestOverDue\_Count));

l\_EnvTestOverDue\_Count := l\_EnvTestOverDue\_Count + 1;

ELSE

l\_EnvTestOK\_Count := l\_EnvTestOK\_Count + 1;

END IF;

ELSE

DBMS\_OUTPUT.put\_line('OVERDUE 2'|| l\_SiteInstId || to\_char(l\_EnvTestOverDue\_Count));

l\_EnvTestOverDue\_Count := l\_EnvTestOverDue\_Count + 1;

-- added 04/21/2006 JLA

l\_EnvNextTestDate := 'ASAP';

l\_EnvTestOverDue := -90;

--

END IF; -- l\_ENVLastTestedDate

END IF; -- l\_NOCC\_Monitored

l\_HasAntennaTower := NULL;

l\_AntennaVZWOwned := NULL;

l\_LightingRequired := NULL;

l\_LightLastTestedDate := NULL;

l\_LightNextTestDate := NULL;

l\_LightTestOverDue := NULL;

-- Antenna details...

OPEN csr\_getSiteEquipment( l\_SiteInstId );

LOOP

FETCH csr\_getSiteEquipment INTO l\_EquipInstId;

EXIT WHEN csr\_getSiteEquipment%NOTFOUND;

l\_HasAntennaTower := NULL;

l\_AntennaVZWOwned := NULL;

l\_LightingRequired := NULL;

l\_LightLastTestedDate := NULL;

l\_LightNextTestDate := NULL;

l\_LightTestOverDue := NULL;

l\_HasAntennaTower := getHasAntennaTower( l\_SiteInstId, l\_EquipInstId );

IF( l\_HasAntennaTower = 'Has Tower' ) THEN

l\_HasAntennaTower\_Count := l\_HasAntennaTower\_Count + 1;

--l\_HasAntennaTower := 'Has Tower';

l\_AntennaVZWOwned := getIsAntennaVZWOwned( l\_SiteInstId, l\_EquipInstId );

-- IF( upper(l\_AntennaVZWOwned) <> 'OTHER' ) THEN

IF( upper(l\_AntennaVZWOwned) = 'VERIZON WIRELESS' ) THEN

-- AntennaStructure is 'VERIZON WIRELESS' or empty, assume its VZW's.

l\_AntennaVZWOwned\_Count := l\_AntennaVZWOwned\_Count + 1;

l\_LightingRequired := getIsLightingRequired( l\_SiteInstId, l\_EquipInstId );

IF( upper(l\_LightingRequired) <> 'NO' ) THEN

-- If it is clearly specified 'NO' then its no, otherwise its required!.

l\_LightingRequired\_Count := l\_LightingRequired\_Count + 1;

l\_LightLastTestedDate := varchar2date( getTowerLightLastTestedDate( l\_SiteInstId , l\_SiteType ));

IF( l\_LightLastTestedDate IS NOT NULL ) THEN

l\_LightTestedDate\_Count := l\_LightTestedDate\_Count + 1;

l\_LightNextTestDate := TO\_CHAR( TO\_DATE( l\_LightLastTestedDate, 'DD-MON-YY') + 90 );

l\_LightTestOverDue := TO\_CHAR( TO\_DATE( l\_LightNextTestDate, 'DD-MON-YY') - TRUNC(SYSDATE) );

IF ( TO\_NUMBER( l\_LightTestOverDue ) <= 0 ) THEN

l\_LightTestOverDue\_Count := l\_LightTestOverDue\_Count + 1;

l\_LightNextTestDate := 'ASAP';

ELSE

l\_LightTest\_OK\_Count := l\_LightTest\_OK\_Count + 1;

END IF; -- l\_LightTestOverDue

ELSE

l\_LightTestOverDue\_Count := l\_LightTestOverDue\_Count + 1;

--added 04/21/2006 JLA

l\_LightNextTestDate := 'ASAP';

l\_LightTestOverDue := '-90';

--

END IF; --l\_LightLastTestedDate

ELSE

l\_LightLastTestedDate := 'Light Not Reqd';

l\_LightNextTestDate := 'N/A';

END IF; -- l\_LightingRequired

ELSE

l\_AntennaVZWOwned := 'Not VZW Owned';

END IF; -- l\_AntennaVZWOwned

ELSE

l\_LightLastTestedDate := 'No Tower';

l\_LightNextTestDate := 'N/A';

END IF; -- l\_HasAntennaTower

-- Save detail record...

Insert\_NOCC\_Last\_Test\_Details( p\_DomainId,

p\_DomainName,

l\_SiteInstId,

l\_SiteName,

l\_SiteId,

l\_SiteType,

l\_SiteTechName,

l\_inventoried\_cells,

l\_NOCC\_Monitored,

l\_AntennaVZWOwned,

l\_LightingRequired,

l\_LightLastTestedDate,

l\_LightNextTestDate,

l\_LightTestOverDue,

l\_EnvLastTestedDate,

l\_EnvNextTestDate,

l\_EnvTestOverDue,

l\_EquipInstId );

l\_saved\_at\_equpment\_level := 2;

END LOOP;

CLOSE csr\_getSiteEquipment;

if ( l\_saved\_at\_equpment\_level = 1 ) then

-- Save detail record...

Insert\_NOCC\_Last\_Test\_Details( p\_DomainId,

p\_DomainName,

l\_SiteInstId,

l\_SiteName,

l\_SiteId,

l\_SiteType,

l\_SiteTechName,

l\_inventoried\_cells,

l\_NOCC\_Monitored,

l\_AntennaVZWOwned,

l\_LightingRequired,

l\_LightLastTestedDate,

l\_LightNextTestDate,

l\_LightTestOverDue,

l\_EnvLastTestedDate,

l\_EnvNextTestDate,

l\_EnvTestOverDue,

l\_EquipInstId );

end if;

END LOOP;

-- Save domain level test summary/count record...

insert\_NOCC\_LAST\_TEST\_SUMMARY( p\_Area,

p\_Region,

p\_DomainId,

p\_DomainName,

l\_Site\_Count,

l\_HasAntennaTower\_Count,

l\_AntennaVZWOwned\_Count,

l\_LightingRequired\_Count,

l\_LightTestedDate\_Count,

l\_LightTestOverDue\_Count,

l\_LightTest\_OK\_Count,

l\_NOCC\_Monitored\_Count,

l\_EnvLastTestedDate\_Count,

l\_EnvTestOverDue\_Count,

l\_EnvTestOK\_Count );

CLOSE csr\_getAllSites;

EXCEPTION

WHEN OTHERS THEN

IF( csr\_getAllSites%ISOPEN ) THEN

CLOSE csr\_getAllSites;

END IF;

IF( csr\_getSiteEquipment%ISOPEN ) THEN

CLOSE csr\_getSiteEquipment;

END IF;

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in GetAllSites(DomainId='

|| p\_DomainId || ' ) -> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

/\*

FUNCTION getSiteTechName( p\_SiteInstId IN vzwnet.Site\_attr\_settings.Site\_Inst\_Id%TYPE )

RETURN vzwnet.Site\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Site\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT sas.attr\_value

INTO retval

FROM vzwnet.Site\_attr\_settings sas,

vzwnet.val\_attr\_Name van

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND sas.val\_attr\_Inst\_Id = van.val\_attr\_Inst\_Id

AND upper(van.attr\_Name) = 'SITE TECHNICIAN'

AND upper(van.group\_Name) = 'EMERGENCY CONTACTS';

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN null;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getSiteTechName('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

\*/

FUNCTION getSiteTechName( p\_SiteInstId IN vzwnet.Site\_attr\_settings.Site\_Inst\_Id%TYPE )

RETURN vzwnet.Site\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Site\_attr\_settings.attr\_value%TYPE;

BEGIN

select

min(ri.name) name into retval

from

vzwnet.resource\_inst ri

join vzwnet.resource\_definition\_inst di

on di.DEFINITION\_INST\_ID = ri.DEFINITION\_INST\_ID

join vzwnet.resource\_associations ra

on ra.resource\_inst\_id = ri.resource\_inst\_id

where

ra.target\_type\_id = 16

and di.name = 'Operations'

and ri.category = 'Technician'

and ra.TARGET\_INST\_ID=p\_SiteInstId

group by ra.target\_inst\_id

;

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN null;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getSiteTechName('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getCellEquipDesc( p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE )

RETURN NOCC\_LAST\_TEST\_DETAILS.INVENTORIED\_CELLS%TYPE

IS

l\_Cell\_Equipment VARCHAR2(3099 );

l\_Equip\_descr VARCHAR2(255 );

CURSOR cursor\_Equipment\_desc( pp\_SiteInstId IN VARCHAR2 )

IS

SELECT NVL( ei.DESCR, 'none' )

FROM vzwnet.Equip\_Inst ei

WHERE ei.Site\_Inst\_Id = pp\_SiteInstId

AND ei.TYPE IN( SELECT xp\_Equip\_Type

FROM xng\_apps.VZ\_Ntwk\_Node\_Map

WHERE end\_point\_Id = 'CELL DEMARK' );

BEGIN

OPEN cursor\_Equipment\_desc( p\_SiteInstId );

LOOP

FETCH cursor\_Equipment\_desc INTO l\_Equip\_descr;

EXIT WHEN cursor\_Equipment\_desc%NOTFOUND;

IF ( l\_Cell\_Equipment IS NULL ) THEN

l\_Cell\_Equipment := l\_Equip\_descr;

ELSE

l\_Cell\_Equipment := l\_Cell\_Equipment || ';' || l\_Equip\_descr;

END IF;

END LOOP;

CLOSE cursor\_Equipment\_desc;

Return l\_Cell\_Equipment;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in loadEquipDesc('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

IF( cursor\_Equipment\_desc%ISOPEN ) THEN

CLOSE cursor\_Equipment\_desc;

END IF;

END;

FUNCTION getIsNOCCMonitored( p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE )

RETURN VARCHAR2

IS

retval VARCHAR2(25);

BEGIN

SELECT NVL(sas.Attr\_value, 'NO' )

INTO retval

FROM vzwnet.Site\_Attr\_settings sas,

vzwnet.val\_Attr\_Name van

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND van.val\_Attr\_Inst\_Id = sas.val\_Attr\_Inst\_Id

AND UPPER(van.Attr\_Name ) = 'ENVIRONMENTALS NOCC MONITORED'

AND UPPER(van.group\_Name ) = 'SITE ALARM VERIFICATION TEST';

-- added 04/21/2006 JLA

IF upper(retval) != 'NO' THEN

retval := 'YES';

ELSE

retval := 'NO';

END IF;

--

RETURN trim( retval );

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'No';

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line( SUBSTR( 'Error, in getIsNoccMonitored('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) ||': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getFailedFirstEnvAttempt( p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE )

RETURN VARCHAR2

IS

retval NUMBER;

BEGIN

SELECT decode(upper(sas.Attr\_value),null,1,

'NO',2,

'YES',1,1)

INTO retval

FROM vzwnet.Site\_Attr\_settings sas,

vzwnet.val\_Attr\_Name van

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND van.val\_Attr\_Inst\_Id = sas.val\_Attr\_Inst\_Id

AND UPPER(van.Attr\_Name ) = 'FAILED FIRST ENV ATTEMPT'

AND UPPER(van.group\_Name ) = 'SITE ALARM VERIFICATION TEST';

-- retval is a multiplier for 1 or 2 year Env Testing interval for CELL Sites

-- IF upper(retval) = 'NO' and retval is not null THEN

-- retval := 2;

-- ELSE

-- retval := 1;

-- END IF;

RETURN ( retval );

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 1;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line( SUBSTR( 'Error, in getFailedFirstEnvAttempt('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) ||': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getEnvLastTestedDate(

p\_SiteInstId IN vzwnet.Site\_attr\_settings.Site\_Inst\_Id%TYPE,

p\_SiteType IN vzwnet.val\_attr\_group.class\_category%TYPE )

RETURN vzwnet.Site\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Site\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT sas.attr\_value

INTO retval

FROM vzwnet.Site\_attr\_settings sas,

vzwnet.val\_attr\_Name van,

vzwnet.val\_attr\_group vag

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND van.val\_attr\_Inst\_Id = sas.val\_attr\_Inst\_Id

AND UPPER(van.attr\_Name ) = 'ENVIRONMENT DATE LAST TESTED'

AND UPPER(vag.group\_Name ) = UPPER(van.group\_Name )

AND vag.class\_category = p\_SiteType

AND vag.appl\_class = 'I'

AND UPPER(vag.group\_Name ) = 'SITE ALARM VERIFICATION TEST';

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN null;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getENVLastTestedDate('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getHasAntennaTower(

p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE )

RETURN VARCHAR2

IS

retval VARCHAR2(20);

BEGIN

SELECT NVL2(ei.Equip\_Inst\_Id, 'Has Tower', 'No Tower' ) -- some sites have multiple antennas

INTO retval

FROM vzwnet.Equip\_Inst ei,

vzwnet.Site\_Inst si

WHERE ei.Equip\_Inst\_Id = p\_EquipInstId

AND UPPER(ei.TYPE) = 'ANTENNA SUPPORT STRUCTURE'

AND si.Site\_inst\_id = ei.site\_inst\_id

AND si.Site\_Inst\_Id = p\_SiteInstId;

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'No Tower';

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getHasAntennaTower('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getIsAntennaVZWOwned(

p\_SiteInstId IN vzwnet.Equip\_Inst.Site\_Inst\_Id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.equip\_inst\_id%TYPE )

RETURN vzwnet.Equip\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Equip\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT UPPER( eas.attr\_value )

INTO retval

FROM vzwnet.Equip\_Inst ei,

vzwnet.Equip\_attr\_settings eas,

vzwnet.val\_attr\_Name van

WHERE ei.Site\_Inst\_Id = p\_SiteInstId

AND ei.equip\_inst\_id = p\_EquipInstId

AND ei.type = 'ANTENNA SUPPORT STRUCTURE'

AND eas.Equip\_Inst\_Id = ei.Equip\_Inst\_Id

AND van.val\_attr\_Inst\_Id = eas.val\_attr\_Inst\_Id

AND UPPER(van.attr\_Name ) = 'OWNER'

AND UPPER(van.group\_Name ) = 'ANTENNA SUPPORT STRUCTURE';

-- added 04/21/2006 JLA

IF upper(retval) = 'OTHER' THEN

retval := 'OTHER';

ELSE

retval := 'VERIZON WIRELESS';

END IF;

--

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'VERIZON WIRELESS';

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line( SUBSTR( 'Error, in getIsAntennaVZWOwned('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getIsLightingRequired(

p\_SiteInstId IN vzwnet.Equip\_Inst.Site\_Inst\_Id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.equip\_inst\_id%TYPE )

RETURN vzwnet.Equip\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Equip\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT UPPER(eas.attr\_value)

INTO retval

FROM vzwnet.Equip\_Inst ei,

vzwnet.Equip\_attr\_settings eas,

vzwnet.val\_attr\_Name van

-- Krishna 8/4/2006 ,vzwnet.val\_attr\_group vag

WHERE ei.Site\_Inst\_Id = p\_SiteInstId

AND ei.equip\_inst\_id = p\_EquipInstId

AND eas.Equip\_Inst\_Id = ei.Equip\_Inst\_Id

AND van.val\_attr\_Inst\_Id = eas.val\_attr\_Inst\_Id

AND UPPER(van.attr\_Name ) = 'LIGHTING REQUIRED'

--Krishna 8/4/2006 AND UPPER(vag.group\_Name ) = UPPER(van.group\_Name )

AND UPPER(van.group\_Name ) = 'FAA INFORMATION';

-- added 04/21/2006 JLA

IF upper(retval) = 'NO' THEN

retval := 'NO';

ELSE

retval := 'YES';

END IF;

--

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'YES'; -- if its empty then we are treating it as Yes.

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getIsLightingRequired('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getTowerLightLastTestedDate(

p\_SiteInstId IN vzwnet.Site\_attr\_settings.Site\_Inst\_Id%TYPE,

p\_SiteType IN vzwnet.val\_attr\_group.class\_category%TYPE )

RETURN vzwnet.Site\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Site\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT sas.attr\_value

INTO retval

FROM vzwnet.Site\_attr\_settings sas,

vzwnet.val\_attr\_Name van,

vzwnet.val\_attr\_group vag

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND van.val\_attr\_Inst\_Id = sas.val\_attr\_Inst\_Id

AND UPPER(van.attr\_Name ) = 'TOWER LIGHT DATE LAST TESTED'

AND UPPER(vag.group\_Name ) = UPPER(van.group\_Name )

AND UPPER(vag.class\_category ) = p\_SiteType

AND UPPER(vag.group\_Name ) = 'SITE ALARM VERIFICATION TEST';

RETURN trim(retval);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN null;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line( SUBSTR( 'Error, in getTowerLightLastTestedDate('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

PROCEDURE insert\_NOCC\_Last\_Test\_Details(

p\_DomainId IN NOCC\_LAST\_TEST\_DETAILS.domain\_Id%TYPE,

p\_DomainName IN NOCC\_LAST\_TEST\_DETAILS.domain\_name%TYPE,

p\_SiteInstId IN NOCC\_LAST\_TEST\_DETAILS.site\_Inst\_Id%TYPE,

p\_SiteName IN NOCC\_LAST\_TEST\_DETAILS.site\_name%TYPE,

p\_SiteId IN NOCC\_LAST\_TEST\_DETAILS.site\_id%TYPE,

p\_Sitetype IN NOCC\_LAST\_TEST\_DETAILS.site\_type%TYPE,

p\_SiteTechName IN NOCC\_LAST\_TEST\_DETAILS.site\_tech\_name%TYPE,

p\_inventoried\_cells IN NOCC\_LAST\_TEST\_DETAILS.inventoried\_cells%TYPE,

p\_NOCC\_Monitored IN NOCC\_LAST\_TEST\_DETAILS.nocc\_Monitored%TYPE,

p\_vzw\_owned IN NOCC\_LAST\_TEST\_DETAILS.vzw\_owned%TYPE,

p\_lights\_reqd IN NOCC\_LAST\_TEST\_DETAILS.lights\_reqd%TYPE,

p\_LightLastTested IN NOCC\_LAST\_TEST\_DETAILS.Light\_last\_tested%TYPE,

p\_LightNextTest IN NOCC\_LAST\_TEST\_DETAILS.Light\_next\_test%TYPE,

p\_LightTestOverDue IN NOCC\_LAST\_TEST\_DETAILS.light\_test\_overdue%TYPE,

p\_ENVLastTested IN NOCC\_LAST\_TEST\_DETAILS.env\_last\_tested%TYPE,

p\_ENVNextTest IN NOCC\_LAST\_TEST\_DETAILS.env\_next\_test%TYPE,

p\_ENVTestOverDue IN NOCC\_LAST\_TEST\_DETAILS.env\_test\_overdue%TYPE,

p\_EquipInstid IN NOCC\_LAST\_TEST\_DETAILS.equip\_inst\_id%TYPE )

IS

BEGIN

INSERT INTO NOCC\_LAST\_TEST\_DETAILS ( Domain\_Id,

Domain\_Name,

Site\_Inst\_Id,

Site\_Name,

Site\_ID,

Site\_Type,

Site\_Tech\_Name,

inventoried\_cells,

NOCC\_Monitored,

vzw\_owned,

lights\_reqd,

Light\_Last\_Tested,

Light\_Next\_Test,

Light\_Test\_OverDue,

ENV\_Last\_Tested,

ENV\_Next\_Test,

ENV\_Test\_OverDue,

Run\_Date,

Equip\_inst\_id )

VALUES(

p\_DomainId,

p\_DomainName,

p\_SiteInstId,

p\_SiteName,

p\_SiteID,

p\_SiteType,

p\_SiteTechName,

p\_inventoried\_cells,

p\_NOCC\_Monitored,

p\_vzw\_owned,

p\_lights\_reqd,

p\_LightLastTested,

p\_LightNextTest,

p\_LightTestOverDue,

p\_ENVLastTested,

p\_ENVNextTest,

p\_ENVTestOverDue,

SYSDATE,

p\_EquipInstid );

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line('Error Insert\_NOCC\_LAST\_TEST\_DETAILS()-> '

||' - '|| p\_DomainId

||' - '|| p\_DomainName

||' - '|| p\_SiteInstId

||' - '|| p\_SiteName

||' - '|| TO\_CHAR(SQLCODE));

END;

PROCEDURE insert\_NOCC\_LAST\_TEST\_SUMMARY(

p\_Area IN NOCC\_LAST\_TEST\_SUMMARY.Area%TYPE,

p\_Region IN NOCC\_LAST\_TEST\_SUMMARY.Region%TYPE,

p\_DomainId IN NOCC\_LAST\_TEST\_SUMMARY.domain\_id%TYPE,

p\_DomainName IN NOCC\_LAST\_TEST\_SUMMARY.domain\_name%TYPE,

p\_Sites IN NOCC\_LAST\_TEST\_SUMMARY.sites%TYPE,

p\_Antennas IN NOCC\_LAST\_TEST\_SUMMARY.antennas%TYPE,

p\_VZWOwned IN NOCC\_LAST\_TEST\_SUMMARY.vzw\_owned%TYPE,

p\_LightsReqd IN NOCC\_LAST\_TEST\_SUMMARY.lights\_reqd%TYPE,

p\_LightsTested IN NOCC\_LAST\_TEST\_SUMMARY.lights\_tested%TYPE,

p\_LightsOverDue IN NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_overdue%TYPE,

p\_LightsOk IN NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_ok%TYPE,

p\_Monitored IN NOCC\_LAST\_TEST\_SUMMARY.NOCC\_monitored\_sites%TYPE,

p\_EnvTested IN NOCC\_LAST\_TEST\_SUMMARY.env\_tested%TYPE,

p\_EnvOverDue IN NOCC\_LAST\_TEST\_SUMMARY.env\_test\_overdue%TYPE,

p\_EnvOK IN NOCC\_LAST\_TEST\_SUMMARY.env\_test\_ok%TYPE

)

IS

BEGIN

INSERT INTO NOCC\_LAST\_TEST\_SUMMARY( Area,

Region,

Domain\_Id,

Domain\_Name,

Sites,

Antennas,

VZW\_Owned,

Lights\_REQD,

Lights\_Tested,

Lights\_Test\_OverDue,

Lights\_Test\_Ok,

NOCC\_Monitored\_Sites,

ENV\_Tested,

ENV\_Test\_OverDue,

ENV\_Test\_Ok,

Run\_Date )

VALUES( p\_Area,

p\_Region,

p\_DomainId,

p\_DomainName,

p\_Sites,

p\_Antennas,

p\_VZWOwned,

p\_LightsReqd,

p\_LightsTested,

p\_LightsOverDue,

p\_LightsOk,

p\_Monitored,

p\_EnvTested,

p\_EnvOverDue,

p\_EnvOK,

SYSDATE );

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( p\_Area

||' - '|| p\_Region

||' - '|| p\_DomainId

||' - '|| p\_DomainName

||' - ERROR: '|| TO\_CHAR(SQLCODE ) ||': '|| SQLERRM, 1, 255));

END;

FUNCTION varchar2date( p\_StrDate IN VARCHAR2 )

RETURN DATE IS

retval DATE := NULL;

BEGIN retval := TO\_DATE( rtrim(ltrim( p\_StrDate)), 'MM-DD-YY');

RETURN (retval);

EXCEPTION WHEN OTHERS THEN

BEGIN retval := TO\_DATE( rtrim(ltrim( p\_StrDate )), 'DD-MON-YY');

RETURN (retval);

EXCEPTION WHEN OTHERS THEN

BEGIN retval := NULL;

RETURN (retval);

END;

END;

END varchar2date;

PROCEDURE removeDetails( p\_DomainId IN NOCC\_LAST\_TEST\_SUMMARY.domain\_id%TYPE ) IS

BEGIN

-- Delete data from BOTH last tested reports tables...

DELETE FROM xng\_reports.NOCC\_LAST\_TEST\_DETAILS WHERE domain\_id = p\_DomainId;

DELETE FROM xng\_reports.NOCC\_LAST\_TEST\_SUMMARY WHERE domain\_id = p\_DomainId;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line('Error when deleting DOMIAN: '|| p\_DomainId );

RAISE;

END;

END; -- END of site\_tested package...

/

--------------------------------------------------------

-- DDL for Package Body RPT\_STD\_NOCC\_SITE\_TEST

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."RPT\_STD\_NOCC\_SITE\_TEST"

IS

--

-- Purpose: NOCC SITE ALARAMS AGGREAGATE SUMMARY and DETAIL reports.

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- ---------------------- ------------ -----------------------------------------

-- Krishna Gadde Oct 2005 INitial release.

-- Krihsna Gadde March 2006 Tunned...

-- Krishna Gadde Aug 2006 fixed query. Issue with val\_attr\_group

-- Catherine Sayre Feb 2012 added DORMANT as site\_type to csr\_getAllSites

--

PROCEDURE getAllDomains

IS

TYPE l\_cursor\_Type IS REF CURSOR;

cursor\_Domains l\_cursor\_Type;

l\_Area xng\_reports.VZW\_Network\_Org.Area%TYPE;

l\_Region xng\_reports.VZW\_Network\_Org.Region%TYPE;

l\_DomainId vzwnet.Domain\_inst.domain\_inst\_id%TYPE;

l\_DomainName vzwnet.Domain\_inst.domain\_name%TYPE;

sql\_Str VARCHAR2 (555);

BEGIN

sql\_Str := 'SELECT vno.Area, vno.Region, di.Domain\_inst\_id, di.domain\_name

FROM xng\_reports.VZW\_NetWork\_Org vno, vzwnet.Domain\_Inst di

WHERE di.domain\_name = vno.region\_domain\_name

and di.domain\_inst\_id != 1012

GROUP BY vno.Area, vno.Region, di.Domain\_inst\_id, di.domain\_name

ORDER BY vno.Area, vno.Region';

OPEN cursor\_Domains FOR sql\_Str;

LOOP

FETCH cursor\_Domains INTO l\_Area, l\_Region, l\_DomainId, l\_DomainName;

EXIT WHEN cursor\_Domains%NOTFOUND;

DBMS\_OUTPUT.put\_line('Loading Domain...: ' || l\_DomainName );

-- Remove the old Detail data by domain's...

removeDetails( l\_DomainId );

-- get all Sites for a passed domain...

getAllSites( l\_DomainId, l\_DomainName, l\_Area, l\_Region );

COMMIT;

END LOOP;

CLOSE cursor\_Domains;

-- Not enough rollback segment...Moved up into the loop COMMIT;

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

IF( cursor\_Domains%ISOPEN ) THEN

CLOSE cursor\_Domains;

END IF;

DBMS\_OUTPUT.put\_line( ' Area : ' || l\_Area ||

' Region : ' || l\_Region ||

' DomainId : ' || l\_DomainId ||

' DomainName:' || l\_DomainName ||

' SQLCODE: ' || TO\_CHAR(SQLCODE));

RAISE;

END;

PROCEDURE getAllSites(

p\_DomainId IN vzwnet.Domain\_Inst.domain\_inst\_id%TYPE,

p\_DomainName IN vzwnet.Domain\_Inst.domain\_name%TYPE,

p\_Area IN xng\_reports.VZW\_Network\_Org.Area%TYPE,

p\_Region IN xng\_reports.VZW\_Network\_Org.Region%TYPE)

IS

-- For NOC\_Aggs Summary report....

l\_EquipInstId vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE;

l\_TowerStatus vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE;

l\_AntennaVZWOwned VARCHAR2(255);

l\_HasAntennaTower VARCHAR2(30);

l\_LightingRequired VARCHAR2(30);

l\_SiteInstId NOCC\_LAST\_TEST\_DETAILS.site\_inst\_id%TYPE;

l\_SiteName NOCC\_LAST\_TEST\_DETAILS.site\_name%TYPE;

l\_SiteId NOCC\_LAST\_TEST\_DETAILS.site\_id%TYPE; -- Site\_Inst.Base\_Num

l\_SiteTechName NOCC\_LAST\_TEST\_DETAILS.site\_tech\_name%TYPE;

l\_inventoried\_cells NOCC\_LAST\_TEST\_DETAILS.inventoried\_cells%TYPE;

l\_SiteType NOCC\_LAST\_TEST\_DETAILS.site\_type%TYPE; -- Site\_Inst.Num

l\_NOCC\_Monitored NOCC\_LAST\_TEST\_DETAILS.nocc\_monitored%TYPE;

l\_LightLastTestedDate NOCC\_LAST\_TEST\_DETAILS.light\_last\_tested%TYPE;

l\_LightNextTestDate NOCC\_LAST\_TEST\_DETAILS.light\_next\_test%TYPE;

l\_LightTestOverDue NOCC\_LAST\_TEST\_DETAILS.light\_test\_overdue%TYPE;

l\_EnvLastTestedDate NOCC\_LAST\_TEST\_DETAILS.env\_last\_tested%TYPE;

l\_EnvNextTestDate NOCC\_LAST\_TEST\_DETAILS.env\_next\_test%TYPE;

l\_EnvTestOverDue NOCC\_LAST\_TEST\_DETAILS.env\_test\_overdue%TYPE;

-- For NOC\_Aggs Summary report....

l\_Site\_Count NOCC\_LAST\_TEST\_SUMMARY.sites%TYPE := 0;

l\_HasAntennaTower\_Count NOCC\_LAST\_TEST\_SUMMARY.antennas%TYPE := 0;

l\_AntennaVZWOwned\_Count NOCC\_LAST\_TEST\_SUMMARY.vzw\_owned%TYPE := 0;

l\_LightingRequired\_Count NOCC\_LAST\_TEST\_SUMMARY.lights\_reqd%TYPE := 0;

l\_LightTestedDate\_Count NOCC\_LAST\_TEST\_SUMMARY.lights\_tested%TYPE := 0;

l\_LightTestOverDue\_Count NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_overdue%TYPE := 0;

l\_LightTest\_OK\_Count NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_ok%TYPE := 0;

l\_NOCC\_Monitored\_Count NOCC\_LAST\_TEST\_SUMMARY.NOCC\_monitored\_sites%TYPE := 0;

l\_EnvLastTestedDate\_Count NOCC\_LAST\_TEST\_SUMMARY.env\_tested%TYPE := 0;

l\_EnvTestOverDue\_Count NOCC\_LAST\_TEST\_SUMMARY.env\_test\_overdue%TYPE := 0;

l\_EnvTestOK\_Count NOCC\_LAST\_TEST\_SUMMARY.env\_test\_ok%TYPE := 0;

l\_saved\_at\_equpment\_level number := 1;

--## ADD Include List for SITE\_TYPES

CURSOR csr\_getAllSites( pp\_DomainId IN vzwnet.Domain\_Inst.domain\_inst\_id%Type )

IS

SELECT distinct si.Site\_Inst\_Id, si.Site\_hum\_Id, si.Base\_Num, si.Num

FROM vzwnet.Site\_Domain\_map sdm,

vzwnet.Site\_Inst si

WHERE sdm.Domain\_Inst\_Id = pp\_DomainId

AND si.Num in ('CELL','CELL/FUT','COW','COW/FUT','DORMANT','DATA CENTER','ENHAN/FUT',

'ENHANCER','HUB','HUB/FUT','MICRO/FUT','MICROCELL','MICROWAVE',

'MTSO','MTSO/FUT','TANDEM')

AND si.Site\_Inst\_Id = sdm.Site\_Inst\_Id;

--and si.site\_inst\_id not in (select site\_inst\_id from vzwnet.site\_domain\_map where domain\_inst\_id = 1191); --to keep out ALLTEL sites

CURSOR csr\_getSiteEquipment( pp\_SiteInstId IN vzwnet.Site\_Inst.site\_inst\_id%Type )

IS

SELECT ei.Equip\_Inst\_Id

FROM vzwnet.Equip\_Inst ei

WHERE ei.Site\_Inst\_Id (+) = pp\_SiteInstId

AND UPPER(ei.type (+)) = 'ANTENNA SUPPORT STRUCTURE';

BEGIN

OPEN csr\_getAllSites( p\_DomainId );

LOOP

FETCH csr\_getAllSites INTO l\_SiteInstId, l\_SiteName, l\_SiteId, l\_SiteType;

EXIT WHEN csr\_getAllSites%NOTFOUND;

l\_NOCC\_Monitored := NULL;

l\_SiteTechName := NULL;

l\_inventoried\_cells := NULL;

l\_TowerStatus := NULL;

l\_EnvLastTestedDate := NULL;

l\_EnvNextTestDate := NULL;

l\_EnvTestOverDue := NULL;

l\_saved\_at\_equpment\_level := 1;

l\_Site\_Count := l\_Site\_Count + 1;

l\_SiteTechName := getSiteTechName( l\_SiteInstId );

l\_inventoried\_cells := getCellEquipDesc( l\_SiteInstId );

-- Environmental details...

l\_NOCC\_Monitored := getIsNOCCMonitored( l\_SiteInstId );

IF( upper(l\_NOCC\_Monitored) <> 'NO' ) THEN

l\_NOCC\_Monitored\_Count := l\_NOCC\_Monitored\_Count + 1;

l\_EnvLastTestedDate := varchar2date( getEnvLastTestedDate( l\_SiteInstId, l\_SiteType ));

IF( l\_EnvLastTestedDate IS NOT NULL ) THEN

l\_EnvLastTestedDate\_Count := l\_EnvLastTestedDate\_Count + 1;

-- Changed according to Jim 2/6/2006, --IF( (upper(l\_SiteType) = 'MTSO' ) OR (upper(l\_SiteType) = 'HUB' ) ) THEN

IF ( upper(l\_SiteType) = 'MTSO' ) THEN

l\_EnvNextTestDate := TO\_CHAR( ADD\_MONTHS( TO\_DATE( l\_EnvLastTestedDate, 'DD\_MON\_YY'), 6 ));

ELSIF ( upper(l\_SiteType) = 'CELL' ) THEN

-- Conditional interval based on Failed First Environmental Attempt UDA

l\_EnvNextTestDate := TO\_CHAR( ADD\_MONTHS( TO\_DATE( l\_EnvLastTestedDate, 'DD\_MON\_YY'),

12\* getFailedFirstEnvAttempt( l\_SiteInstId )));

ELSE

l\_EnvNextTestDate := TO\_CHAR( ADD\_MONTHS( TO\_DATE( l\_EnvLastTestedDate, 'DD\_MON\_YY'), 12 ));

END IF;

l\_EnvTestOverDue := TO\_CHAR( TO\_DATE( l\_EnvNextTestDate, 'DD\_MON\_YY') - TRUNC(SYSDATE ));

IF( TO\_NUMBER(l\_EnvTestOverDue ) <= 0 ) THEN

DBMS\_OUTPUT.put\_line('OVERDUE 1 '|| l\_SiteInstId || to\_char(l\_EnvTestOverDue\_Count));

l\_EnvTestOverDue\_Count := l\_EnvTestOverDue\_Count + 1;

ELSE

l\_EnvTestOK\_Count := l\_EnvTestOK\_Count + 1;

END IF;

ELSE

DBMS\_OUTPUT.put\_line('OVERDUE 2'|| l\_SiteInstId || to\_char(l\_EnvTestOverDue\_Count));

l\_EnvTestOverDue\_Count := l\_EnvTestOverDue\_Count + 1;

-- added 04/21/2006 JLA

l\_EnvNextTestDate := 'ASAP';

l\_EnvTestOverDue := -90;

--

END IF; -- l\_ENVLastTestedDate

END IF; -- l\_NOCC\_Monitored

l\_HasAntennaTower := NULL;

l\_AntennaVZWOwned := NULL;

l\_LightingRequired := NULL;

l\_LightLastTestedDate := NULL;

l\_LightNextTestDate := NULL;

l\_LightTestOverDue := NULL;

-- Antenna details...

OPEN csr\_getSiteEquipment( l\_SiteInstId );

LOOP

FETCH csr\_getSiteEquipment INTO l\_EquipInstId;

EXIT WHEN csr\_getSiteEquipment%NOTFOUND;

l\_HasAntennaTower := NULL;

l\_AntennaVZWOwned := NULL;

l\_LightingRequired := NULL;

l\_LightLastTestedDate := NULL;

l\_LightNextTestDate := NULL;

l\_LightTestOverDue := NULL;

l\_HasAntennaTower := getHasAntennaTower( l\_SiteInstId, l\_EquipInstId );

IF( l\_HasAntennaTower = 'Has Tower' ) THEN

l\_HasAntennaTower\_Count := l\_HasAntennaTower\_Count + 1;

--l\_HasAntennaTower := 'Has Tower';

l\_AntennaVZWOwned := getIsAntennaVZWOwned( l\_SiteInstId, l\_EquipInstId );

-- IF( upper(l\_AntennaVZWOwned) <> 'OTHER' ) THEN

IF( upper(l\_AntennaVZWOwned) = 'VERIZON WIRELESS' ) THEN

-- AntennaStructure is 'VERIZON WIRELESS' or empty, assume its VZW's.

l\_AntennaVZWOwned\_Count := l\_AntennaVZWOwned\_Count + 1;

l\_LightingRequired := getIsLightingRequired( l\_SiteInstId, l\_EquipInstId );

IF( upper(l\_LightingRequired) <> 'NO' ) THEN

-- If it is clearly specified 'NO' then its no, otherwise its required!.

l\_LightingRequired\_Count := l\_LightingRequired\_Count + 1;

l\_LightLastTestedDate := varchar2date( getTowerLightLastTestedDate( l\_SiteInstId , l\_SiteType ));

IF( l\_LightLastTestedDate IS NOT NULL ) THEN

l\_LightTestedDate\_Count := l\_LightTestedDate\_Count + 1;

l\_LightNextTestDate := TO\_CHAR( TO\_DATE( l\_LightLastTestedDate, 'DD-MON-YY') + 90 );

l\_LightTestOverDue := TO\_CHAR( TO\_DATE( l\_LightNextTestDate, 'DD-MON-YY') - TRUNC(SYSDATE) );

IF ( TO\_NUMBER( l\_LightTestOverDue ) <= 0 ) THEN

l\_LightTestOverDue\_Count := l\_LightTestOverDue\_Count + 1;

l\_LightNextTestDate := 'ASAP';

ELSE

l\_LightTest\_OK\_Count := l\_LightTest\_OK\_Count + 1;

END IF; -- l\_LightTestOverDue

ELSE

l\_LightTestOverDue\_Count := l\_LightTestOverDue\_Count + 1;

--added 04/21/2006 JLA

l\_LightNextTestDate := 'ASAP';

l\_LightTestOverDue := '-90';

--

END IF; --l\_LightLastTestedDate

ELSE

l\_LightLastTestedDate := 'Light Not Reqd';

l\_LightNextTestDate := 'N/A';

END IF; -- l\_LightingRequired

ELSE

l\_AntennaVZWOwned := 'Not VZW Owned';

END IF; -- l\_AntennaVZWOwned

ELSE

l\_LightLastTestedDate := 'No Tower';

l\_LightNextTestDate := 'N/A';

END IF; -- l\_HasAntennaTower

-- Save detail record...

IF p\_DomainName != 'OSS\_DOMAIN'

THEN

Insert\_NOCC\_Last\_Test\_Details( p\_DomainId,

p\_DomainName,

l\_SiteInstId,

l\_SiteName,

l\_SiteId,

l\_SiteType,

l\_SiteTechName,

l\_inventoried\_cells,

l\_NOCC\_Monitored,

l\_AntennaVZWOwned,

l\_LightingRequired,

l\_LightLastTestedDate,

l\_LightNextTestDate,

l\_LightTestOverDue,

l\_EnvLastTestedDate,

l\_EnvNextTestDate,

l\_EnvTestOverDue,

l\_EquipInstId );

END IF;

l\_saved\_at\_equpment\_level := 2;

END LOOP;

CLOSE csr\_getSiteEquipment;

if ( l\_saved\_at\_equpment\_level = 1 ) then

-- Save detail record...

IF p\_DomainName != 'OSS\_DOMAIN'

THEN

Insert\_NOCC\_Last\_Test\_Details( p\_DomainId,

p\_DomainName,

l\_SiteInstId,

l\_SiteName,

l\_SiteId,

l\_SiteType,

l\_SiteTechName,

l\_inventoried\_cells,

l\_NOCC\_Monitored,

l\_AntennaVZWOwned,

l\_LightingRequired,

l\_LightLastTestedDate,

l\_LightNextTestDate,

l\_LightTestOverDue,

l\_EnvLastTestedDate,

l\_EnvNextTestDate,

l\_EnvTestOverDue,

l\_EquipInstId );

END IF;

end if;

END LOOP;

-- Save domain level test summary/count record...

IF p\_Region != 'OSS'

THEN

insert\_NOCC\_LAST\_TEST\_SUMMARY( p\_Area,

p\_Region,

p\_DomainId,

p\_DomainName,

l\_Site\_Count,

l\_HasAntennaTower\_Count,

l\_AntennaVZWOwned\_Count,

l\_LightingRequired\_Count,

l\_LightTestedDate\_Count,

l\_LightTestOverDue\_Count,

l\_LightTest\_OK\_Count,

l\_NOCC\_Monitored\_Count,

l\_EnvLastTestedDate\_Count,

l\_EnvTestOverDue\_Count,

l\_EnvTestOK\_Count );

END IF;

CLOSE csr\_getAllSites;

EXCEPTION

WHEN OTHERS THEN

IF( csr\_getAllSites%ISOPEN ) THEN

CLOSE csr\_getAllSites;

END IF;

IF( csr\_getSiteEquipment%ISOPEN ) THEN

CLOSE csr\_getSiteEquipment;

END IF;

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in GetAllSites(DomainId='

|| p\_DomainId || ' ) -> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

/\*

FUNCTION getSiteTechName( p\_SiteInstId IN vzwnet.Site\_attr\_settings.Site\_Inst\_Id%TYPE )

RETURN vzwnet.Site\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Site\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT sas.attr\_value

INTO retval

FROM vzwnet.Site\_attr\_settings sas,

vzwnet.val\_attr\_Name van

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND sas.val\_attr\_Inst\_Id = van.val\_attr\_Inst\_Id

AND upper(van.attr\_Name) = 'SITE TECHNICIAN'

AND upper(van.group\_Name) = 'EMERGENCY CONTACTS';

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN null;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getSiteTechName('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

\*/

FUNCTION getSiteTechName( p\_SiteInstId IN vzwnet.Site\_attr\_settings.Site\_Inst\_Id%TYPE )

RETURN vzwnet.Site\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Site\_attr\_settings.attr\_value%TYPE;

BEGIN

select

min(ri.name) name into retval

from

vzwnet.resource\_inst ri

join vzwnet.resource\_definition\_inst di

on di.DEFINITION\_INST\_ID = ri.DEFINITION\_INST\_ID

join vzwnet.resource\_associations ra

on ra.resource\_inst\_id = ri.resource\_inst\_id

where

ra.target\_type\_id = 16

and di.name = 'Operations'

and ri.category = 'Technician'

and ra.TARGET\_INST\_ID=p\_SiteInstId

group by ra.target\_inst\_id

;

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN null;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getSiteTechName('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getCellEquipDesc( p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE )

RETURN NOCC\_LAST\_TEST\_DETAILS.INVENTORIED\_CELLS%TYPE

IS

l\_Cell\_Equipment VARCHAR2(3099 );

l\_Equip\_descr VARCHAR2(255 );

CURSOR cursor\_Equipment\_desc( pp\_SiteInstId IN VARCHAR2 )

IS

SELECT NVL( ei.DESCR, 'none' )

FROM vzwnet.Equip\_Inst ei

WHERE ei.Site\_Inst\_Id = pp\_SiteInstId

AND ei.TYPE IN( SELECT xp\_Equip\_Type

FROM xng\_apps.VZ\_Ntwk\_Node\_Map

WHERE end\_point\_Id = 'CELL DEMARK' );

BEGIN

OPEN cursor\_Equipment\_desc( p\_SiteInstId );

LOOP

FETCH cursor\_Equipment\_desc INTO l\_Equip\_descr;

EXIT WHEN cursor\_Equipment\_desc%NOTFOUND;

IF ( l\_Cell\_Equipment IS NULL ) THEN

l\_Cell\_Equipment := l\_Equip\_descr;

ELSE

l\_Cell\_Equipment := l\_Cell\_Equipment || ';' || l\_Equip\_descr;

END IF;

END LOOP;

CLOSE cursor\_Equipment\_desc;

Return l\_Cell\_Equipment;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in loadEquipDesc('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

IF( cursor\_Equipment\_desc%ISOPEN ) THEN

CLOSE cursor\_Equipment\_desc;

END IF;

END;

FUNCTION getIsNOCCMonitored( p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE )

RETURN VARCHAR2

IS

retval VARCHAR2(25);

BEGIN

SELECT NVL(sas.Attr\_value, 'NO' )

INTO retval

FROM vzwnet.Site\_Attr\_settings sas,

vzwnet.val\_Attr\_Name van

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND van.val\_Attr\_Inst\_Id = sas.val\_Attr\_Inst\_Id

AND UPPER(van.Attr\_Name ) = 'ENVIRONMENTALS NOCC MONITORED'

AND UPPER(van.group\_Name ) = 'SITE ALARM VERIFICATION TEST';

-- added 04/21/2006 JLA

IF upper(retval) != 'NO' THEN

retval := 'YES';

ELSE

retval := 'NO';

END IF;

--

RETURN trim( retval );

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'No';

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line( SUBSTR( 'Error, in getIsNoccMonitored('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) ||': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getFailedFirstEnvAttempt( p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE )

RETURN VARCHAR2

IS

retval NUMBER;

BEGIN

SELECT decode(upper(sas.Attr\_value),null,1,

'NO',2,

'YES',1,1)

INTO retval

FROM vzwnet.Site\_Attr\_settings sas,

vzwnet.val\_Attr\_Name van

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND van.val\_Attr\_Inst\_Id = sas.val\_Attr\_Inst\_Id

AND UPPER(van.Attr\_Name ) = 'FAILED FIRST ENV ATTEMPT'

AND UPPER(van.group\_Name ) = 'SITE ALARM VERIFICATION TEST';

-- retval is a multiplier for 1 or 2 year Env Testing interval for CELL Sites

-- IF upper(retval) = 'NO' and retval is not null THEN

-- retval := 2;

-- ELSE

-- retval := 1;

-- END IF;

RETURN ( retval );

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 1;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line( SUBSTR( 'Error, in getFailedFirstEnvAttempt('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) ||': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getEnvLastTestedDate(

p\_SiteInstId IN vzwnet.Site\_attr\_settings.Site\_Inst\_Id%TYPE,

p\_SiteType IN vzwnet.val\_attr\_group.class\_category%TYPE )

RETURN vzwnet.Site\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Site\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT sas.attr\_value

INTO retval

FROM vzwnet.Site\_attr\_settings sas,

vzwnet.val\_attr\_Name van,

vzwnet.val\_attr\_group vag

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND van.val\_attr\_Inst\_Id = sas.val\_attr\_Inst\_Id

AND UPPER(van.attr\_Name ) = 'ENVIRONMENT DATE LAST TESTED'

AND UPPER(vag.group\_Name ) = UPPER(van.group\_Name )

AND vag.class\_category = p\_SiteType

AND vag.appl\_class = 'I'

AND UPPER(vag.group\_Name ) = 'SITE ALARM VERIFICATION TEST';

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN null;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getENVLastTestedDate('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getHasAntennaTower(

p\_SiteInstId IN vzwnet.Site\_Inst.Site\_Inst\_Id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.Equip\_Inst\_Id%TYPE )

RETURN VARCHAR2

IS

retval VARCHAR2(20);

BEGIN

SELECT NVL2(ei.Equip\_Inst\_Id, 'Has Tower', 'No Tower' ) -- some sites have multiple antennas

INTO retval

FROM vzwnet.Equip\_Inst ei,

vzwnet.Site\_Inst si

WHERE ei.Equip\_Inst\_Id = p\_EquipInstId

AND UPPER(ei.TYPE) = 'ANTENNA SUPPORT STRUCTURE'

AND si.Site\_inst\_id = ei.site\_inst\_id

AND si.Site\_Inst\_Id = p\_SiteInstId;

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'No Tower';

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getHasAntennaTower('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getIsAntennaVZWOwned(

p\_SiteInstId IN vzwnet.Equip\_Inst.Site\_Inst\_Id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.equip\_inst\_id%TYPE )

RETURN vzwnet.Equip\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Equip\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT UPPER( eas.attr\_value )

INTO retval

FROM vzwnet.Equip\_Inst ei,

vzwnet.Equip\_attr\_settings eas,

vzwnet.val\_attr\_Name van

WHERE ei.Site\_Inst\_Id = p\_SiteInstId

AND ei.equip\_inst\_id = p\_EquipInstId

AND ei.type = 'ANTENNA SUPPORT STRUCTURE'

AND eas.Equip\_Inst\_Id = ei.Equip\_Inst\_Id

AND van.val\_attr\_Inst\_Id = eas.val\_attr\_Inst\_Id

AND UPPER(van.attr\_Name ) = 'OWNER'

AND UPPER(van.group\_Name ) = 'ANTENNA SUPPORT STRUCTURE';

-- added 04/21/2006 JLA

IF upper(retval) = 'OTHER' THEN

retval := 'OTHER';

ELSE

retval := 'VERIZON WIRELESS';

END IF;

--

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'VERIZON WIRELESS';

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line( SUBSTR( 'Error, in getIsAntennaVZWOwned('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getIsLightingRequired(

p\_SiteInstId IN vzwnet.Equip\_Inst.Site\_Inst\_Id%TYPE,

p\_EquipInstId IN vzwnet.Equip\_Inst.equip\_inst\_id%TYPE )

RETURN vzwnet.Equip\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Equip\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT UPPER(eas.attr\_value)

INTO retval

FROM vzwnet.Equip\_Inst ei,

vzwnet.Equip\_attr\_settings eas,

vzwnet.val\_attr\_Name van

-- Krishna 8/4/2006 ,vzwnet.val\_attr\_group vag

WHERE ei.Site\_Inst\_Id = p\_SiteInstId

AND ei.equip\_inst\_id = p\_EquipInstId

AND eas.Equip\_Inst\_Id = ei.Equip\_Inst\_Id

AND van.val\_attr\_Inst\_Id = eas.val\_attr\_Inst\_Id

AND UPPER(van.attr\_Name ) = 'LIGHTING REQUIRED'

--Krishna 8/4/2006 AND UPPER(vag.group\_Name ) = UPPER(van.group\_Name )

AND UPPER(van.group\_Name ) = 'FAA INFORMATION';

-- added 04/21/2006 JLA

IF upper(retval) = 'NO' THEN

retval := 'NO';

ELSE

retval := 'YES';

END IF;

--

RETURN retval;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'YES'; -- if its empty then we are treating it as Yes.

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( 'Error, in getIsLightingRequired('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

FUNCTION getTowerLightLastTestedDate(

p\_SiteInstId IN vzwnet.Site\_attr\_settings.Site\_Inst\_Id%TYPE,

p\_SiteType IN vzwnet.val\_attr\_group.class\_category%TYPE )

RETURN vzwnet.Site\_attr\_settings.attr\_value%TYPE

IS

retval vzwnet.Site\_attr\_settings.attr\_value%TYPE;

BEGIN

SELECT sas.attr\_value

INTO retval

FROM vzwnet.Site\_attr\_settings sas,

vzwnet.val\_attr\_Name van,

vzwnet.val\_attr\_group vag

WHERE sas.Site\_Inst\_Id = p\_SiteInstId

AND van.val\_attr\_Inst\_Id = sas.val\_attr\_Inst\_Id

AND UPPER(van.attr\_Name ) = 'TOWER LIGHT DATE LAST TESTED'

AND UPPER(vag.group\_Name ) = UPPER(van.group\_Name )

AND UPPER(vag.class\_category ) = p\_SiteType

AND UPPER(vag.group\_Name ) = 'SITE ALARM VERIFICATION TEST';

RETURN trim(retval);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN null;

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line( SUBSTR( 'Error, in getTowerLightLastTestedDate('

|| p\_SiteInstId || ' )-> '

|| TO\_CHAR(SQLCODE ) || ': '|| SQLERRM, 1, 255 ) );

END;

PROCEDURE insert\_NOCC\_Last\_Test\_Details(

p\_DomainId IN NOCC\_LAST\_TEST\_DETAILS.domain\_Id%TYPE,

p\_DomainName IN NOCC\_LAST\_TEST\_DETAILS.domain\_name%TYPE,

p\_SiteInstId IN NOCC\_LAST\_TEST\_DETAILS.site\_Inst\_Id%TYPE,

p\_SiteName IN NOCC\_LAST\_TEST\_DETAILS.site\_name%TYPE,

p\_SiteId IN NOCC\_LAST\_TEST\_DETAILS.site\_id%TYPE,

p\_Sitetype IN NOCC\_LAST\_TEST\_DETAILS.site\_type%TYPE,

p\_SiteTechName IN NOCC\_LAST\_TEST\_DETAILS.site\_tech\_name%TYPE,

p\_inventoried\_cells IN NOCC\_LAST\_TEST\_DETAILS.inventoried\_cells%TYPE,

p\_NOCC\_Monitored IN NOCC\_LAST\_TEST\_DETAILS.nocc\_Monitored%TYPE,

p\_vzw\_owned IN NOCC\_LAST\_TEST\_DETAILS.vzw\_owned%TYPE,

p\_lights\_reqd IN NOCC\_LAST\_TEST\_DETAILS.lights\_reqd%TYPE,

p\_LightLastTested IN NOCC\_LAST\_TEST\_DETAILS.Light\_last\_tested%TYPE,

p\_LightNextTest IN NOCC\_LAST\_TEST\_DETAILS.Light\_next\_test%TYPE,

p\_LightTestOverDue IN NOCC\_LAST\_TEST\_DETAILS.light\_test\_overdue%TYPE,

p\_ENVLastTested IN NOCC\_LAST\_TEST\_DETAILS.env\_last\_tested%TYPE,

p\_ENVNextTest IN NOCC\_LAST\_TEST\_DETAILS.env\_next\_test%TYPE,

p\_ENVTestOverDue IN NOCC\_LAST\_TEST\_DETAILS.env\_test\_overdue%TYPE,

p\_EquipInstid IN NOCC\_LAST\_TEST\_DETAILS.equip\_inst\_id%TYPE )

IS

BEGIN

INSERT INTO NOCC\_LAST\_TEST\_DETAILS ( Domain\_Id,

Domain\_Name,

Site\_Inst\_Id,

Site\_Name,

Site\_ID,

Site\_Type,

Site\_Tech\_Name,

inventoried\_cells,

NOCC\_Monitored,

vzw\_owned,

lights\_reqd,

Light\_Last\_Tested,

Light\_Next\_Test,

Light\_Test\_OverDue,

ENV\_Last\_Tested,

ENV\_Next\_Test,

ENV\_Test\_OverDue,

Run\_Date,

Equip\_inst\_id )

VALUES(

p\_DomainId,

p\_DomainName,

p\_SiteInstId,

p\_SiteName,

p\_SiteID,

p\_SiteType,

p\_SiteTechName,

p\_inventoried\_cells,

p\_NOCC\_Monitored,

p\_vzw\_owned,

p\_lights\_reqd,

p\_LightLastTested,

p\_LightNextTest,

p\_LightTestOverDue,

p\_ENVLastTested,

p\_ENVNextTest,

p\_ENVTestOverDue,

SYSDATE,

p\_EquipInstid );

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line('Error Insert\_NOCC\_LAST\_TEST\_DETAILS()-> '

||' - '|| p\_DomainId

||' - '|| p\_DomainName

||' - '|| p\_SiteInstId

||' - '|| p\_SiteName

||' - '|| TO\_CHAR(SQLCODE));

END;

PROCEDURE insert\_NOCC\_LAST\_TEST\_SUMMARY(

p\_Area IN NOCC\_LAST\_TEST\_SUMMARY.Area%TYPE,

p\_Region IN NOCC\_LAST\_TEST\_SUMMARY.Region%TYPE,

p\_DomainId IN NOCC\_LAST\_TEST\_SUMMARY.domain\_id%TYPE,

p\_DomainName IN NOCC\_LAST\_TEST\_SUMMARY.domain\_name%TYPE,

p\_Sites IN NOCC\_LAST\_TEST\_SUMMARY.sites%TYPE,

p\_Antennas IN NOCC\_LAST\_TEST\_SUMMARY.antennas%TYPE,

p\_VZWOwned IN NOCC\_LAST\_TEST\_SUMMARY.vzw\_owned%TYPE,

p\_LightsReqd IN NOCC\_LAST\_TEST\_SUMMARY.lights\_reqd%TYPE,

p\_LightsTested IN NOCC\_LAST\_TEST\_SUMMARY.lights\_tested%TYPE,

p\_LightsOverDue IN NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_overdue%TYPE,

p\_LightsOk IN NOCC\_LAST\_TEST\_SUMMARY.lights\_test\_ok%TYPE,

p\_Monitored IN NOCC\_LAST\_TEST\_SUMMARY.NOCC\_monitored\_sites%TYPE,

p\_EnvTested IN NOCC\_LAST\_TEST\_SUMMARY.env\_tested%TYPE,

p\_EnvOverDue IN NOCC\_LAST\_TEST\_SUMMARY.env\_test\_overdue%TYPE,

p\_EnvOK IN NOCC\_LAST\_TEST\_SUMMARY.env\_test\_ok%TYPE

)

IS

BEGIN

INSERT INTO NOCC\_LAST\_TEST\_SUMMARY( Area,

Region,

Domain\_Id,

Domain\_Name,

Sites,

Antennas,

VZW\_Owned,

Lights\_REQD,

Lights\_Tested,

Lights\_Test\_OverDue,

Lights\_Test\_Ok,

NOCC\_Monitored\_Sites,

ENV\_Tested,

ENV\_Test\_OverDue,

ENV\_Test\_Ok,

Run\_Date )

VALUES( p\_Area,

p\_Region,

p\_DomainId,

p\_DomainName,

p\_Sites,

p\_Antennas,

p\_VZWOwned,

p\_LightsReqd,

p\_LightsTested,

p\_LightsOverDue,

p\_LightsOk,

p\_Monitored,

p\_EnvTested,

p\_EnvOverDue,

p\_EnvOK,

SYSDATE );

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line(SUBSTR( p\_Area

||' - '|| p\_Region

||' - '|| p\_DomainId

||' - '|| p\_DomainName

||' - ERROR: '|| TO\_CHAR(SQLCODE ) ||': '|| SQLERRM, 1, 255));

END;

FUNCTION varchar2date( p\_StrDate IN VARCHAR2 )

RETURN DATE IS

retval DATE := NULL;

BEGIN retval := TO\_DATE( rtrim(ltrim( p\_StrDate)), 'MM-DD-YY');

RETURN (retval);

EXCEPTION WHEN OTHERS THEN

BEGIN retval := TO\_DATE( rtrim(ltrim( p\_StrDate )), 'DD-MON-YY');

RETURN (retval);

EXCEPTION WHEN OTHERS THEN

BEGIN retval := NULL;

RETURN (retval);

END;

END;

END varchar2date;

PROCEDURE removeDetails( p\_DomainId IN NOCC\_LAST\_TEST\_SUMMARY.domain\_id%TYPE ) IS

BEGIN

-- Delete data from BOTH last tested reports tables...

DELETE FROM xng\_reports.NOCC\_LAST\_TEST\_DETAILS WHERE domain\_id = p\_DomainId;

DELETE FROM xng\_reports.NOCC\_LAST\_TEST\_SUMMARY WHERE domain\_id = p\_DomainId;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line('Error when deleting DOMIAN: '|| p\_DomainId );

RAISE;

END;

END; -- END of site\_tested package...

/

--------------------------------------------------------

-- DDL for Package Body SAM\_EBH\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."SAM\_EBH\_AUDIT"

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure auditAlngMLS \*

\* \*

\* Purpose: this method compares Xng data to discovered SAM data \*

\* and records only the matched equip \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE insert\_al\_ngmls\_equip\_issues (processname VARCHAR2)

IS

methodname VARCHAR2 (30) := 'insert\_AL\_ngMLS\_equip\_issues';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

-- if some ne\_hostname matched > 1 granite ne\_inst\_id then mark it

--Duplicate in NE

sqlstmt :=

'insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select site\_name, 10, ''AL''

from sam\_ebh\_health i

group by i.site\_name

having count(server\_name) > 1';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK issue Duplicate in NE ';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

BEGIN

-- if some ne\_hostname matched > 1 granite ne\_inst\_id then mark it

--Duplicate in GI

sqlstmt :=

' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_wk

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 1, ''AL''

from SAM\_GI\_AL\_NGMLS\_AUDIT\_WK i

where match\_code <> ''Xng Only''

group by i.ngMLS\_device\_name

having count(1) > 1';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t mark Duplicate in Granite';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- check the ne\_hostname matches MTCE std, else update match\_code = ''NE Only''

-- and match\_status = ''ne\_hostname not MTCE Stds compliant''

sqlstmt :=

' update sam\_gi\_al\_ngMLS\_audit\_wk nxa set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

where exists (select 1

from sam\_ebh\_health i

where ngmls\_vendor = ''AL''

and nxa.NGMLS\_DEVICE\_NAME = I.SITE\_NAME

and parse\_status like ''%11%''

)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update name does not match MTCE Std';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

sqlstmt :=

' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 11, ngMLS\_vendor

from sam\_ebh\_health n, sam\_gi\_al\_ngmls\_audit\_wk nxa

where

nxa.NGMLS\_VENDOR = ''AL''

and nxa.NGMLS\_DEVICE\_NAME = N.SITE\_NAME

and n.parse\_status like ''%11%''';

--and regexp\_substr(parse\_status, '[[:digit:]]+') = '11'

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert err "Does not match MTCE std" in NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: Checking GI Data Std');

sqlstmt :=

' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 7, ngMLS\_vendor

from xng\_ngmls\_equip\_wk

where ngmls\_vendor = ''AL''

and eq\_class\_type = ''NE''

and parse\_status like ''%7%''';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update eq name does not match GI Std';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- check if ne\_hostnames' don't match first 6-char match some MTSO CLLI

-- then match\_code = ''NE Only'' and match\_status = ''1st 6-chars of don''''t match any MTSO CLLIs''

sqlstmt :=

'insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.ngMLS\_device\_name,

SUBSTR (ei.ngMLS\_device\_name, 1, 8) ngMLS\_clli, cdm6.clli\_6 vsm\_clli

FROM sam\_gi\_al\_ngmls\_audit\_wk ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.ngMLS\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.ngMLS\_device\_name, 1, 8) = cdm.clli)),

BAD\_CLLI AS

(

SELECT ei.ngMLS\_device\_name

FROM sam\_gi\_al\_ngmls\_audit\_wk ei

MINUS

SELECT ei.ngMLS\_device\_name

FROM sam\_gi\_al\_ngmls\_audit\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.ngMLS\_device\_name, 1, 8)

MINUS

SELECT ei.ngMLS\_device\_name

FROM sam\_gi\_al\_ngmls\_audit\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.ngMLS\_device\_name, 1, 6)

),

ISSUES AS

(

SELECT cm.ngMLS\_device\_name, 3 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.ngMLS\_device\_name, 2 issue\_id

FROM bad\_clli bc

)

SELECT i.ngMLS\_device\_name, i.issue\_id, ''AL''

FROM ISSUES i

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

NULL;

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t mark 1st 6-char don''t match MTSO CLLIs';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in GI

sqlstmt :=

' insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select ngMLS\_device\_name, 15, ''AL''

from sam\_gi\_al\_ngmls\_audit\_wk aud

where aud.ne\_inst\_id is null';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname

|| '(): Cant insert issue ''Not in GI''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

sqlstmt :=

'insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK columns (ngMLS\_device\_name, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct ngMLS\_device\_name, 12, ''AL''

from sam\_gi\_al\_ngmls\_audit\_wk aud

where SAM\_ngmls\_DEVICE\_NAME is null';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert into NGMLS\_DEVICE\_AUDIT\_ISSUES\_WK issue 12';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

-- add more errors for not parseable etc????????????????????

BEGIN

-- IF BOTH and has issues then mark it NE

sqlstmt :=

' update sam\_gi\_al\_ngmls\_audit\_wk aud

set match\_code = ''NE Only''

where exists (select 1

from ngMLS\_device\_AUDIT\_ISSUEs\_wk cdai, ngmls\_issue n

where aud.MATCH\_CODE = ''BOTH''

and aud.ngMLS\_vendor = ''AL''

and N.IS\_CRITICAL=''Y''

and N.NGMLS\_ISSUE\_ID = CDAI.NGMLS\_ISSUE\_ID

and AUD.NGMLS\_DEVICE\_NAME = CDAI.NGMLS\_DEVICE\_NAME

)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant update match\_status to ''NE Only'' issue ';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

END insert\_al\_ngmls\_equip\_issues;

PROCEDURE audit\_al\_ngmls\_equip

IS

methodname VARCHAR2 (30) := 'audit\_AL\_ngMLS\_equip';

processname VARCHAR2 (30) := 'SAM\_GI\_NGMLS\_AUDIT';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

-- verified with shirley --- assumption is they wont embed all ngMLSs within a contianer of type ngMLS. This logic will break if they so that.

-- does it matter if the the equipment name in GI does not match Std... verified with Shirley. yes it is fatal

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE 'truncate table sam\_gi\_al\_ngmls\_audit\_wk';

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t delete records from table: SAM\_GI\_AL\_NGMLS\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

BEGIN

DELETE FROM ngmls\_device\_audit\_issues\_wk iss

WHERE ngmls\_vendor = 'AL';

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t delete records from table: ngmls\_device\_audit\_issues\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

/\* audit ngmls devices from SAM vs Xng. Match them on device\_name \*/

BEGIN

sqlstmt :=

' insert into sam\_gi\_al\_ngMLS\_audit\_wk

columns (ngMLS\_vendor, ngMLS\_device\_name

, sam\_ngMLS\_device\_name, sam\_reachability, sam\_resync\_status, sam\_resync\_state, sam\_server\_name, sam\_last\_resync\_start, sam\_last\_resync\_end

, gi\_device\_name, ne\_inst\_id

, Live\_in\_xng, eq\_status

, match\_code

, match\_status

)

WITH

SAM\_ngMLSs as

(

select distinct SITE\_NAME --, act\_mgmt\_ip sam\_ngMLS\_device\_ip

, reachability sam\_reachability

, resync\_status sam\_resync\_status, resync\_state sam\_resync\_state

, server\_name sam\_server\_name, last\_resyn\_start sam\_last\_resync\_start, last\_resyn\_end sam\_last\_resync\_end

from sam\_ebh\_health where device\_type = ''ngMLS''

)

,

Xng as

(

select ngMLS.\*

from xng\_ngMLS\_equip\_wk ngMLS

where ngMLS\_vendor = ''AL''

and ngMLS.eq\_class\_type = ''NE''

)

select distinct ''AL'' ngMLS\_vendor,

case when sam.site\_name is not null then sam.site\_name

when xng.ngMLS\_device\_name is not null then xng.ngMLS\_device\_name

else ''zzzz''||xng.descr

end ngMLS\_device\_name

, site\_name sam\_ngMLS\_device\_name, sam\_reachability, sam\_resync\_status, sam\_resync\_state, sam\_server\_name, sam\_last\_resync\_start, sam\_last\_resync\_end

, xng.descr gi\_device\_name, equip\_inst\_id

, case when status = ''Live'' then ''Y''

when status is not null then ''N''

end Live\_in\_xng

, status

, case when sam.site\_name is not null

and xng.ne\_inst\_id is not null

then ''BOTH''

-- it is in both SAM extracts but not in Xng then NE Only

when sam.site\_name is not null

and xng.ne\_inst\_id is null

then ''NE Only''

when sam.site\_name is null and xng.ne\_inst\_id is not null then ''Xng Only''

else ''XXXXX'' -- when does the control reach here

end match\_code

, parse\_status match\_status

from SAM\_ngMLSs sam

full outer join Xng

on xng.ngmls\_device\_name like ''%''||sam.site\_name ||''%''

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert insert AL ngMLS audit into table: sam\_gi\_al\_ngmls\_audit\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

-- identify all issues with the audited equipment

insert\_al\_ngmls\_equip\_issues (processname);

END;

/\* procedure calculates what % of devices discovered from SAM are found in GI \*/

PROCEDURE load\_ngmls\_regional\_summary

IS

-- get the all ngMLSs found in SAM ngMLS Extract

CURSOR ngmlss\_sam

IS

SELECT area, region, COUNT (\*) sam\_ngmlss

FROM sam\_gi\_al\_ngmls\_audit\_wk

WHERE SAM\_NGMLS\_DEVICE\_NAME IS NOT NULL

GROUP BY ROLLUP (area, region);

-- make sure duplicate matches in GI are not considered as good match-------??????

-- DUP means > 1 Live ones in Xng

-- if 1 is live and others are not, then that is NOT a DUP. Shirley's logic

-- this should be identifiable from the audit

CURSOR sam\_gi\_mismatch\_ngmlss

IS

WITH

--SAM GI mismatch details

sam\_gi\_mismatch AS

(

-- any issue with the device is a mismatch

SELECT DISTINCT cdai.ngmls\_device\_name

FROM ngmls\_device\_audit\_issues\_wk cdai

WHERE ngmls\_vendor = 'AL'),

gud\_matches AS

(

-- all devices which are not in Xng Only

SELECT ngmls\_device\_name

FROM sam\_gi\_al\_ngmls\_audit\_wk aud

WHERE match\_code <> 'Xng Only'

MINUS

-- remove all devices that have critical issues

SELECT cdai.ngmls\_device\_name

FROM ngmls\_device\_audit\_issues\_wk cdai, ngmls\_issue n

WHERE ngmls\_vendor = 'AL'

AND n.is\_critical = 'Y'

AND n.ngmls\_issue\_id = cdai.ngmls\_issue\_id),

sam\_gi\_match AS

(SELECT aud.ngmls\_device\_name,

CASE

WHEN live\_in\_xng = 'Y'

THEN 1

ELSE 0

END sam\_gi\_l\_match,

CASE

WHEN live\_in\_xng <> 'Y'

THEN 1

ELSE 0

END sam\_gi\_nl\_match

FROM sam\_gi\_al\_ngmls\_audit\_wk aud JOIN gud\_matches gud

ON gud.ngmls\_device\_name = aud.ngmls\_device\_name

),

match\_counts AS

(SELECT CASE

WHEN ma.ngmls\_device\_name IS NOT NULL

THEN ma.ngmls\_device\_name

WHEN ms.ngmls\_device\_name IS NOT NULL

THEN ms.ngmls\_device\_name

END ngmls\_device\_name,

COUNT (ma.ngmls\_device\_name) sam\_gi\_match,

SUM (ma.sam\_gi\_l\_match) sam\_gi\_l\_match,

SUM (ma.sam\_gi\_nl\_match) sam\_gi\_nl\_match,

COUNT (ms.ngmls\_device\_name) sam\_gi\_mismatch

FROM sam\_gi\_match ma FULL OUTER JOIN sam\_gi\_mismatch ms

ON ma.ngmls\_device\_name = ms.ngmls\_device\_name

GROUP BY CASE

WHEN ma.ngmls\_device\_name IS NOT NULL

THEN ma.ngmls\_device\_name

WHEN ms.ngmls\_device\_name IS NOT NULL

THEN ms.ngmls\_device\_name

END),

domains AS

(SELECT DISTINCT ngmls\_device\_name, area, region

FROM sam\_gi\_al\_ngmls\_audit\_wk

WHERE region <> 'NNO' AND region <> 'OSS')

SELECT area, region, SUM (sam\_gi\_match) sam\_gi\_match,

SUM (sam\_gi\_l\_match) sam\_gi\_l\_match,

SUM (sam\_gi\_nl\_match) sam\_gi\_nl\_match,

SUM (sam\_gi\_mismatch) sam\_gi\_mismatch

FROM match\_counts cnt JOIN domains di ON cnt.ngmls\_device\_name = di.ngmls\_device\_name

GROUP BY ROLLUP (area, region);

methodname VARCHAR2 (30) := 'load\_ngMLS\_regional\_summary ';

MESSAGE VARCHAR2 (300);

updstmt VARCHAR2 (32767);

updareastmt VARCHAR2 (32767);

updvzwstmt VARCHAR2 (32767);

insstmt VARCHAR2 (32767);

insareastmt VARCHAR2 (32767);

insvzwstmt VARCHAR2 (32767);

sqlstmt VARCHAR2 (32767);

l\_cnt NUMBER;

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE 'truncate table sam\_gi\_al\_ngMLS\_region\_summ\_wk';

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t truncate '

|| ' GI metrics table: sam\_gi\_al\_ngMLS\_region\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

BEGIN

-- from xng insert all CLLI and xng# in the table

sqlstmt :=

'insert into sam\_gi\_al\_ngMLS\_region\_summ\_wk columns (area, region, GI\_NGMLSS, gi\_l\_ngMLSs, gi\_nl\_ngMLSs)

WITH

DOMAINS AS

(

select distinct area, region, market, leaf\_domain\_inst\_id, ngmls\_device\_name

from sam\_gi\_al\_ngmls\_audit\_wk

where region <> ''NNO''

and region <> ''OSS''

union

select ''unknown'' area, ''unknown'' region, ''unknown'' market, null leaf\_domain\_inst\_id, ''unknown'' ngmls\_device\_name

from dual

)

,

Xng as

(

select distinct ngMLS\_device\_name, ngMLS.ne\_inst\_id

, nvl(case when status = ''Live'' then 1 else 0 end, 0) l

, nvl(case when status <> ''Live'' then 1 else 0 end, 0) nl

from XNG\_ngMLS\_equip\_wk ngMLS

where ngMLS\_vendor like ''AL''

and eq\_class\_type = ''NE''

)

,

GI\_CNTS as

(

select nvl(area, ''unknown'') area, nvl(region, ''unknown'') region, case when di.ngmls\_device\_name is null then ''unknown'' else xng.ngmls\_device\_name end ngmls\_device\_name

, sum(l) l

, sum(nl) nl

from DOMAINS di

right outer join Xng

on xng.ngmls\_device\_name = di.ngmls\_device\_name

group by nvl(area, ''unknown''), nvl(region, ''unknown'')

, case when di.ngmls\_device\_name is null then ''unknown'' else xng.ngmls\_device\_name end

)

select di.area, di.region

, sum(nvl(l, 0)) + sum (nvl(nl, 0)) gi\_ngMLSs

, sum(l) gi\_l\_ngMLS

, sum(nl) gi\_nl\_ngMLS

from DOMAINS di

left outer join GI\_CNTS xng

on xng.ngmls\_device\_name = di.ngmls\_device\_name

group by rollup(di.area, di.region)

order by di.area, di.region

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update '

|| ' GI metrics in table: sam\_gi\_al\_ngMLS\_region\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

-- insert new or update metric in the same table for total\_ngMLSs found in SAM in the ngMLS-extract

updstmt :=

'update sam\_gi\_al\_ngMLS\_region\_summ\_wk

set SAM\_ngMLSs = :SAM\_ngMLSs

where area = :area

and region = :region

';

updareastmt :=

'update sam\_gi\_al\_ngMLS\_region\_summ\_wk

set SAM\_ngMLSs = :SAM\_ngMLSs

where area = :area

and region is null

';

updvzwstmt :=

'update sam\_gi\_al\_ngMLS\_region\_summ\_wk

set SAM\_ngMLSs = :SAM\_ngMLSs

where region is null

and area is null

';

FOR rec IN ngmlss\_sam

LOOP

BEGIN

-- dbms\_output.put\_line(rec.area ||'/'|| rec.region||'/'||rec.SAM\_ngMLSs);

IF rec.area IS NULL

THEN

-- dbms\_output.put\_line('In ngMLSs\_SAM VZW');

EXECUTE IMMEDIATE updvzwstmt

USING rec.sam\_ngmlss;

ELSE

IF rec.region IS NULL

THEN

-- dbms\_output.put\_line('In ngMLSs\_SAM Area');

EXECUTE IMMEDIATE updareastmt

USING rec.sam\_ngmlss, rec.area;

ELSE

-- dbms\_output.put\_line('In ngMLSs\_SAM Region');

EXECUTE IMMEDIATE updstmt

USING rec.sam\_ngmlss, rec.area, rec.region;

END IF;

END IF;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update '

|| rec.area

|| '/'

|| rec.region

|| '/'

|| rec.sam\_ngmlss

|| ' for SAM-ngMLS-extract in table: sam\_gi\_al\_ngMLS\_region\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

-- calc matches / mismatches metric between SAM and gi

updstmt :=

'update sam\_gi\_al\_ngMLS\_region\_summ\_wk

set SAM\_gi\_mismatch = :SAM\_gi\_mismatch

, SAM\_gi\_match = :SAM\_gi\_match

, SAM\_gi\_l\_match = :SAM\_gi\_l\_match

, SAM\_gi\_nl\_match = :SAM\_gi\_nl\_match

where area = :area

and region = :region

';

updareastmt :=

'update sam\_gi\_al\_ngMLS\_region\_summ\_wk

set SAM\_gi\_mismatch = :SAM\_gi\_mismatch

, SAM\_gi\_match = :SAM\_gi\_match

, SAM\_gi\_l\_match = :SAM\_gi\_l\_match

, SAM\_gi\_nl\_match = :SAM\_gi\_nl\_match

where area = :area

and region is null

';

updvzwstmt :=

'update sam\_gi\_al\_ngMLS\_region\_summ\_wk

set SAM\_gi\_mismatch = :SAM\_gi\_mismatch

, SAM\_gi\_match = :SAM\_gi\_match

, SAM\_gi\_l\_match = :SAM\_gi\_l\_match

, SAM\_gi\_nl\_match = :SAM\_gi\_nl\_match

where area is null

and region is null

';

FOR rec IN sam\_gi\_mismatch\_ngmlss

LOOP

BEGIN

IF rec.area IS NULL

THEN

EXECUTE IMMEDIATE updvzwstmt

USING rec.sam\_gi\_mismatch,

rec.sam\_gi\_match,

rec.sam\_gi\_l\_match,

rec.sam\_gi\_nl\_match;

ELSE

IF rec.region IS NULL

THEN

EXECUTE IMMEDIATE updareastmt

USING rec.sam\_gi\_mismatch,

rec.sam\_gi\_match,

rec.sam\_gi\_l\_match,

rec.sam\_gi\_nl\_match,

rec.area;

ELSE

EXECUTE IMMEDIATE updstmt

USING rec.sam\_gi\_mismatch,

rec.sam\_gi\_match,

rec.sam\_gi\_l\_match,

rec.sam\_gi\_nl\_match,

rec.area,

rec.region;

END IF;

END IF;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update SAM-GI mismatch in table: sam\_gi\_al\_ngMLS\_dev\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

-- RAISE;

END;

END LOOP;

COMMIT;

-- calculate the percentages

BEGIN

UPDATE sam\_gi\_al\_ngmls\_region\_summ\_wk

SET sam\_gi\_per =

ROUND ( NVL (sam\_gi\_match, 0)

\* 2

/ DECODE (NVL (sam\_ngmlss, 0) + NVL (gi\_ngmlss, 0),

0, 1,

NVL (sam\_ngmlss, 0) + NVL (gi\_ngmlss, 0)

)

\* 100,

2

);

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update percentages in table: sam\_gi\_al\_ngMLS\_region\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END load\_ngmls\_regional\_summary;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* assign area / region to all devices

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE assign\_area\_region\_to\_ngmls

IS

methodname VARCHAR2 (30) := 'assign\_area\_region\_to\_ngMLS';

MESSAGE VARCHAR2 (300);

updstmt VARCHAR2 (32767);

CURSOR area\_region\_dev

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS')

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

ngmls\_device\_name

FROM sam\_gi\_al\_ngmls\_audit\_wk aud JOIN domains di

ON di.clli\_6 = SUBSTR (aud.ngmls\_device\_name, 1, 6)

;

CURSOR eq\_area\_region\_dev

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS')

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

ngmls\_device\_name

FROM sam\_gi\_al\_ngmls\_audit\_wk aud JOIN vzwnet.equip\_domain\_map edm

ON edm.equip\_inst\_id = aud.ne\_inst\_id

JOIN domains di ON di.leaf\_domain\_inst\_id = edm.domain\_inst\_id

WHERE aud.area IS NULL;

CURSOR unknown\_domain

IS

SELECT dlr.area AS area, dlr.region AS region, dlr.market AS market,

dlr.domain\_inst\_id AS domain\_inst\_id,

sgalwk.ngmls\_device\_name AS ngmls\_device\_name

FROM vzwnet.equip\_inst equip,

sam\_gi\_al\_ngmls\_audit\_wk sgalwk,

vzwnet.equip\_domain\_map equipmap,

xng\_reports.domains\_leaf\_reporting dlr

WHERE sgalwk.leaf\_domain\_inst\_id = 0

AND equip.descr = sgalwk.sam\_ngmls\_device\_name

AND equipmap.equip\_inst\_id = equip.equip\_inst\_id

AND dlr.domain\_inst\_id = equipmap.domain\_inst\_id;

BEGIN

updstmt :=

'update SAM\_GI\_AL\_NGMLS\_AUDIT\_WK set area = :area

, region = :region, market = :market

, leaf\_domain\_inst\_id = :leaf\_domain\_inst\_id

where ngMLS\_device\_name = :ngMLS\_device\_name';

FOR rec IN area\_region\_dev

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.ngmls\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region for '

|| rec.ngmls\_device\_name

|| ' in table: SAM\_GI\_AL\_NGMLS\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (methodname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

-- area region for put xng\_only based on the area/region of ne\_inst\_id???????????

FOR rec IN eq\_area\_region\_dev

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.ngmls\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from Xng Only for '

|| rec.ngmls\_device\_name

|| ' in table: SAM\_GI\_AL\_NGMLS\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

BEGIN

-- if there still are some devices which cant be mapped to any domain then mark those as unknown

updstmt :=

' update SAM\_GI\_AL\_NGMLS\_AUDIT\_WK

set area = ''unknown''

, region = ''unknown''

, market = ''unknown''

, leaf\_domain\_inst\_id = 0

where area is null ';

EXECUTE IMMEDIATE updstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from to unknown in table: SAM\_GI\_AL\_NGMLS\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

updstmt :=

' update SAM\_GI\_AL\_NGMLS\_AUDIT\_WK

set area = :area

, region = :region, market = :market

, leaf\_domain\_inst\_id = :DOMAIN\_INST\_ID

where ngMLS\_device\_name = :ngMLS\_device\_name ';

FOR rec IN unknown\_domain

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.domain\_inst\_id,

rec.ngmls\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from Xng Only for '

|| rec.ngmls\_device\_name

|| ' in table: SAM\_GI\_AL\_NGMLS\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

END assign\_area\_region\_to\_ngmls;

PROCEDURE audit\_al\_ngmls\_vlans

IS

methodname VARCHAR2 (30) := 'audit\_al\_ngmls\_vlans';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE ('delete SAM\_NGMLS\_VLAN\_AUDIT\_WK where ngMLS\_vendor = ''AL''');

COMMIT;

sqlstmt :=

'insert into SAM\_NGMLS\_VLAN\_AUDIT\_WK columns (ngMLS\_Device\_Name, ngMLS\_vendor

, ne\_hostName, displayed\_name, node\_id, ne\_vlan\_number

, match\_code, match\_status

, NE\_INST\_ID

, xng\_vlan\_number

, vlan\_inst\_id

, vlan\_status

, port\_name

)

WITH

al\_vlans AS

(SELECT xnvp.\*, xne.ngmls\_device\_name

FROM xng\_ngmls\_vlan\_paths\_wk xnvp JOIN xng\_ngmls\_equip\_wk xne

ON xne.ne\_inst\_id = xnvp.ne\_inst\_id

WHERE xne.ngmls\_vendor = ''AL'' AND xnvp.vlan\_number IS NOT NULL),

matches AS

(SELECT aud.ngmls\_device\_name, sam.out\_encap\_val ne\_vlan\_number,

sam.displayed\_name, sam.node\_id, ''BOTH'' match\_code,

NULL match\_status, xnvp.ne\_inst\_id,

xnvp.vlan\_number xng\_vlan\_number, xnvp.vlan\_inst\_id,

xnvp.vlan\_status, aud.ngmls\_vendor, SUBSTR(sam.PORT\_NAME, 0, INSTR(sam.PORT\_NAME, '':'')-1) AS PORT\_NAME

FROM sam\_ebh\_rtr sam JOIN sam\_gi\_al\_ngmls\_audit\_wk aud

ON aud.ngmls\_device\_name = sam.node\_name

JOIN al\_vlans xnvp

ON aud.ne\_inst\_id = xnvp.ne\_inst\_id

AND sam.out\_encap\_val = xnvp.vlan\_number

WHERE sam.node\_id IS NOT NULL AND sam.device\_type = ''ngMLS''),

ne\_only AS

(SELECT aud.ngmls\_device\_name, sam.displayed\_name, sam.node\_id,

sam.out\_encap\_val ne\_vlan\_number, ''NE Only'' match\_code,

''Missing in Granite'' match\_status, aud.ne\_inst\_id,

null xng\_vlan\_number, NULL vlan\_inst\_id, NULL vlan\_status,

aud.ngmls\_vendor, SUBSTR(sam.PORT\_NAME, 0, INSTR(sam.PORT\_NAME, '':'')-1) AS PORT\_NAME

FROM sam\_ebh\_rtr sam JOIN sam\_gi\_al\_ngmls\_audit\_wk aud

ON aud.ngmls\_device\_name = sam.node\_name

WHERE sam.out\_encap\_val IS NOT NULL

AND sam.node\_id IS NOT NULL

AND sam.device\_type = ''ngMLS''

AND NOT EXISTS (

SELECT 1

FROM al\_vlans xnvp

WHERE aud.ne\_inst\_id = xnvp.ne\_inst\_id

AND xnvp.ngmls\_device\_name = aud.ngmls\_device\_name

AND sam.out\_encap\_val = xnvp.vlan\_number)),

xng\_only AS

(SELECT aud.ngmls\_device\_name, NULL displayed\_name, NULL node\_id,

null ne\_vlan\_number, ''Xng Only'' match\_code,

''Missing in NE'' match\_status, xnvp.ne\_inst\_id,

xnvp.vlan\_number xng\_vlan\_number, xnvp.vlan\_inst\_id, xnvp.vlan\_status,

aud.ngmls\_vendor, NULL PORT\_NAME

FROM al\_vlans xnvp JOIN sam\_gi\_al\_ngmls\_audit\_wk aud

ON aud.ne\_inst\_id = xnvp.ne\_inst\_id

AND AUD.NGMLS\_DEVICE\_NAME = xnvp.ngmls\_device\_name

and not exists(select 1 from sam\_ebh\_rtr sam

where aud.ngmls\_device\_name = sam.node\_name

AND sam.out\_encap\_val = xnvp.vlan\_number

and sam.out\_encap\_val IS NOT NULL AND sam.device\_type = ''ngMLS''))

SELECT ngmls\_device\_name, ngmls\_vendor, ngmls\_device\_name, displayed\_name,

node\_id, ne\_vlan\_number, match\_code, match\_status, ne\_inst\_id,

xng\_vlan\_number, vlan\_inst\_id, vlan\_status, PORT\_NAME

FROM matches

UNION

SELECT ngmls\_device\_name, ngmls\_vendor, ngmls\_device\_name, displayed\_name,

node\_id, ne\_vlan\_number, match\_code, match\_status, ne\_inst\_id,

xng\_vlan\_number, vlan\_inst\_id, vlan\_status, PORT\_NAME

FROM ne\_only

UNION

SELECT ngmls\_device\_name, ngmls\_vendor, NULL, displayed\_name, node\_id,

ne\_vlan\_number, match\_code, match\_status, ne\_inst\_id, xng\_vlan\_number,

vlan\_inst\_id, vlan\_status, PORT\_NAME

FROM xng\_only

';

EXECUTE IMMEDIATE (sqlstmt);

COMMIT;

insert\_al\_ngmls\_vlan\_issues ();

EXCEPTION

WHEN OTHERS

THEN

MESSAGE := 'Error: ' || methodname || '(): Can''t audit AL VLANs';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

/\*watchdog.logerror (methodName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

); \*/

RAISE;

END;

END audit\_al\_ngmls\_vlans;

PROCEDURE insert\_al\_ngmls\_vlan\_issues

IS

methodname VARCHAR2 (30) := 'insert\_AL\_ngMLS\_vlan\_issues';

MESSAGE VARCHAR2 (300);

sqlstmt VARCHAR2 (32767);

CURSOR dup\_l\_vlan

IS

-- if the same DEVICE\_NAME, nxa.VLAN\_number has > 1 Live status then

-- mark it as dup

WITH vlan\_status AS

(SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM sam\_ngmls\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

dup\_l\_vlans AS

(SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number,

COUNT (vlan\_inst\_id)

FROM vlan\_status nxa

WHERE status = 'Live' AND match\_code <> 'Xng Only'

GROUP BY nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, status

HAVING COUNT (vlan\_inst\_id) > 1)

SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM dup\_l\_vlans dup JOIN vlan\_status nxa

ON nxa.ngmls\_device\_name = dup.ngmls\_device\_name

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

ORDER BY nxa.ngmls\_device\_name, nxa.xng\_vlan\_number;

-- if the same ngmls\_device\_name, nxa.VLAN\_number has > 1 status then

-- and 1 is Live and other is not then mark NL as 'NE Only' & 'Live path exists'

CURSOR live\_vlan\_exists

IS

WITH vlan\_status AS

(SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM sam\_ngmls\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

l\_and\_nl\_vlans AS

(SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status = 'Live'

INTERSECT

SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status <> 'Live')

SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number,

nxa.vlan\_inst\_id

FROM l\_and\_nl\_vlans dup JOIN vlan\_status nxa

ON nxa.ngmls\_device\_name = dup.ngmls\_device\_name

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

WHERE nxa.status <> 'Live'

ORDER BY nxa.ngmls\_device\_name, nxa.xng\_vlan\_number;

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE ('delete NGMLS\_VLAN\_AUDIT\_ISSUES\_WK where ngMLS\_vendor = ''AL''');

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t delete rows in table: NGMLS\_VLAN\_AUDIT\_ISSUES\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

-- mark duplicate in Granite as NE Only, or Live Path Exists

-- if there is a path revision then the same VLAN will be present twice.

-- BUT that is NOT a duplicate

-- if NOT a revision path and VLAN# is present multiple times THEN it is dup

-- mark duplicate in Granite as NE Only, or Live Path Exists

-- else 2 DUP vlans on a csr then Duplicate

sqlstmt :=

'update SAM\_NGMLS\_VLAN\_AUDIT\_WK

set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

, match\_status = nvl2(match\_status, match\_status|| '', Dup in GI'', ''Dup in GI'')

where vlan\_inst\_id = :p\_vlan\_inst\_id

and XNG\_VLAN\_NUMBER = :p\_xng\_vlan\_number

and NGMLS\_DEVICE\_NAME = :p\_ngmls\_device\_name

';

FOR rec IN dup\_l\_vlan

LOOP

BEGIN

EXECUTE IMMEDIATE sqlstmt

USING rec.vlan\_inst\_id,

rec.xng\_vlan\_number,

rec.ngmls\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update ''Dup in GI'' in table: SAM\_NGMLS\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

BEGIN

sqlstmt :=

'insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct W.NGMLS\_DEVICE\_NAME

, case when ne\_vlan\_number is null then xng\_vlan\_number

else ne\_vlan\_number

end vlan\_number,1,''AL''

from SAM\_NGMLS\_VLAN\_AUDIT\_wk w

where W.MATCH\_STATUS like ''%Dup in GI%''';

EXECUTE IMMEDIATE sqlstmt;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK issue Duplicate in GI ';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

COMMIT;

BEGIN

-- check the ne\_hostname matches MTCE std, else update match\_code = ''NE Only''

-- and match\_status = ''ne\_hostname not MTCE Stds compliant''

sqlstmt :=

' update SAM\_NGMLS\_VLAN\_AUDIT\_WK aud set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

where exists (select 1

from SAM\_EBH\_RTR i

where DEVICE\_TYPE = ''ngMLS''

and AUD.NGMLS\_DEVICE\_NAME = I.NODE\_NAME

and parse\_status like ''%11%''

)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

sqlstmt :=

'insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct W.NGMLS\_DEVICE\_NAME

, case when ne\_vlan\_number is null then xng\_vlan\_number

else ne\_vlan\_number

end vlan\_number,17,''AL''

from SAM\_NGMLS\_VLAN\_AUDIT\_wk w, SAM\_EBH\_RTR n

where

W.NGMLS\_VENDOR = ''AL''

and W.NGMLS\_DEVICE\_NAME = N.NODE\_NAME

and n.parse\_status like ''%11%''

and n.DEVICE\_TYPE = ''ngMLS''';

--and regexp\_substr(parse\_status, '[[:digit:]]+') = '11'

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update name does not match MTCE Std';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- check if ne\_hostnames' don't match first 6-char match some MTSO CLLI

-- then match\_code = ''NE Only'' and match\_status = ''1st 6-chars of don''''t match any MTSO CLLIs''

sqlstmt :=

'insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.ngMLS\_device\_name,

SUBSTR (ei.ngMLS\_device\_name, 1, 8) ngMLS\_clli, cdm6.clli\_6 vsm\_clli

FROM SAM\_NGMLS\_VLAN\_AUDIT\_wk ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.ngMLS\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.ngMLS\_device\_name, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.ngMLS\_device\_name

FROM SAM\_NGMLS\_VLAN\_AUDIT\_wk ei

MINUS

SELECT ei.ngMLS\_device\_name

FROM SAM\_NGMLS\_VLAN\_AUDIT\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.ngMLS\_device\_name, 1, 8)

MINUS

SELECT ei.ngMLS\_device\_name

FROM SAM\_NGMLS\_VLAN\_AUDIT\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.ngMLS\_device\_name, 1, 6)

),

ISSUES AS

(

SELECT cm.ngMLS\_device\_name, 22 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.ngMLS\_device\_name, 2 issue\_id

FROM bad\_clli bc

)

SELECT distinct i.ngMLS\_device\_name, W.NE\_VLAN\_NUMBER, i.issue\_id, ''AL''

FROM ISSUES i, SAM\_NGMLS\_VLAN\_AUDIT\_wk w

where i.ngMLS\_device\_name = W.NGMLS\_DEVICE\_NAME

and W.NE\_VLAN\_NUMBER is not null

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

NULL;

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t mark 1st 6-char don''t match MTSO CLLIs';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in sam vlan extract

sqlstmt :=

' insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct aud.NGMLS\_DEVICE\_NAME, xng\_vlan\_number, 13, ''AL''

from SAM\_NGMLS\_VLAN\_AUDIT\_wk aud

where aud.ne\_hostname is null';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant insert issue ''Not in SAM VLAN Extract''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in GI

sqlstmt :=

' insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

select distinct aud.NGMLS\_DEVICE\_NAME, ne\_vlan\_number, 15, ''AL''

from SAM\_NGMLS\_VLAN\_AUDIT\_wk aud

where aud.vlan\_inst\_id is null';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname

|| '(): Cant insert issue ''Not in GI''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

-- if there is a path revision then the same VLAN will be present twice.

-- BUT that is NOT a duplicate

-- if NOT a revision path and VLAN# is present multiple times THEN it is dup

-- if 1 == Live and the other <> Live then mark <> Live as 'Live in Xng'

sqlstmt :=

'update SAM\_NGMLS\_VLAN\_AUDIT\_WK

set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

, match\_status = nvl2(match\_status, ''Live Path Exists,'' ||match\_status , ''Live Path Exists'')

where vlan\_inst\_id = :p\_vlan\_inst\_id

and XNG\_VLAN\_NUMBER = :p\_xng\_vlan\_number

and NGMLS\_DEVICE\_NAME = :p\_ngmls\_device\_name

--and ngmls\_vendor = :p\_ngMLS\_vendor

';

FOR rec IN live\_vlan\_exists

LOOP

BEGIN

EXECUTE IMMEDIATE sqlstmt

USING rec.vlan\_inst\_id,

rec.xng\_vlan\_number,

rec.ngmls\_device\_name; --, p\_ngMLS\_vendor;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update ''Live Path Exists'' in table: SAM\_NGMLS\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

BEGIN

-- IF BOTH and has issues then mark it NE

sqlstmt :=

' update SAM\_NGMLS\_VLAN\_AUDIT\_WK aud

set match\_code = ''NE Only''

where exists (select 1

from

ngMLS\_VLAN\_AUDIT\_ISSUEs\_wk vi,

ngmls\_issue n

where aud.MATCH\_CODE = ''BOTH''

and aud.ngMLS\_vendor = ''AL''

and VI.NGMLS\_VENDOR=''AL''

and AUD.NGMLS\_DEVICE\_NAME = VI.NGMLS\_DEVICE\_NAME

and AUD.NE\_VLAN\_NUMBER = VI.VLAN\_NUMBER

and N.IS\_CRITICAL=''Y''

and N.NGMLS\_ISSUE\_ID = vi.NGMLS\_ISSUE\_ID

)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant update match\_status to ''NE Only''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in sam vlan extract

sqlstmt :=

' insert into NGMLS\_VLAN\_AUDIT\_ISSUES\_WK

columns (ngMLS\_device\_name, vlan\_number, ngMLS\_issue\_id, ngMLS\_vendor)

WITH vlan\_status AS

(SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, status, match\_code,

nxa.vlan\_inst\_id

FROM sam\_ngmls\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

l\_and\_nl\_vlans AS

(SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status = ''Live''

INTERSECT

SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status <> ''Live''),

live\_vlan\_exists AS

(SELECT nxa.ngmls\_device\_name, nxa.xng\_vlan\_number, nxa.vlan\_inst\_id

FROM l\_and\_nl\_vlans dup JOIN vlan\_status nxa

ON nxa.ngmls\_device\_name = dup.ngmls\_device\_name

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

WHERE nxa.status <> ''Live'')

SELECT DISTINCT aud.ngmls\_device\_name, xng\_vlan\_number, 31, ''AL''

FROM sam\_ngmls\_vlan\_audit\_wk aud

WHERE vlan\_inst\_id IS NOT NULL

AND match\_code = ''NE Only''

AND NOT EXISTS (

SELECT 1

FROM ngmls\_vlan\_audit\_issues\_wk nvai

WHERE nvai.vlan\_number = aud.xng\_vlan\_number

AND aud.ngmls\_device\_name = nvai.ngmls\_device\_name

AND nvai.ngmls\_vendor = ''AL'')

AND EXISTS (

SELECT 1

FROM live\_vlan\_exists lle

WHERE lle.ngmls\_device\_name = aud.ngmls\_device\_name

AND aud.xng\_vlan\_number = lle.xng\_vlan\_number)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant insert issue ''Live VLAN Exists''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

END insert\_al\_ngmls\_vlan\_issues;

PROCEDURE load\_al\_vlan\_device\_summary

IS

methodname VARCHAR2 (30) := 'load\_AL\_vlan\_device\_summary';

MESSAGE VARCHAR2 (300);

sqlstmt VARCHAR2 (32767);

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

sqlstmt :=

'delete from SAM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK where ngMLS\_vendor = ''AL''';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

-- generate summary for CI devices ONLY

sqlstmt :=

' insert into SAM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK

columns (area, region, market, leaf\_domain\_inst\_id

, ngmls\_device\_name, ngMLS\_Vendor

, ne\_total\_ngmls\_vlan

, matched\_l\_ngmls\_vlan

, matched\_nl\_ngmls\_vlan

, mismatched\_ngmls\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_COMPLIANCE)

WITH

ngMLS as

(

select distinct area

, region, market, leaf\_domain\_inst\_id

, ngMLS\_device\_name, ngMLS\_vendor

from SAM\_NGMLS\_VLAN\_AUDIT\_WK ng

where ng.NGMLS\_DEVICE\_NAME is not null

),

ngMLS\_VLAN as

(

select distinct node\_name NGMLS\_DEVICE\_NAME, OUT\_ENCAP\_VAL vlan\_number

from SAM\_EBH\_RTR ng

where ng.OUT\_ENCAP\_VAL > 0

and DEVICE\_TYPE =''ngMLS''

)

,

CNT\_ngMLS\_VLANs as

(

select distinct ng.NGMLS\_DEVICE\_NAME, count(vlan\_number) total\_ne\_ngmls\_vlan

from ngMLS\_VLAN nv

join ngMLS ng

on ng.NGMLS\_DEVICE\_NAME = nv.NGMLS\_DEVICE\_NAME

group by ng.NGMLS\_DEVICE\_NAME

),

live\_match as

(

select distinct aud.NGMLS\_DEVICE\_NAME, aud.ne\_vlan\_number, vlan\_status

from SAM\_NGMLS\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.ngMLS\_vendor = ''AL''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status = ''Live''

)

,nonlive\_match as

(

select distinct aud.NGMLS\_DEVICE\_NAME, aud.ne\_vlan\_number

from SAM\_NGMLS\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.ngMLS\_vendor = ''AL''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status <> ''Live''

minus

select distinct aud.NGMLS\_DEVICE\_NAME, aud.ne\_vlan\_number

from SAM\_NGMLS\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.ngMLS\_vendor = ''AL''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status = ''Live''

) ,

CNT\_BOTH\_LIVE\_VLANs as

( -- good matches

select NGMLS\_DEVICE\_NAME, nvl (count(NGMLS\_DEVICE\_NAME), 0) matched\_l\_ngmls\_vlan

from live\_match aud

group by NGMLS\_DEVICE\_NAME

),

CNT\_BOTH\_NONLIVE\_VLANs as

( -- good matches

select NGMLS\_DEVICE\_NAME, nvl( count(NGMLS\_DEVICE\_NAME), 0) matched\_nl\_ngmls\_vlan

-- , nvl(sum(case when vlan\_status = ''Live'' then 1 else 0 end), 0) matched\_l\_ngmls\_vlan

-- , nvl(sum(case when vlan\_status <> ''Live'' then 1 else 0 end), 0) matched\_nl\_ngmls\_vlan

from nonlive\_match aud

group by aud.NGMLS\_DEVICE\_NAME

) ,

MISMATCH\_VLANs as

( -- bad matches

select distinct aud.NGMLS\_DEVICE\_NAME, aud.ne\_vlan\_number, aud.xng\_vlan\_number, vlan\_status

from SAM\_NGMLS\_VLAN\_AUDIT\_WK aud

where match\_code <> ''BOTH''

and aud.ngMLS\_vendor = ''AL''

and (aud.ne\_vlan\_number > 0 or aud.xng\_vlan\_number > 0)

)

,

CNT\_MISMATCH\_VLANs as

( -- bad match cnts

select aud.NGMLS\_DEVICE\_NAME

, count(1) mismatch\_cnt

from MISMATCH\_VLANs aud

join ngMLS ng

on ng.NGMLS\_DEVICE\_NAME = aud.NGMLS\_DEVICE\_NAME

group by aud.NGMLS\_DEVICE\_NAME

)

select area, region, market, leaf\_domain\_inst\_id

, ng.ngmls\_device\_name, ngMLS\_Vendor

, sum(nvl(nv.total\_ne\_ngmls\_vlan, 0)) ne\_total\_ngmls\_vlan

, sum(nvl(matched\_l\_ngmls\_vlan, 0)) matched\_l\_ngmls\_vlan

, sum(nvl(matched\_nl\_ngmls\_vlan, 0)) matched\_nl\_ngmls\_vlan

, sum(nvl(mismatch\_cnt, 0)) mismatched\_ngmls\_vlan

, round(sum(NVL(matched\_nl\_ngmls\_vlan, 0)) / decode(sum(NVL(matched\_l\_ngmls\_vlan, 0)), 0, 1, sum(nvl(matched\_l\_ngmls\_vlan, 0)))

\* 100, 2) gi\_nl\_2\_total\_vlan\_pct

, round (sum(nvl(matched\_l\_ngmls\_vlan,0)) / decode (sum(nvl(nv.total\_ne\_ngmls\_vlan, 0)) - sum(nvl(matched\_nl\_ngmls\_vlan, 0)), 0 , 1

, sum(nvl(nv.total\_ne\_ngmls\_vlan,0)) - sum(nvl(matched\_nl\_ngmls\_vlan,0)))

\* 100 , 2) vlan\_COMPLIANCE

from ngMLS ng

left outer join CNT\_ngMLS\_VLANs nv

on nv.NGMLS\_DEVICE\_NAME = ng.NGMLS\_DEVICE\_NAME

left outer join CNT\_BOTH\_LIVE\_VLANs lmv

on lmv.NGMLS\_DEVICE\_NAME = ng.NGMLS\_DEVICE\_NAME

left outer join CNT\_BOTH\_NONLIVE\_VLANs nlmv

on nlmv.NGMLS\_DEVICE\_NAME = ng.NGMLS\_DEVICE\_NAME

left outer join CNT\_MISMATCH\_VLANs mv

on mv.NGMLS\_DEVICE\_NAME = ng.NGMLS\_DEVICE\_NAME

group by ng.area, ng.region, ng.market, leaf\_domain\_inst\_id, ngMLS\_Vendor, ng.NGMLS\_DEVICE\_NAME

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname || '(): Can''t summarize AL devices';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END load\_al\_vlan\_device\_summary;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Procedure load\_vlan\_regional\_summary \*

\* \*

\* Purpose: From device Vlan summary Realize regional summary \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE load\_vlan\_regional\_summary

IS

methodname VARCHAR2 (30) := 'load\_vlan\_regional\_summary';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

EXECUTE IMMEDIATE 'truncate table SAM\_NGMLS\_VLAN\_REGION\_SUMM\_WK';

-- generate device summary

sqlstmt :=

' insert into SAM\_NGMLS\_VLAN\_REGION\_SUMM\_WK

columns (area, region

, ne\_total\_ngmls\_vlan

, matched\_l\_ngmls\_vlan

, matched\_nl\_ngmls\_vlan

, mismatched\_ngmls\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_compliance)

select dlr.area, dlr.region

, sum(ne\_total\_ngmls\_vlan) ne\_total\_ngmls\_vlan

, sum(matched\_l\_ngmls\_vlan) matched\_l\_ngmls\_vlan

, sum(matched\_nl\_ngmls\_vlan) matched\_nl\_ngmls\_vlan

, sum(mismatched\_ngmls\_vlan) mismatched\_ngmls\_vlan

, round(sum(NVL(matched\_nl\_ngmls\_vlan, 0)) / decode(sum(NVL(matched\_l\_ngmls\_vlan, 0)), 0, 1, sum(nvl(matched\_l\_ngmls\_vlan, 0)))

\* 100, 2) gi\_nl\_2\_total\_vlan\_pct

, round (sum(nvl(matched\_l\_ngmls\_vlan,0)) / decode (sum(nvl(ne\_total\_ngmls\_vlan, 0)) - sum(nvl(matched\_nl\_ngmls\_vlan, 0)), 0 , 1

, sum(nvl(ne\_total\_ngmls\_vlan,0)) - sum(nvl(matched\_nl\_ngmls\_vlan,0)))

\* 100 , 2) vlan\_compliance

from domains\_leaf\_reporting dlr

left outer join SAM\_NGMLS\_VLAN\_DEVICE\_SUMM\_WK ng

on dlr.market = ng.market

where dlr.AREA not in (''NNO'', ''OSS'')

group by rollup (dlr.area, dlr.region)

ORDER BY AREA, REGION ';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname || '(): Can''t generate regional summary';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END load\_vlan\_regional\_summary;

PROCEDURE assign\_area\_region\_ngmls\_vlan

IS

methodname VARCHAR2 (30) := 'assign\_area\_region\_ngmls\_vlan';

MESSAGE VARCHAR2 (300);

updstmt VARCHAR2 (32767);

processname VARCHAR2 (30) := 'SAM\_GI\_CSR\_AUDIT';

CURSOR area\_region\_dev

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS')

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

NGMLS\_device\_name

FROM SAM\_NGMLS\_VLAN\_AUDIT\_WK aud JOIN domains di

ON di.clli\_6 = SUBSTR (aud.ngmls\_device\_name, 1, 6)

;

CURSOR area\_region\_SAM

IS

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

di.ngmls\_device\_name

FROM SAM\_NGMLS\_VLAN\_AUDIT\_WK aud JOIN SAM\_GI\_AL\_NGMLS\_AUDIT\_WK di

ON di.ngmls\_device\_name = aud.ngmls\_device\_name

where aud.area is null;

BEGIN

updstmt :=

'update SAM\_NGMLS\_VLAN\_AUDIT\_WK set area = :area

, region = :region, market = :market

, leaf\_domain\_inst\_id = :leaf\_domain\_inst\_id

where ngmls\_device\_name = :ngmls\_device\_name';

FOR rec IN area\_region\_dev

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.ngmls\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area, market, region for '

|| rec.ngmls\_device\_name

|| ' in table: SAM\_NGMLS\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (methodname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

FOR rec IN area\_region\_SAM

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.ngmls\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area, market, region for '

|| rec.ngmls\_device\_name

|| ' in table: SAM\_NGMLS\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (methodname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

-- area region for put xng\_only based on the area/region of ne\_inst\_id???????????

BEGIN

-- if there still are some devices which cant be mapped to any domain then mark those as unknown

updstmt :=

' update SAM\_NGMLS\_VLAN\_AUDIT\_WK

set area = ''unknown''

, region = ''unknown''

, market = ''unknown''

, leaf\_domain\_inst\_id = 0

where AREA is null ';

EXECUTE IMMEDIATE updstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from to unknown in table: SAM\_NGMLS\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END assign\_area\_region\_ngmls\_vlan;

PROCEDURE audit\_al\_csr\_equip

IS

methodname VARCHAR2 (30) := 'audit\_AL\_csr\_equip';

processname VARCHAR2 (30) := 'SAM\_GI\_CSR\_AUDIT';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

-- verified with shirley --- assumption is they wont embed all csrs within a contianer of type csr. This logic will break if they so that.

-- does it matter if the the equipment name in GI does not match Std... verified with Shirley. yes it is fatal

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE 'truncate table SAM\_GI\_AL\_CSR\_AUDIT\_WK';

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t delete records from table: SAM\_GI\_AL\_CSR\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

BEGIN

DELETE FROM csr\_device\_audit\_issues\_wk iss

WHERE csr\_vendor = 'AL';

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t delete records from table: csr\_device\_audit\_issues\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

/\* audit csr devices from SAM vs Xng. Match them on device\_name \*/

BEGIN

sqlstmt :=

' insert into SAM\_GI\_AL\_CSR\_AUDIT\_WK

(csr\_vendor, csr\_device\_name

, sam\_csr\_device\_name, sam\_reachability, sam\_resync\_status, sam\_resync\_state, sam\_server\_name, sam\_last\_resync\_start, sam\_last\_resync\_end

, gi\_device\_name, equip\_inst\_id

, Live\_in\_xng, eq\_status

, match\_code

, match\_status

, eh\_csr\_hostname, eh\_csr\_ipaddress

, brix\_csr\_site

)

WITH

SAM\_csrs as

(

select distinct upper(SITE\_NAME) site\_name --, act\_mgmt\_ip sam\_csr\_device\_ip

, reachability sam\_reachability

, resync\_status sam\_resync\_status, resync\_state sam\_resync\_state

, server\_name sam\_server\_name, last\_resyn\_start sam\_last\_resync\_start, last\_resyn\_end sam\_last\_resync\_end

from SAM\_EBH\_HEALTH where device\_type = ''CSR''

)

,

Xng as

(

select csr.\*, ei.status status

from XNG\_CSR\_PARSED\_WK csr

join vzwnet.equip\_inst ei

on ei.EQUIP\_INST\_ID = csr.EQUIP\_INST\_ID

where csr.csr\_device\_name like ''%-AL-%''

)

,

EHEALTH\_CSR as

(

select upper(ecl.hostname) hostname,ecl.ip

from xng\_reports.ehealth\_csr\_list ecl

where hostname like ''%-AL-%'' -- put this is a separate column. then the query will be faster

)

,

BRIX\_CSR as

(

select upper(brx.cell\_site\_name) cell\_site\_name

from xng\_reports.BRIX\_CSR\_EXTRACT brx

where CELL\_SITE\_NAME like ''%-AL-%''

)

,

GI\_SAM\_AUD as

(

select distinct

case when sam.site\_name is not null then sam.site\_name

when xng.csr\_device\_name is not null then xng.csr\_device\_name

else ''zzzz''||xng.descr

end csr\_device\_name

, site\_name sam\_csr\_device\_name, sam\_reachability, sam\_resync\_status, sam\_resync\_state, sam\_server\_name, sam\_last\_resync\_start, sam\_last\_resync\_end

, xng.descr gi\_device\_name, equip\_inst\_id

, case when status = ''Live'' then ''Y''

when status is not null then ''N''

end Live\_in\_xng

, status

, case when sam.site\_name is not null

and xng.equip\_inst\_id is not null

then ''BOTH''

-- it is in both sam extracts but not in Xng then NE Only

when sam.site\_name is not null

and xng.equip\_inst\_id is null

then ''NE Only''

when sam.site\_name is null and xng.equip\_inst\_id is not null then ''Xng Only''

else ''XXXXX'' -- when does the control reach here

end match\_code

, parse\_status match\_status

from SAM\_csrs sam

full outer join Xng

on xng.csr\_device\_name = sam.site\_name

)

select ''AL'' csr\_vendor,

case when ecsr.hostname is not null then ecsr.hostname

when brx.cell\_site\_name is not null then brx.cell\_site\_name

when gsf.csr\_device\_name is not null then gsf.csr\_device\_name

end csr\_device\_name

, sam\_csr\_device\_name, sam\_reachability, sam\_resync\_status

, sam\_resync\_state, sam\_server\_name, sam\_last\_resync\_start, sam\_last\_resync\_end

, gi\_device\_name, equip\_inst\_id, Live\_in\_xng, status

-- it is in all 4 systems EH, SAM and GI, BRIX

, case when gsf.match\_code = ''BOTH'' and ecsr.hostname is not null and brx.cell\_site\_name is not null

then ''BOTH''

-- it is in gi and sam but not in EH then NE\_ONLY

when gsf.match\_code = ''BOTH'' and (ecsr.hostname is null or brx.cell\_site\_name is null)

then ''NE Only''

-- it is in gi and EH but not in SAM then NE\_ONLY

when gsf.match\_code = ''Xng Only'' and ecsr.hostname is not null

then ''NE Only''

else gsf.match\_code

end match\_code

, gsf.match\_status

, ecsr.hostname eh\_csr\_hostname, ecsr.ip eh\_csr\_ipaddress

, brx.cell\_site\_name

from GI\_SAM\_AUD gsf

full outer join ehealth\_csr ecsr

on gsf.csr\_device\_name = ecsr.hostname

full outer join BRIX\_CSR brx

on gsf.csr\_device\_name = brx.cell\_site\_name

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert insert AL csr audit into table: SAM\_GI\_AL\_CSR\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

BEGIN

-- if some ne\_hostname matched > 1 granite ne\_inst\_id then mark it dup

-- does stmt query work even when there are "Dup in NE"???

UPDATE sam\_gi\_al\_csr\_audit\_wk aud

SET match\_code = 'NE Only'

WHERE match\_code IS NULL;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t mark NE only for match code as blank';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

-- identify all issues with the audited equipment

insert\_al\_csr\_equip\_issues (processname);

END;

PROCEDURE insert\_al\_csr\_equip\_issues (processname VARCHAR2)

IS

methodname VARCHAR2 (30) := 'insert\_AL\_csr\_equip\_issues';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

insertSqlStart VARCHAR2(100) := 'insert into csr\_device\_audit\_issues\_wk columns (csr\_device\_name, csr\_issue\_id, csr\_vendor )';

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

-- CSR name in sam CSR Extract not Std Complaint --

insert\_csr\_issues (processname , methodname, insertSqlStart ||

'select distinct site\_name, 16, ''AL'' from sam\_ebh\_health aud where parse\_status like ''%16%''',

'Can''t insert into csr\_device\_audit\_issues\_wk issue 16 ');

-- CSR name in eHealth Extract not Std Complaint --

insert\_csr\_issues (processname , methodname, insertSqlStart ||

'select distinct hostname, 25, ''AL''

from ehealth\_csr\_list

where hostname like ''%-AL-%'' /\* CHANGE THIS TO CSR\_VENDOR \*/

and error\_code = 25 ',

'Can''t insert into csr\_device\_audit\_issues\_wk issue ehealth 25');

-- GI name not Std Complaint --

insert\_csr\_issues (processname , methodname, insertSqlStart ||

' select distinct csr\_device\_name, 7, ''AL''

from XNG\_CSR\_PARSED\_WK csr

where csr.csr\_device\_name like ''%-AL-%''

and parse\_status like ''%Xng CSR Name does not match Standard%''',

'Can''t insert into csr\_device\_audit\_issues\_wk issue issue 7');

insert\_csr\_issues (processname , methodname, insertSqlStart ||

'select distinct csr\_device\_name, 8, ''AL''

from XNG\_CSR\_PARSED\_WK csr

where csr\_device\_name like ''%-AL-%''

and parse\_status like ''%Invalid CSR Name%''',

'Can''t insert into csr\_device\_audit\_issues\_wk issue issue 8');

-- 12-14 issues in the previous are critical ONLY IF match\_code <> 'Xng Only'

-- the scheme needs to change somehow on the csr\_issues-------???????

-- device not found in sam csr extract

insert\_csr\_issues (processname , methodname, insertSqlStart ||

'select distinct csr\_device\_name, 12, ''AL''

from SAM\_GI\_AL\_CSR\_AUDIT\_WK aud

where SAM\_CSR\_DEVICE\_NAME is null',

'Can''t insert into csr\_device\_audit\_issues\_wk issue issue 12');

-- device not found in eHealth extract

insert\_csr\_issues (processname , methodname, insertSqlStart ||

'select distinct csr\_device\_name, 24, ''AL''

from SAM\_GI\_AL\_CSR\_AUDIT\_WK aud

where eh\_csr\_hostname is null',

'Can''t insert into csr\_device\_audit\_issues\_wk ehealth-issue 24');

-- device not found in eHealth extract

insert\_csr\_issues (processname , methodname, insertSqlStart ||

'select distinct csr\_device\_name, 33, ''AL''

from SAM\_GI\_AL\_CSR\_AUDIT\_WK aud

where brix\_csr\_site is null',

'Can''t insert into csr\_device\_audit\_issues\_wk ehealth-issue 33');

-- device not found in Xng

insert\_csr\_issues (processname , methodname, insertSqlStart ||

'select distinct csr\_device\_name, 15, ''AL''

from SAM\_GI\_AL\_CSR\_AUDIT\_WK aud

where equip\_inst\_id is null',

'Can''t insert into csr\_device\_audit\_issues\_wk issue 15');

-- DUP CSR match found in Xng

insert\_csr\_issues (processname , methodname, insertSqlStart ||

'select distinct csr\_device\_name, 1, ''AL''

from SAM\_GI\_AL\_CSR\_AUDIT\_WK i

where match\_code <> ''Xng Only''

group by i.csr\_device\_name

having count(equip\_inst\_id) > 1',

'Can''t insert into csr\_device\_audit\_issues\_wk issue 1');

-- DUPs found in NE. is this possible ????

-- mark 6-char match-not all- use from before

-- mark Invalid CLLI

-- We might have discovered some CLLI that is not in the list of MTSO CLLI

-- mark those as bad

-- We might have discovered some CLLI where 1st 6-char match MTSO CLLI but all 8 dont

-- mark those as bad

-- some times there are 2 legitimate cllis for the same site

-- in those cases the o/p comes as followd

-- clli\_1\_csr clli\_2\_vsm CLLI Mismatch

-- clli\_2\_csr clli\_1\_vsm CLLI Mismatch

-- so reverse the order and subtract it. That will give only the ones that are

-- truly bad

/\* want to be more specific to sam/sam VLAN and HPOV \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--sam issues 18 and 20

insert\_csr\_issues (processname , methodname, insertSqlStart ||

' WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.csr\_device\_name,

SUBSTR (ei.csr\_device\_name, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM SAM\_GI\_AL\_CSR\_AUDIT\_WK ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.csr\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.csr\_device\_name, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.csr\_device\_name

FROM SAM\_GI\_AL\_CSR\_AUDIT\_WK ei

MINUS

SELECT ei.csr\_device\_name

FROM SAM\_GI\_AL\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.csr\_device\_name, 1, 8)

MINUS

SELECT ei.csr\_device\_name

FROM SAM\_GI\_AL\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.csr\_device\_name, 1, 6)

),

ISSUES AS

(

SELECT cm.csr\_device\_name, 20 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.csr\_device\_name, 18 issue\_id

FROM bad\_clli bc

)

SELECT distinct WK.CSR\_DEVICE\_NAME , i.issue\_id, ''AL''

FROM ISSUES i, SAM\_GI\_AL\_CSR\_AUDIT\_WK wk

where i.csr\_device\_name = Wk.csr\_device\_name',

'Can''t insert into csr\_device\_audit\_issues\_wk issue 18 and 20');

--eHealth issues 26 and 27

insert\_csr\_issues (processname , methodname, insertSqlStart ||

' WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.eh\_csr\_hostname,

SUBSTR (ei.eh\_csr\_hostname, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM SAM\_GI\_AL\_CSR\_AUDIT\_WK ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.csr\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.eh\_csr\_hostname, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.eh\_csr\_hostname

FROM SAM\_GI\_AL\_CSR\_AUDIT\_WK ei

MINUS

SELECT ei.eh\_csr\_hostname

FROM SAM\_GI\_AL\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.eh\_csr\_hostname, 1, 8)

MINUS

SELECT ei.eh\_csr\_hostname

FROM SAM\_GI\_AL\_CSR\_AUDIT\_WK ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.eh\_csr\_hostname, 1, 6)

),

ISSUES AS

(

SELECT cm.eh\_csr\_hostname, 27 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.eh\_csr\_hostname, 26 issue\_id

FROM bad\_clli bc

)

SELECT distinct WK.CSR\_DEVICE\_NAME , i.issue\_id, ''AL''

FROM ISSUES i, SAM\_GI\_AL\_CSR\_AUDIT\_WK wk

where i.eh\_csr\_hostname = Wk.csr\_device\_name',

'Can''t insert into csr\_device\_audit\_issues\_wk issues 27 and 26');

BEGIN

-- if some ne\_hostname matched > 1 granite ne\_inst\_id then mark it dup

-- does stmt query work even when there are "Dup in NE"???

UPDATE sam\_gi\_al\_csr\_audit\_wk aud

SET match\_code = 'NE Only'

WHERE EXISTS (

SELECT 1

FROM csr\_device\_audit\_issues\_wk i

WHERE aud.csr\_device\_name = i.csr\_device\_name

AND i.csr\_vendor = 'AL'

GROUP BY i.csr\_device\_name)

AND match\_code = 'BOTH';

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update NE only if there is any issue';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END insert\_al\_csr\_equip\_issues;

PROCEDURE insert\_csr\_issues (processname VARCHAR2, methodname VARCHAR2, insertSql VARCHAR2, errorMsg VARCHAR2)

IS

MESSAGE VARCHAR2 (500);

BEGIN

BEGIN

EXECUTE IMMEDIATE insertSql;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE := 'Error: ' || methodname|| '(): ' || errorMsg;

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END insert\_csr\_issues;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

sam vs HPOV vs Xng, store summary by CLLI\_6

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE load\_csr\_clli\_summary

IS

processname VARCHAR2 (30) := 'SAM\_GI\_CSR\_AUDIT';

-- get the all CSRs found in sam CSR Extract

CURSOR csrs\_sam

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS'),

sam AS

(SELECT DISTINCT site\_name, SUBSTR (site\_name, 1, 6) clli\_6

FROM sam\_ebh\_health sam

WHERE device\_type = 'CSR')

SELECT NVL (area, 'unknown') area, NVL (region, 'unknown') region,

NVL (market, 'unknown') market,

NVL (di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id,

sam.clli\_6, COUNT (1) sam\_csrs

FROM domains di RIGHT OUTER JOIN sam ON sam.clli\_6 = di.clli\_6

GROUP BY area, region, market, di.leaf\_domain\_inst\_id, sam.clli\_6;

-- get the all CSRs found in EH

CURSOR csrs\_eh

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS'),

ehealth AS

(SELECT DISTINCT SUBSTR (hostname, 1, 6) clli\_6, hostname

FROM ehealth\_csr\_list hp

WHERE hostname LIKE '%-AL-%')

SELECT NVL (area, 'unknown') area, NVL (region, 'unknown') region,

NVL (market, 'unknown') market,

NVL (di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id,

eh.clli\_6, COUNT (1) eh\_csrs

FROM domains di RIGHT OUTER JOIN ehealth eh ON eh.clli\_6 =

di.clli\_6

GROUP BY area, region, market, di.leaf\_domain\_inst\_id, eh.clli\_6;

-- get the all CSRs found in BRIX

CURSOR csrs\_brix

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS'),

brix AS

(SELECT DISTINCT SUBSTR (cell\_site\_name, 1, 6) clli\_6, cell\_site\_name

FROM brix\_csr\_extract

WHERE cell\_site\_name LIKE '%-AL-%')

SELECT NVL (area, 'unknown') area, NVL (region, 'unknown') region,

NVL (market, 'unknown') market,

NVL (di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id,

b.clli\_6, COUNT (1) brix\_csrs

FROM domains di RIGHT OUTER JOIN brix b ON b.clli\_6 = di.clli\_6

GROUP BY area, region, market, di.leaf\_domain\_inst\_id, b.clli\_6;

-- ALL issues, critical or not

CURSOR csrs\_w\_issues

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS'),

csr\_w\_issues AS

(SELECT DISTINCT cai.csr\_device\_name,

SUBSTR (cai.csr\_device\_name, 1, 6) clli\_6

FROM csr\_device\_audit\_issues\_wk cai

WHERE cai.csr\_vendor = 'AL')

SELECT NVL (area, 'unknown') area, NVL (region, 'unknown') region,

NVL (market, 'unknown') market,

NVL (di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id,

sam.clli\_6, COUNT (1) sam\_eh\_gi\_issues

FROM domains di RIGHT OUTER JOIN csr\_w\_issues sam

ON sam.clli\_6 = di.clli\_6

GROUP BY area, region, market, di.leaf\_domain\_inst\_id, sam.clli\_6

ORDER BY area, region;

-- make sure duplicate matches in GI are not considered as good match-------??????

-- DUP means > 1 Live ones in Xng

-- if 1 is live and others are not, then that is NOT a DIP. Shirley's logic

-- this should be identifiable from the audit

-- get mismatches between sam-csr-vlan extract and gi

CURSOR sam\_gi\_match\_csrs

IS

WITH sam\_gi\_match AS

(SELECT csr\_device\_name,

CASE

WHEN live\_in\_xng = 'Y'

THEN 1

ELSE 0

END sam\_gi\_l\_match,

CASE

WHEN live\_in\_xng <> 'Y'

THEN 1

ELSE 0

END sam\_gi\_nl\_match

FROM sam\_gi\_al\_csr\_audit\_wk aud

WHERE aud.sam\_csr\_device\_name IS NOT NULL

AND aud.gi\_device\_name IS NOT NULL

MINUS

-- DUPS are NOT good

SELECT aud.csr\_device\_name,

CASE

WHEN live\_in\_xng = 'Y'

THEN 1

ELSE 0

END sam\_gi\_l\_match,

CASE

WHEN live\_in\_xng <> 'Y'

THEN 1

ELSE 0

END sam\_gi\_nl\_match

FROM sam\_gi\_al\_csr\_audit\_wk aud JOIN csr\_device\_audit\_issues\_wk cdai

ON cdai.csr\_device\_name = aud.csr\_device\_name

WHERE csr\_issue\_id = 1 AND match\_code <> 'Xng Only')

SELECT CASE

WHEN ma.csr\_device\_name IS NOT NULL

THEN SUBSTR (ma.csr\_device\_name, 1, 6)

END clli\_6,

COUNT (ma.csr\_device\_name) sam\_gi\_match,

SUM (ma.sam\_gi\_l\_match) sam\_gi\_l\_match,

SUM (ma.sam\_gi\_nl\_match) sam\_gi\_nl\_match

FROM sam\_gi\_match ma

GROUP BY CASE

WHEN ma.csr\_device\_name IS NOT NULL

THEN SUBSTR (ma.csr\_device\_name, 1, 6)

END;

-- make sure duplicate matches in GI are not considered as good match-------??????

-- DUP means > 1 Live ones in Xng

-- if 1 is live and others are not, then that is NOT a DUP. Shirley's logic

-- this should be identifiable from the audit

CURSOR eh\_gi\_match\_csrs

IS

WITH eh\_gi\_match AS

(SELECT csr\_device\_name --, match\_status, live\_in\_xng

,

CASE

WHEN live\_in\_xng = 'Y'

THEN 1

ELSE 0

END eh\_gi\_l\_match,

CASE

WHEN live\_in\_xng <> 'Y'

THEN 1

ELSE 0

END eh\_gi\_nl\_match

FROM sam\_gi\_al\_csr\_audit\_wk aud

WHERE aud.eh\_csr\_hostname IS NOT NULL

AND aud.gi\_device\_name IS NOT NULL

MINUS

-- DUPS are NOT good

SELECT aud.csr\_device\_name,

CASE

WHEN live\_in\_xng = 'Y'

THEN 1

ELSE 0

END sam\_gi\_l\_match,

CASE

WHEN live\_in\_xng <> 'Y'

THEN 1

ELSE 0

END sam\_gi\_nl\_match

FROM sam\_gi\_al\_csr\_audit\_wk aud JOIN csr\_device\_audit\_issues\_wk cdai

ON cdai.csr\_device\_name = aud.csr\_device\_name

WHERE csr\_issue\_id = 1 AND match\_code <> 'Xng Only')

SELECT CASE

WHEN ma.csr\_device\_name IS NOT NULL

THEN SUBSTR (ma.csr\_device\_name, 1, 6)

END clli\_6,

COUNT (ma.csr\_device\_name) eh\_gi\_match,

SUM (ma.eh\_gi\_l\_match) eh\_gi\_l\_match,

SUM (ma.eh\_gi\_nl\_match) eh\_gi\_nl\_match

FROM eh\_gi\_match ma

GROUP BY CASE

WHEN ma.csr\_device\_name IS NOT NULL

THEN SUBSTR (ma.csr\_device\_name, 1, 6)

END;

-- make sure duplicate matches in GI are not considered as good match-------??????

-- DUP means > 1 Live ones in Xng

-- if 1 is live and others are not, then that is NOT a DUP. Shirley's logic

-- this should be identifiable from the audit

CURSOR brix\_gi\_match\_csrs

IS

WITH brix\_gi\_match AS

(SELECT csr\_device\_name --, match\_status, live\_in\_xng

,

CASE

WHEN live\_in\_xng = 'Y'

THEN 1

ELSE 0

END brix\_gi\_l\_match,

CASE

WHEN live\_in\_xng <> 'Y'

THEN 1

ELSE 0

END brix\_gi\_nl\_match

FROM sam\_gi\_al\_csr\_audit\_wk aud

WHERE aud.brix\_csr\_site IS NOT NULL

AND aud.gi\_device\_name IS NOT NULL

MINUS

-- DUPS are NOT good

SELECT aud.csr\_device\_name,

CASE

WHEN live\_in\_xng = 'Y'

THEN 1

ELSE 0

END brix\_gi\_l\_match,

CASE

WHEN live\_in\_xng <> 'Y'

THEN 1

ELSE 0

END brix\_gi\_nl\_match

FROM sam\_gi\_al\_csr\_audit\_wk aud JOIN csr\_device\_audit\_issues\_wk cdai

ON cdai.csr\_device\_name = aud.csr\_device\_name

WHERE csr\_issue\_id = 1 AND match\_code <> 'Xng Only')

SELECT CASE

WHEN ma.csr\_device\_name IS NOT NULL

THEN SUBSTR (ma.csr\_device\_name, 1, 6)

END clli\_6,

COUNT (ma.csr\_device\_name) brix\_gi\_match,

SUM (ma.brix\_gi\_l\_match) brix\_gi\_l\_match,

SUM (ma.brix\_gi\_nl\_match) brix\_gi\_nl\_match

FROM brix\_gi\_match ma

GROUP BY CASE

WHEN ma.csr\_device\_name IS NOT NULL

THEN SUBSTR (ma.csr\_device\_name, 1, 6)

END;

CURSOR sam\_gi\_mis\_match\_csrs

IS

WITH sam\_gi\_mis\_match AS

(SELECT DISTINCT SUBSTR (csr\_device\_name, 1, 6) clli\_6, csr\_device\_name , match\_code, live\_in\_xng

FROM sam\_gi\_al\_csr\_audit\_wk aud

WHERE (( aud.sam\_csr\_device\_name IS NULL

AND aud.gi\_device\_name IS NOT NULL

)

OR ( aud.gi\_device\_name IS NULL

AND aud.sam\_csr\_device\_name IS NOT NULL

))

AND match\_code <> 'BOTH'

)

select clli\_6, sum(cc) sam\_gi\_mis\_match from (SELECT CASE

WHEN csr\_device\_name IS NOT NULL and match\_code = 'NE Only' or csr\_device\_name IS NOT NULL and live\_in\_xng = 'Y'

THEN 1 else 0

END cc,

csr\_device\_name, clli\_6

FROM sam\_gi\_mis\_match )

GROUP BY clli\_6;

CURSOR eh\_gi\_mis\_match\_csrs

IS

WITH eh\_gi\_mis\_match AS

(SELECT DISTINCT SUBSTR (csr\_device\_name, 1, 6) clli\_6, csr\_device\_name , match\_code, live\_in\_xng

FROM sam\_gi\_al\_csr\_audit\_wk aud

WHERE (( aud.eh\_csr\_hostname IS NULL

AND aud.gi\_device\_name IS NOT NULL

)

OR ( aud.gi\_device\_name IS NULL

AND aud.eh\_csr\_hostname IS NOT NULL

))

AND match\_code <> 'BOTH'

)

select clli\_6, sum(cc) eh\_gi\_mis\_match from (SELECT CASE

WHEN csr\_device\_name IS NOT NULL and match\_code = 'NE Only' or csr\_device\_name IS NOT NULL and live\_in\_xng = 'Y'

THEN 1 else 0

END cc,

csr\_device\_name, clli\_6

FROM eh\_gi\_mis\_match )

GROUP BY clli\_6;

CURSOR brix\_gi\_mis\_match\_csrs

IS

WITH brix\_gi\_mis\_match AS

(SELECT DISTINCT SUBSTR (csr\_device\_name, 1, 6) clli\_6, csr\_device\_name , match\_code, live\_in\_xng

FROM sam\_gi\_al\_csr\_audit\_wk aud

WHERE (( aud.brix\_csr\_site IS NULL

AND aud.gi\_device\_name IS NOT NULL

)

OR ( aud.gi\_device\_name IS NULL

AND aud.brix\_csr\_site IS NOT NULL

))

AND match\_code <> 'BOTH'

)

select clli\_6, sum(cc) brix\_gi\_mis\_match from (SELECT CASE

WHEN csr\_device\_name IS NOT NULL and match\_code = 'NE Only' or csr\_device\_name IS NOT NULL and live\_in\_xng = 'Y'

THEN 1 else 0

END cc,

csr\_device\_name, clli\_6

FROM brix\_gi\_mis\_match )

GROUP BY clli\_6;

methodname VARCHAR2 (30) := 'load\_csr\_clli\_summary';

MESSAGE VARCHAR2 (300);

updstmt VARCHAR2 (32767);

insstmt VARCHAR2 (32767);

sqlstmt VARCHAR2 (32767);

l\_cnt NUMBER;

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

DELETE sam\_eh\_gi\_csr\_clli\_summ\_wk;

-- from xng insert all CLLI and xng# in the table

sqlstmt :=

'insert into sam\_eh\_gi\_csr\_clli\_summ\_wk columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, gi\_csrs, gi\_l\_csrs, gi\_nl\_csrs)

WITH

DOMAINS AS

(

select distinct substr(clli, 1, 6) clli\_6, area, region, market, leaf\_domain\_inst\_id

from clli\_domain\_map\_v

where region <> ''NNO''

and region <> ''OSS''

)

,

-- get all devices from Xng along with live and NL status

GI as

(

select distinct csr\_device\_name, substr(csr\_device\_name, 1, 6) clli\_6

, nvl(case when ei.status = ''Live'' then 1 else 0 end, 0) l

, nvl(case when ei.status <> ''Live'' then 1 else 0 end, 0) nl

from XNG\_CSR\_PARSED\_WK csr join vzwnet.equip\_inst ei

on ei.EQUIP\_INST\_ID = csr.EQUIP\_INST\_ID

where csr.csr\_device\_name like ''%-AL-%''

)

,

GI\_CLLI\_DOMAIN as

(

-- determine the xng device domain based on CLLI

select nvl(di.area, ''unknown'') area, nvl(di.region, ''unknown'') region, nvl(di.market, ''unknown'') market

, nvl(di.leaf\_domain\_inst\_id, 0) leaf\_domain\_inst\_id

, xng.clli\_6

, xng.csr\_device\_name

, xng.l

, xng.nl

from DOMAINS di

join GI xng

on xng.clli\_6 = di.clli\_6

)

,

GI\_EQ\_NO\_CLLI\_DOMAIN as

(

-- get all devices for which domain cant be found based on CLLI

select xng.clli\_6

, xng.csr\_device\_name

, l

, nl

from GI xng

minus

-- determine the device domain based on CLLI

select xng.clli\_6

, xng.csr\_device\_name

, l

, nl

from GI\_CLLI\_DOMAIN xng

)

,

GI\_EQ\_W\_GI\_DOMAIN as

(

-- for those devices get the domain from Xng

select nvl(area, ''unknown'') area, nvl(region, ''unknown'') region, nvl(market, ''unknown'') market

, nvl(dlr.domain\_inst\_id, 0) leaf\_domain\_inst\_id

, xng.clli\_6

, xng.csr\_device\_name

, l

, nl

from GI\_EQ\_NO\_CLLI\_DOMAIN xng

join xng\_csr\_parsed\_WK xcp

on xcp.CSR\_DEVICE\_NAME = xng.csr\_device\_name

join vzwnet.equip\_domain\_map edm

on xcp.equip\_inst\_id = edm.equip\_inst\_id

join domains\_leaf\_reporting dlr

on dlr.domain\_inst\_id = edm.domain\_inst\_id

)

,

ALL\_GI\_EQUIP as

(

select area, region, market, leaf\_domain\_inst\_id

, xng.clli\_6

, xng.csr\_device\_name

, l

, nl

from GI\_CLLI\_DOMAIN xng

union all

select area, region, market, leaf\_domain\_inst\_id

, xng.clli\_6

, xng.csr\_device\_name

, l

, nl

from GI\_EQ\_W\_GI\_DOMAIN xng

)

-- determine the device domain based on CLLI

select gi.area, gi.region, gi.market, gi.leaf\_domain\_inst\_id

, gi.clli\_6

, count (gi.csr\_device\_name) gi\_csrs

, sum (l) gi\_l\_csr

, sum (nl) gi\_nl\_csr

from ALL\_GI\_EQUIP gi

group by gi.area, gi.region, gi.market, gi.leaf\_domain\_inst\_id

, gi.clli\_6

order by gi.area, gi.region, gi.market, gi.leaf\_domain\_inst\_id

, gi.clli\_6

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

-- insert new or update metric in the same table for total\_csrs found in sam in the csr-extract

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set sam\_csrs = :sam\_csrs

where clli\_6 = :clli\_6

';

insstmt :=

'insert into sam\_eh\_gi\_csr\_clli\_summ\_wk columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, sam\_csrs)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :sam\_csrs)

';

FOR rec IN csrs\_sam

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.sam\_csrs, rec.clli\_6;

l\_cnt := SQL%ROWCOUNT;

IF (l\_cnt = 0)

THEN

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

DBMS\_OUTPUT.put\_line ( 'FYI: '

|| l\_cnt

|| ' matching rows for sam CLLI'

);

DBMS\_OUTPUT.put\_line ( 'FYI: should insert in sam '

|| rec.area

|| ' '

|| rec.region

|| ' '

|| rec.market

|| ' '

|| rec.leaf\_domain\_inst\_id

|| ' '

|| rec.clli\_6

|| ' '

|| rec.sam\_csrs

);

BEGIN

EXECUTE IMMEDIATE insstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli\_6,

rec.sam\_csrs;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert '

|| rec.clli\_6

|| ' sam only CSR extract CLLI in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

--RAISE;

END;

END IF;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update '

|| rec.clli\_6

|| ' for sam-CSR-extract in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

-- insert new or update metric in the same table for ehealth total\_csrs

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set eh\_csrs = :eh\_csrs

where clli\_6 = :clli\_6

';

insstmt :=

'insert into sam\_eh\_gi\_csr\_clli\_summ\_wk columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, eh\_csrs)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :eh\_csrs)

';

FOR rec IN csrs\_eh

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.eh\_csrs, rec.clli\_6;

l\_cnt := SQL%ROWCOUNT;

IF (l\_cnt = 0)

THEN

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

DBMS\_OUTPUT.put\_line ( 'FYI: '

|| l\_cnt

|| ' matching rows for sam CLLI'

);

DBMS\_OUTPUT.put\_line ( 'FYI: should insert in ehealth '

|| rec.area

|| ' '

|| rec.region

|| ' '

|| rec.market

|| ' '

|| rec.leaf\_domain\_inst\_id

|| ' '

|| rec.clli\_6

|| ' '

|| rec.eh\_csrs

);

BEGIN

EXECUTE IMMEDIATE insstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli\_6,

rec.eh\_csrs;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert '

|| rec.clli\_6

|| ' sam-CSR-VLAN-extract CLLI in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

--RAISE;

END;

END IF;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update '

|| rec.clli\_6

|| ' for sam-CSR-VLAN-extract in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

-- insert new or update metric in the same table for brix total\_csrs

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set brix\_csrs = :brix\_csrs

where clli\_6 = :clli\_6

';

insstmt :=

'insert into sam\_eh\_gi\_csr\_clli\_summ\_wk columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, brix\_csrs)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :brix\_csrs)

';

FOR rec IN csrs\_brix

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.brix\_csrs, rec.clli\_6;

l\_cnt := SQL%ROWCOUNT;

IF (l\_cnt = 0)

THEN

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

DBMS\_OUTPUT.put\_line ( 'FYI: '

|| l\_cnt

|| ' matching rows for sam CLLI'

);

DBMS\_OUTPUT.put\_line ( 'FYI: should insert in brix '

|| rec.area

|| ' '

|| rec.region

|| ' '

|| rec.market

|| ' '

|| rec.leaf\_domain\_inst\_id

|| ' '

|| rec.clli\_6

|| ' '

|| rec.brix\_csrs

);

BEGIN

EXECUTE IMMEDIATE insstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli\_6,

rec.brix\_csrs;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert '

|| rec.clli\_6

|| ' sam-CSR-VLAN-extract CLLI in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

--RAISE;

END;

END IF;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update '

|| rec.clli\_6

|| ' for sam-CSR-VLAN-extract in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

-- insert new or update metric in the same table for sam-ehealth-gi csrs w issues

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set sam\_eh\_gi\_issues = :sam\_eh\_gi\_issues

where clli\_6 = :clli\_6

';

insstmt :=

'insert into sam\_eh\_gi\_csr\_clli\_summ\_wk columns (area, region, market, leaf\_domain\_inst\_id, clli\_6, sam\_eh\_gi\_issues)

values (:area, :region, :market, :leaf\_domain\_inst\_id, :clli\_6, :sam\_eh\_gi\_issues)

';

FOR rec IN csrs\_w\_issues

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.sam\_eh\_gi\_issues, rec.clli\_6;

l\_cnt := SQL%ROWCOUNT;

IF (l\_cnt = 0)

THEN

-- there is a possibility that all CLLIs did not make into the table yet.

-- if it did not then insert the clli and the metric

DBMS\_OUTPUT.put\_line ( 'FYI: '

|| l\_cnt

|| ' matching rows for sam CLLI'

);

DBMS\_OUTPUT.put\_line ( 'FYI: should insert in sam '

|| rec.area

|| ' '

|| rec.region

|| ' '

|| rec.market

|| ' '

|| rec.leaf\_domain\_inst\_id

|| ' '

|| rec.clli\_6

|| ' '

|| rec.sam\_eh\_gi\_issues

);

BEGIN

EXECUTE IMMEDIATE insstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli\_6,

rec.sam\_eh\_gi\_issues;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert '

|| rec.clli\_6

|| ' sam-ehealth-gi-CSRs-W-ISSUES CLLI in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

--RAISE;

END;

END IF;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update '

|| rec.clli\_6

|| ' for sam-eh-gi-CSR-issues in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

-- calc matches / mismatches metric between sam and gi

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set sam\_gi\_match = :sam\_gi\_match

, sam\_gi\_l\_match = :sam\_gi\_l\_match

, sam\_gi\_nl\_match = :sam\_gi\_nl\_match

where clli\_6 = :clli\_6

';

FOR rec IN sam\_gi\_match\_csrs

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.sam\_gi\_match,

rec.sam\_gi\_l\_match,

rec.sam\_gi\_nl\_match,

rec.clli\_6;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update sam-GI mismatch in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

-- RAISE;

END;

END LOOP;

COMMIT;

-- calc matches / mismatches metric between hpov and gi

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set eh\_gi\_match = :eh\_gi\_match

, eh\_gi\_l\_match = :eh\_gi\_l\_match

, eh\_gi\_nl\_match = :eh\_gi\_nl\_match

where clli\_6 = :clli\_6

';

FOR rec IN eh\_gi\_match\_csrs

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.eh\_gi\_match,

rec.eh\_gi\_l\_match,

rec.eh\_gi\_nl\_match,

rec.clli\_6;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update EH match in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

-- RAISE;

END;

END LOOP;

-- calc matches / mismatches metric between brix and gi

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set brix\_gi\_match = :brix\_gi\_match

, brix\_gi\_l\_match = :brix\_gi\_l\_match

, brix\_gi\_nl\_match = :brix\_gi\_nl\_match

where clli\_6 = :clli\_6';

FOR rec IN brix\_gi\_match\_csrs

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.brix\_gi\_match,

rec.brix\_gi\_l\_match,

rec.brix\_gi\_nl\_match,

rec.clli\_6;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update BRIX match in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

-- RAISE;

END;

END LOOP;

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set sam\_gi\_mis\_match = :sam\_gi\_mis\_match

where clli\_6 = :clli\_6

';

FOR rec IN sam\_gi\_mis\_match\_csrs

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.sam\_gi\_mis\_match,

rec.clli\_6;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update sam-GI mismatch in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

-- RAISE;

END;

END LOOP;

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set eh\_gi\_mis\_match = :eh\_gi\_mis\_match

where clli\_6 = :clli\_6

';

FOR rec IN eh\_gi\_mis\_match\_csrs

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.eh\_gi\_mis\_match,

rec.clli\_6;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update EH mismatch in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

-- RAISE;

END;

END LOOP;

updstmt :=

'update sam\_eh\_gi\_csr\_clli\_summ\_wk

set brix\_gi\_mis\_match = :brix\_gi\_mis\_match

where clli\_6 = :clli\_6

';

FOR rec IN brix\_gi\_mis\_match\_csrs

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.brix\_gi\_mis\_match,

rec.clli\_6;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update BRIX mismatch in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

-- RAISE;

END;

END LOOP;

-- calculate the percentages

BEGIN

UPDATE sam\_eh\_gi\_csr\_clli\_summ\_wk

SET eh\_gi\_per =

ROUND ( NVL (eh\_gi\_l\_match, 0)

\* 2

/ DECODE ((NVL (eh\_csrs, 0) + NVL (gi\_l\_csrs, 0)),

0, 1,

NVL (eh\_csrs, 0) + NVL (gi\_l\_csrs, 0)

)

\* 100,

2

),

brix\_gi\_per =

ROUND ( NVL (brix\_gi\_l\_match, 0)

\* 2

/ DECODE ((NVL (brix\_csrs, 0) + NVL (gi\_l\_csrs, 0)),

0, 1,

NVL (brix\_csrs, 0) + NVL (gi\_l\_csrs, 0)

)

\* 100,

2

),

sam\_gi\_per =

ROUND ( NVL (sam\_gi\_l\_match, 0)

\* 2

/ DECODE ((NVL (sam\_csrs, 0) + NVL (gi\_l\_csrs, 0)),

0, 1,

NVL (sam\_csrs, 0) + NVL (gi\_l\_csrs, 0)

)

\* 100,

2

);

-- for rec in (select \* from sam\_eh\_gi\_csr\_clli\_summ\_wk)

-- loop

-- dbms\_output.put\_line('round(nvl('||rec.eh\_gi\_l\_match||', 0) \* 2 / decode ((nvl('||rec.eh\_csrs||',0) + nvl('||rec.gi\_l\_csrs||',0)), 0, 1, nvl('||rec.eh\_csrs||',0) + nvl('||rec.gi\_l\_csrs||',0)) \* 100, 2)');

-- dbms\_output.put\_line('round(nvl('||rec.sam\_gi\_l\_match||', 0) \* 2 / decode ((nvl('||rec.sam\_csrs||',0) + nvl('||rec.gi\_l\_csrs||',0)), 0, 1, nvl('||rec.sam\_csrs||',0) + nvl('||rec.gi\_l\_csrs||',0)) \* 100, 2)');

-- end loop;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update percentages in table: sam\_eh\_gi\_csr\_clli\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

-- RAISE;

END;

END load\_csr\_clli\_summary;

PROCEDURE load\_csr\_regional\_summary

IS

methodname VARCHAR2 (30) := 'load\_csr\_regional\_summary';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

err VARCHAR2 (250);

processname VARCHAR2 (30) := 'SAM\_GI\_CSR\_AUDIT';

BEGIN

BEGIN

-- calculate the percentages

sqlstmt := 'truncate table sam\_eh\_gi\_csr\_reg\_summ\_wk';

EXECUTE IMMEDIATE sqlstmt;

-- rollup % from CLLI

INSERT INTO sam\_eh\_gi\_csr\_reg\_summ\_wk COLUMNS

(area, region, gi\_csrs, gi\_l\_csrs, gi\_nl\_csrs, eh\_csrs,

eh\_gi\_match, eh\_gi\_l\_match, eh\_gi\_nl\_match,

sam\_csrs, sam\_gi\_match, sam\_gi\_l\_match, sam\_gi\_nl\_match,

sam\_eh\_gi\_issues,sam\_gi\_mis\_match,eh\_gi\_mis\_match,

brix\_csrs, brix\_gi\_match, brix\_gi\_l\_match, brix\_gi\_nl\_match, brix\_gi\_mis\_match)

SELECT area, region, SUM (NVL (gi\_csrs, 0)) gi\_csrs,

SUM (NVL (gi\_l\_csrs, 0)) gi\_l\_csrs,

SUM (NVL (gi\_nl\_csrs, 0)) gi\_nl\_csrs,

SUM (NVL (eh\_csrs, 0)) eh\_csrs,

SUM (NVL (eh\_gi\_match, 0)) eh\_gi\_match,

SUM (NVL (eh\_gi\_l\_match, 0)) eh\_gi\_l\_match,

SUM (NVL (eh\_gi\_nl\_match, 0)) eh\_gi\_nl\_match,

SUM (NVL (sam\_csrs, 0)) sam\_csrs,

SUM (NVL (sam\_gi\_match, 0)) sam\_gi\_match,

SUM (NVL (sam\_gi\_l\_match, 0)) sam\_gi\_l\_match,

SUM (NVL (sam\_gi\_nl\_match, 0)) sam\_gi\_nl\_match,

SUM (NVL (sam\_eh\_gi\_issues, 0)) sam\_eh\_gi\_issues,

SUM (NVL (sam\_gi\_mis\_match, 0)) sam\_gi\_mis\_match,

SUM (NVL (eh\_gi\_mis\_match, 0)) eh\_gi\_mis\_match,

SUM (NVL (brix\_csrs, 0)) brix\_csrs,

SUM (NVL (brix\_gi\_match, 0)) brix\_gi\_match,

SUM (NVL (brix\_gi\_l\_match, 0)) brix\_gi\_l\_match,

SUM (NVL (brix\_gi\_nl\_match, 0)) brix\_gi\_nl\_match,

SUM (NVL (brix\_gi\_mis\_match, 0)) brix\_gi\_mis\_match

FROM sam\_eh\_gi\_csr\_clli\_summ\_wk

GROUP BY ROLLUP (area, region)

ORDER BY area, region;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t rollup percentages for the region in table: sam\_eh\_gi\_csr\_reg\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

END;

BEGIN

UPDATE sam\_eh\_gi\_csr\_reg\_summ\_wk

SET eh\_gi\_per =

ROUND ( NVL (eh\_gi\_l\_match, 0)

\* 2

/ DECODE ((NVL (eh\_csrs, 0) + NVL (gi\_l\_csrs, 0)),

0, 1,

NVL (eh\_csrs, 0) + NVL (gi\_l\_csrs, 0)

)

\* 100,

2

),

sam\_gi\_per =

ROUND ( NVL (sam\_gi\_l\_match, 0)

\* 2

/ DECODE ((NVL (sam\_csrs, 0) + NVL (gi\_l\_csrs, 0)),

0, 1,

NVL (sam\_csrs, 0) + NVL (gi\_l\_csrs, 0)

)

\* 100,

2

),

brix\_gi\_per =

ROUND ( NVL (brix\_gi\_l\_match, 0)

\* 2

/ DECODE ((NVL (brix\_csrs, 0) + NVL (gi\_l\_csrs, 0)),

0, 1,

NVL (brix\_csrs, 0) + NVL (gi\_l\_csrs, 0)

)

\* 100,

2

);

-- for rec in (select \* from sam\_eh\_gi\_csr\_clli\_summ\_wk)

-- loop

-- dbms\_output.put\_line('round(nvl('||rec.eh\_gi\_l\_match||', 0) \* 2 / decode ((nvl('||rec.eh\_csrs||',0) + nvl('||rec.gi\_l\_csrs||',0)), 0, 1, nvl('||rec.eh\_csrs||',0) + nvl('||rec.gi\_l\_csrs||',0)) \* 100, 2)');

-- dbms\_output.put\_line('round(nvl('||rec.sam\_gi\_l\_match||', 0) \* 2 / decode ((nvl('||rec.sam\_csrs||',0) + nvl('||rec.gi\_l\_csrs||',0)), 0, 1, nvl('||rec.sam\_csrs||',0) + nvl('||rec.gi\_l\_csrs||',0)) \* 100, 2)');

-- end loop;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update percentages in table: sam\_eh\_gi\_csr\_reg\_summ\_wk';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

-- RAISE;

END;

-- RAISE;

END load\_csr\_regional\_summary;

PROCEDURE assign\_area\_region\_to\_csr

IS

methodname VARCHAR2 (30) := 'assign\_area\_region\_to\_CSR';

MESSAGE VARCHAR2 (300);

updstmt VARCHAR2 (32767);

processname VARCHAR2 (30) := 'SAM\_GI\_CSR\_AUDIT';

CURSOR area\_region\_dev

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS')

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

csr\_device\_name, clli\_6

FROM sam\_gi\_al\_csr\_audit\_wk aud JOIN domains di

ON di.clli\_6 = SUBSTR (aud.csr\_device\_name, 1, 6)

;

CURSOR eq\_area\_region\_dev

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS')

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

csr\_device\_name, clli\_6

FROM sam\_gi\_al\_csr\_audit\_wk aud JOIN vzwnet.equip\_domain\_map edm

ON edm.equip\_inst\_id = aud.equip\_inst\_id

JOIN domains di ON di.leaf\_domain\_inst\_id = edm.domain\_inst\_id

WHERE aud.area IS NULL;

CURSOR unknown\_domain

IS

SELECT dlr.area AS area, dlr.region AS region, dlr.market AS market,

dlr.domain\_inst\_id AS domain\_inst\_id,

sgalwk.csr\_device\_name AS csr\_device\_name

FROM vzwnet.equip\_inst equip,

sam\_gi\_al\_csr\_audit\_wk sgalwk,

vzwnet.equip\_domain\_map equipmap,

xng\_reports.domains\_leaf\_reporting dlr

WHERE sgalwk.leaf\_domain\_inst\_id = 0

AND equip.descr = sgalwk.sam\_csr\_device\_name

AND equipmap.equip\_inst\_id = equip.equip\_inst\_id

AND dlr.domain\_inst\_id = equipmap.domain\_inst\_id;

BEGIN

updstmt :=

'update SAM\_GI\_AL\_CSR\_AUDIT\_WK set area = :area

, region = :region, market = :market

, leaf\_domain\_inst\_id = :leaf\_domain\_inst\_id

, clli = :clli\_6

where csr\_device\_name = :csr\_device\_name';

FOR rec IN area\_region\_dev

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli\_6,

rec.csr\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region for '

|| rec.csr\_device\_name

|| ' in table: SAM\_GI\_AL\_CSR\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (methodname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

-- area region for put xng\_only based on the area/region of ne\_inst\_id???????????

FOR rec IN eq\_area\_region\_dev

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli\_6,

rec.csr\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from Xng Only for '

|| rec.csr\_device\_name

|| ' in table: SAM\_GI\_AL\_CSR\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

BEGIN

-- if there still are some devices which cant be mapped to any domain then mark those as unknown

updstmt :=

' update SAM\_GI\_AL\_CSR\_AUDIT\_WK

set area = ''unknown''

, region = ''unknown''

, market = ''unknown''

, leaf\_domain\_inst\_id = 0

, clli = ''unknown''

where area is null ';

EXECUTE IMMEDIATE updstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from to unknown in table: SAM\_GI\_AL\_CSR\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

updstmt :=

' update SAM\_GI\_AL\_CSR\_AUDIT\_WK

set area = :area

, region = :region, market = :market

, leaf\_domain\_inst\_id = :DOMAIN\_INST\_ID

where csr\_device\_name = :csr\_device\_name ';

FOR rec IN unknown\_domain

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.domain\_inst\_id,

rec.csr\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from Xng Only for '

|| rec.csr\_device\_name

|| ' in table: SAM\_GI\_AL\_CSR\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

BEGIN

-- if some ne\_hostname matched > 1 granite ne\_inst\_id then mark it dup

-- does stmt query work even when there are "Dup in NE"???

UPDATE sam\_gi\_al\_csr\_audit\_wk aud

SET clli = SUBSTR (aud.csr\_device\_name, 1, 6)

WHERE clli = 'unknown';

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname

|| '(): Can''t update clli for unknown';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (processname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

RAISE;

END;

END assign\_area\_region\_to\_csr;

PROCEDURE audit\_al\_csr\_vlans

IS

methodname VARCHAR2 (30) := 'audit\_al\_csr\_vlans';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE ('delete SAM\_CSR\_VLAN\_AUDIT\_WK where CSR\_vendor = ''AL''');

COMMIT;

sqlstmt :=

'insert into SAM\_CSR\_VLAN\_AUDIT\_WK columns (CSR\_Device\_Name, CSR\_vendor

, ne\_hostName, displayed\_name, node\_id, ne\_vlan\_number

, match\_code, match\_status

, xng\_vlan\_number

, vlan\_inst\_id

, vlan\_status

, port\_name

)

WITH

al\_vlans AS

(SELECT xnvp.\*, xne.CSR\_DEVICE\_NAME

FROM xng\_CSR\_vlan\_paths\_wk xnvp , XNG\_CSR\_PARSED\_WK xne

where xne.EQUIP\_INST\_ID = xnvp.EQUIP\_INST\_ID and

xne.CSR\_DEVICE\_NAME like ''%-AL-%'' AND xnvp.vlan\_number IS NOT NULL),

matches AS

(SELECT aud.csr\_device\_name, sam.out\_encap\_val ne\_vlan\_number,

sam.displayed\_name, sam.node\_id, ''BOTH'' match\_code,

NULL match\_status,

xnvp.vlan\_number xng\_vlan\_number, xnvp.vlan\_inst\_id,

xnvp.vlan\_status, aud.CSR\_VENDOR, SUBSTR(sam.PORT\_NAME, 0, INSTR(sam.PORT\_NAME, '':'')-1) AS PORT\_NAME

FROM sam\_ebh\_rtr sam JOIN SAM\_GI\_AL\_CSR\_AUDIT\_WK aud

ON upper(aud.CSR\_DEVICE\_NAME) = upper(sam.node\_name)

JOIN al\_vlans xnvp

ON aud.EQUIP\_INST\_ID = xnvp.EQUIP\_INST\_ID

AND sam.out\_encap\_val = xnvp.vlan\_number

WHERE sam.node\_id IS NOT NULL AND sam.device\_type = ''CSR''),

ne\_only AS

(SELECT aud.CSR\_DEVICE\_NAME, sam.displayed\_name, sam.node\_id,

sam.out\_encap\_val ne\_vlan\_number, ''NE Only'' match\_code,

''Missing in Granite'' match\_status, null,

null xng\_vlan\_number, NULL vlan\_inst\_id, NULL vlan\_status,

aud.CSR\_VENDOR, SUBSTR(sam.PORT\_NAME, 0, INSTR(sam.PORT\_NAME, '':'')-1) AS PORT\_NAME

FROM sam\_ebh\_rtr sam JOIN SAM\_GI\_AL\_CSR\_AUDIT\_WK aud

ON upper(aud.CSR\_DEVICE\_NAME) = upper(sam.node\_name)

WHERE sam.out\_encap\_val IS NOT NULL

AND sam.node\_id IS NOT NULL

AND sam.device\_type = ''CSR''

AND NOT EXISTS (

SELECT 1

FROM al\_vlans xnvp

WHERE aud.EQUIP\_INST\_ID = xnvp.EQUIP\_INST\_ID

AND upper(xnvp.csr\_device\_name) = upper(aud.csr\_device\_name)

AND sam.out\_encap\_val = xnvp.vlan\_number)),

xng\_only AS

(SELECT aud.csr\_device\_name, NULL displayed\_name, NULL node\_id,

null ne\_vlan\_number, ''Xng Only'' match\_code,

''Missing in NE'' match\_status,

xnvp.vlan\_number xng\_vlan\_number, xnvp.vlan\_inst\_id, xnvp.vlan\_status,

aud.CSR\_VENDOR, NULL PORT\_NAME

FROM al\_vlans xnvp JOIN SAM\_GI\_AL\_CSR\_AUDIT\_WK aud

ON aud.EQUIP\_INST\_ID = xnvp.EQUIP\_INST\_ID

AND upper(AUD.CSR\_DEVICE\_NAME) = upper(xnvp.CSR\_DEVICE\_NAME)

and not exists(select 1 from sam\_ebh\_rtr sam

where upper(aud.csr\_device\_name) = upper(sam.node\_name)

AND sam.out\_encap\_val = xnvp.vlan\_number

and sam.out\_encap\_val IS NOT NULL AND sam.device\_type = ''CSR''))

SELECT CSR\_DEVICE\_NAME, CSR\_vendor, CSR\_device\_name, displayed\_name,

node\_id, ne\_vlan\_number, match\_code, match\_status,

xng\_vlan\_number, vlan\_inst\_id, vlan\_status, port\_name

FROM matches

UNION

SELECT CSR\_DEVICE\_NAME, CSR\_vendor, CSR\_device\_name, displayed\_name,

node\_id, ne\_vlan\_number, match\_code, match\_status,

xng\_vlan\_number, vlan\_inst\_id, vlan\_status, port\_name

FROM ne\_only

UNION

SELECT CSR\_DEVICE\_NAME, CSR\_vendor, NULL, displayed\_name, node\_id,

ne\_vlan\_number, match\_code, match\_status, xng\_vlan\_number,

vlan\_inst\_id, vlan\_status, port\_name

FROM xng\_only

';

EXECUTE IMMEDIATE (sqlstmt);

COMMIT;

insert\_al\_CSR\_vlan\_issues ();

EXCEPTION

WHEN OTHERS

THEN

MESSAGE := 'Error: ' || methodname || '(): Can''t audit AL VLANs';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

/\*watchdog.logerror (methodName,

8000,

SUBSTR (message

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

); \*/

RAISE;

END;

END audit\_al\_csr\_vlans;

PROCEDURE load\_csr\_vlan\_regional\_summ

IS

methodname VARCHAR2 (30) := 'load\_vlan\_regional\_summary';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

EXECUTE IMMEDIATE 'truncate table SAM\_csr\_VLAN\_REGION\_SUMM\_WK';

-- generate device summary

sqlstmt :=

' insert into SAM\_CSR\_VLAN\_REGION\_SUMM\_WK

columns (area, region

, ne\_total\_csr\_vlan

, matched\_l\_csr\_vlan

, matched\_nl\_csr\_vlan

, mismatched\_csr\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_compliance)

select dlr.area, dlr.region

, sum(ne\_total\_csr\_vlan) ne\_total\_csr\_vlan

, sum(matched\_l\_csr\_vlan) matched\_l\_csr\_vlan

, sum(matched\_nl\_csr\_vlan) matched\_nl\_csr\_vlan

, sum(mismatched\_csr\_vlan) mismatched\_csr\_vlan

, round(sum(NVL(matched\_nl\_csr\_vlan, 0)) / decode(sum(NVL(matched\_l\_csr\_vlan, 0)), 0, 1, sum(nvl(matched\_l\_csr\_vlan, 0)))

\* 100, 2) gi\_nl\_2\_total\_vlan\_pct

, round (sum(nvl(matched\_l\_csr\_vlan,0)) / decode (sum(nvl( ne\_total\_csr\_vlan, 0)) - sum(nvl(matched\_nl\_csr\_vlan, 0)), 0 , 1

, sum(nvl(ne\_total\_csr\_vlan,0)) - sum(nvl(matched\_nl\_csr\_vlan,0)))

\* 100 , 2) vlan\_compliance

from

SAM\_CSR\_VLAN\_CLLI\_SUMM\_WK dlr

group by rollup (dlr.area, dlr.region)

ORDER BY AREA, REGION ';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname || '(): Can''t generate regional summary';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END load\_csr\_vlan\_regional\_summ;

PROCEDURE insert\_al\_CSR\_vlan\_issues

IS

methodname VARCHAR2 (30) := 'insert\_al\_CSR\_vlan\_issues';

MESSAGE VARCHAR2 (300);

sqlstmt VARCHAR2 (32767);

CURSOR dup\_l\_vlan

IS

-- if the same DEVICE\_NAME, nxa.VLAN\_number has > 1 Live status then

-- mark it as dup

WITH vlan\_status AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM sam\_csr\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

dup\_l\_vlans AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number,

COUNT (vlan\_inst\_id)

FROM vlan\_status nxa

WHERE status = 'Live' AND match\_code <> 'Xng Only'

GROUP BY nxa.csr\_device\_name, nxa.xng\_vlan\_number, status

HAVING COUNT (vlan\_inst\_id) > 1)

SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM dup\_l\_vlans dup JOIN vlan\_status nxa

ON nxa.csr\_device\_name = dup.csr\_device\_name

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

ORDER BY nxa.csr\_device\_name, nxa.xng\_vlan\_number;

-- if the same csr\_device\_name, nxa.VLAN\_number has > 1 status then

-- and 1 is Live and other is not then mark NL as 'NE Only' & 'Live path exists'

CURSOR live\_vlan\_exists

IS

WITH vlan\_status AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, status,

match\_code, nxa.vlan\_inst\_id

FROM sam\_csr\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

l\_and\_nl\_vlans AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status = 'Live'

INTERSECT

SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status <> 'Live')

SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number,

nxa.vlan\_inst\_id

FROM l\_and\_nl\_vlans dup JOIN vlan\_status nxa

ON nxa.csr\_device\_name = dup.csr\_device\_name

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

WHERE nxa.status <> 'Live'

ORDER BY nxa.csr\_device\_name, nxa.xng\_vlan\_number;

BEGIN

DBMS\_OUTPUT.put\_line ('FYI: In ' || methodname);

BEGIN

EXECUTE IMMEDIATE ('delete CSR\_VLAN\_AUDIT\_ISSUES\_WK where csr\_vendor = ''AL''');

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t delete rows in table: CSR\_VLAN\_AUDIT\_ISSUES\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

-- mark duplicate in Granite as NE Only, or Live Path Exists

-- if there is a path revision then the same VLAN will be present twice.

-- BUT that is NOT a duplicate

-- if NOT a revision path and VLAN# is present multiple times THEN it is dup

-- mark duplicate in Granite as NE Only, or Live Path Exists

-- else 2 DUP vlans on a csr then Duplicate

sqlstmt :=

'update SAM\_CSR\_VLAN\_AUDIT\_WK

set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

, match\_status = nvl2(match\_status, match\_status|| '', Dup in GI'', ''Dup in GI'')

where vlan\_inst\_id = :p\_vlan\_inst\_id

and XNG\_VLAN\_NUMBER = :p\_xng\_vlan\_number

and CSR\_DEVICE\_NAME = :p\_csr\_device\_name

';

FOR rec IN dup\_l\_vlan

LOOP

BEGIN

EXECUTE IMMEDIATE sqlstmt

USING rec.vlan\_inst\_id,

rec.xng\_vlan\_number,

rec.csr\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update ''Dup in GI'' in table: SAM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

BEGIN

sqlstmt :=

'insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

select distinct W.CSR\_DEVICE\_NAME

, case when ne\_vlan\_number is null then xng\_vlan\_number

else ne\_vlan\_number

end vlan\_number,1,''AL''

from SAM\_CSR\_VLAN\_AUDIT\_wk w

where W.MATCH\_STATUS like ''%Dup in GI%''';

EXECUTE IMMEDIATE sqlstmt;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK issue Duplicate in GI ';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

COMMIT;

BEGIN

-- check the ne\_hostname matches MTCE std, else update match\_code = ''NE Only''

-- and match\_status = ''ne\_hostname not MTCE Stds compliant''

sqlstmt :=

' update SAM\_CSR\_VLAN\_AUDIT\_WK aud set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

where exists (select 1

from SAM\_EBH\_RTR i

where DEVICE\_TYPE = ''CSR''

and AUD.CSR\_DEVICE\_NAME = I.NODE\_NAME

and parse\_status like ''%11%''

)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

sqlstmt :=

'insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

select distinct W.CSR\_DEVICE\_NAME

, case when ne\_vlan\_number is null then xng\_vlan\_number

else ne\_vlan\_number

end vlan\_number,17,''AL''

from SAM\_CSR\_VLAN\_AUDIT\_wk w, SAM\_EBH\_RTR n

where

W.CSR\_VENDOR = ''AL''

and W.CSR\_DEVICE\_NAME = N.NODE\_NAME

and n.parse\_status like ''%11%''

and n.DEVICE\_TYPE = ''CSR''';

--and regexp\_substr(parse\_status, '[[:digit:]]+') = '11'

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update name does not match MTCE Std';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- check if ne\_hostnames' don't match first 6-char match some MTSO CLLI

-- then match\_code = ''NE Only'' and match\_status = ''1st 6-chars of don''''t match any MTSO CLLIs''

sqlstmt :=

'insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

WITH

CLLI\_6\_MATCH AS

(

SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6

FROM clli\_domain\_map cdm

where cdm.LEAF\_DOMAIN\_INST\_ID not in (1021, 1016)

)

,

CLLI\_MISMATCH AS

(

-- get devices where 6-chars

SELECT ei.CSR\_device\_name,

SUBSTR (ei.csr\_device\_name, 1, 8) csr\_clli, cdm6.clli\_6 vsm\_clli

FROM SAM\_CSR\_VLAN\_AUDIT\_wk ei JOIN clli\_6\_match cdm6

ON cdm6.clli\_6 = SUBSTR (ei.csr\_device\_name, 1, 6)

WHERE NOT EXISTS (SELECT cdm.clli

FROM clli\_domain\_map cdm

WHERE SUBSTR (ei.csr\_device\_name, 1, 8) = cdm.clli)

),

BAD\_CLLI AS

(

SELECT ei.csr\_device\_name

FROM SAM\_CSR\_VLAN\_AUDIT\_wk ei

MINUS

SELECT ei.csr\_device\_name

FROM SAM\_CSR\_VLAN\_AUDIT\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 8) = SUBSTR (ei.csr\_device\_name, 1, 8)

MINUS

SELECT ei.csr\_device\_name

FROM SAM\_CSR\_VLAN\_AUDIT\_wk ei JOIN clli\_domain\_map cdm

ON SUBSTR (cdm.clli, 1, 6) = SUBSTR (ei.csr\_device\_name, 1, 6)

),

ISSUES AS

(

SELECT cm.csr\_device\_name, 22 issue\_id

FROM clli\_mismatch cm

UNION

SELECT bc.csr\_device\_name, 2 issue\_id

FROM bad\_clli bc

)

SELECT distinct i.csr\_device\_name, W.NE\_VLAN\_NUMBER, i.issue\_id, ''AL''

FROM ISSUES i, SAM\_CSR\_VLAN\_AUDIT\_wk w

where i.csr\_device\_name = W.CSR\_DEVICE\_NAME

and W.NE\_VLAN\_NUMBER is not null

';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

NULL;

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t mark 1st 6-char don''t match MTSO CLLIs';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in sam vlan extract

sqlstmt :=

' insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

select distinct aud.CSR\_DEVICE\_NAME, xng\_vlan\_number, 13, ''AL''

from SAM\_CSR\_VLAN\_AUDIT\_wk aud

where aud.ne\_hostname is null';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant insert issue ''Not in SAM VLAN Extract''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in GI

sqlstmt :=

' insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

select distinct aud.CSR\_DEVICE\_NAME, ne\_vlan\_number, 15, ''AL''

from SAM\_CSR\_VLAN\_AUDIT\_wk aud

where aud.vlan\_inst\_id is null';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname

|| '(): Cant insert issue ''Not in GI''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

-- if there is a path revision then the same VLAN will be present twice.

-- BUT that is NOT a duplicate

-- if NOT a revision path and VLAN# is present multiple times THEN it is dup

-- if 1 == Live and the other <> Live then mark <> Live as 'Live in Xng'

sqlstmt :=

'update SAM\_CSR\_VLAN\_AUDIT\_WK

set match\_code = case when match\_code <> ''Xng Only'' then ''NE Only'' else ''Xng Only'' end

, match\_status = nvl2(match\_status, ''Live Path Exists,'' ||match\_status , ''Live Path Exists'')

where vlan\_inst\_id = :p\_vlan\_inst\_id

and XNG\_VLAN\_NUMBER = :p\_xng\_vlan\_number

and CSR\_DEVICE\_NAME = :p\_csr\_device\_name

--and csr\_vendor = :p\_csr\_vendor

';

FOR rec IN live\_vlan\_exists

LOOP

BEGIN

EXECUTE IMMEDIATE sqlstmt

USING rec.vlan\_inst\_id,

rec.xng\_vlan\_number,

rec.csr\_device\_name; --, p\_csr\_vendor;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update ''Live Path Exists'' in table: SAM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END LOOP;

COMMIT;

BEGIN

-- IF BOTH and has issues then mark it NE

sqlstmt :=

' update SAM\_CSR\_VLAN\_AUDIT\_WK aud

set match\_code = ''NE Only''

where exists (select 1

from

CSR\_VLAN\_AUDIT\_ISSUEs\_wk vi,

CSR\_issue n

where aud.MATCH\_CODE = ''BOTH''

and aud.csr\_vendor = ''AL''

and VI.CSR\_VENDOR=''AL''

and AUD.CSR\_DEVICE\_NAME = VI.CSR\_DEVICE\_NAME

and AUD.NE\_VLAN\_NUMBER = VI.VLAN\_NUMBER

and N.IS\_CRITICAL=''Y''

and N.CSR\_ISSUE\_ID = vi.CSR\_ISSUE\_ID

)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant update match\_status to ''NE Only''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

-- device not found in sam vlan extract

sqlstmt :=

' insert into CSR\_VLAN\_AUDIT\_ISSUES\_WK

columns (csr\_device\_name, vlan\_number, csr\_issue\_id, csr\_vendor)

WITH vlan\_status AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, status, match\_code,

nxa.vlan\_inst\_id

FROM sam\_csr\_vlan\_audit\_wk nxa JOIN vzwnet.circ\_path\_inst cpi

ON cpi.circ\_path\_inst\_id = nxa.vlan\_inst\_id

),

l\_and\_nl\_vlans AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status = ''Live''

INTERSECT

SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number

FROM vlan\_status nxa

WHERE status <> ''Live''),

live\_vlan\_exists AS

(SELECT nxa.csr\_device\_name, nxa.xng\_vlan\_number, nxa.vlan\_inst\_id

FROM l\_and\_nl\_vlans dup JOIN vlan\_status nxa

ON nxa.csr\_device\_name = dup.csr\_device\_name

AND nxa.xng\_vlan\_number = dup.xng\_vlan\_number

WHERE nxa.status <> ''Live'')

SELECT DISTINCT aud.csr\_device\_name, xng\_vlan\_number, 31, ''AL''

FROM sam\_csr\_vlan\_audit\_wk aud

WHERE vlan\_inst\_id IS NOT NULL

AND match\_code = ''NE Only''

AND NOT EXISTS (

SELECT 1

FROM csr\_vlan\_audit\_issues\_wk nvai

WHERE nvai.vlan\_number = aud.xng\_vlan\_number

AND aud.csr\_device\_name = nvai.csr\_device\_name

AND nvai.csr\_vendor = ''AL'')

AND EXISTS (

SELECT 1

FROM live\_vlan\_exists lle

WHERE lle.csr\_device\_name = aud.csr\_device\_name

AND aud.xng\_vlan\_number = lle.xng\_vlan\_number)';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Cant insert issue ''Live VLAN Exists''';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

END insert\_al\_CSR\_vlan\_issues;

PROCEDURE load\_vlan\_csr\_clli\_summary

IS

methodname VARCHAR2 (30) := 'load\_vlan\_csr\_clli\_summary';

MESSAGE VARCHAR2 (500);

sqlstmt VARCHAR2 (32767);

BEGIN

EXECUTE IMMEDIATE 'truncate table SAM\_CSR\_VLAN\_CLLI\_SUMM\_WK';

-- generate device summary

sqlstmt :=

' insert into SAM\_CSR\_VLAN\_CLLI\_SUMM\_WK

columns (area, region, market

, clli

, ne\_total\_csr\_vlan

, matched\_l\_csr\_vlan

, matched\_nl\_csr\_vlan

, mismatched\_csr\_vlan

, gi\_nl\_2\_total\_vlan\_pct

, vlan\_COMPLIANCE)

WITH

csr as

(

select distinct area

, region, market, leaf\_domain\_inst\_id, clli

, csr\_device\_name, csr\_vendor

from SAM\_CSR\_VLAN\_AUDIT\_WK ng

where ng.CSR\_DEVICE\_NAME is not null

),

csr\_VLAN as

(

select distinct node\_name CSR\_DEVICE\_NAME, OUT\_ENCAP\_VAL vlan\_number

from SAM\_EBH\_RTR ng

where ng.OUT\_ENCAP\_VAL > 0

and DEVICE\_TYPE =''CSR''

)

,

CNT\_csr\_VLANs as

(

select distinct ng.CSR\_DEVICE\_NAME, count(vlan\_number) total\_ne\_csr\_vlan

from csr\_VLAN nv

join csr ng

on ng.CSR\_DEVICE\_NAME = nv.CSR\_DEVICE\_NAME

group by ng.CSR\_DEVICE\_NAME

),

live\_match as

(

select distinct aud.CSR\_DEVICE\_NAME, aud.ne\_vlan\_number, vlan\_status

from SAM\_CSR\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.csr\_vendor = ''AL''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status = ''Live''

)

,nonlive\_match as

(

select distinct aud.csr\_DEVICE\_NAME, aud.ne\_vlan\_number

from SAM\_CSR\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.csr\_vendor = ''AL''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status <> ''Live''

minus

select distinct aud.csr\_DEVICE\_NAME, aud.ne\_vlan\_number

from SAM\_CSR\_VLAN\_AUDIT\_WK aud

where match\_code = ''BOTH''

and aud.csr\_vendor = ''AL''

and aud.ne\_vlan\_number > 0

and aud.vlan\_status = ''Live''

) ,

CNT\_BOTH\_LIVE\_VLANs as

( -- good matches

select csr\_DEVICE\_NAME, nvl (count(csr\_DEVICE\_NAME), 0) matched\_l\_csr\_vlan

from live\_match aud

group by csr\_DEVICE\_NAME

),

CNT\_BOTH\_NONLIVE\_VLANs as

( -- good matches

select csr\_DEVICE\_NAME, nvl( count(csr\_DEVICE\_NAME), 0) matched\_nl\_csr\_vlan

-- , nvl(sum(case when vlan\_status = ''Live'' then 1 else 0 end), 0) matched\_l\_csr\_vlan

-- , nvl(sum(case when vlan\_status <> ''Live'' then 1 else 0 end), 0) matched\_nl\_csr\_vlan

from nonlive\_match aud

group by aud.csr\_DEVICE\_NAME

) ,

MISMATCH\_VLANs as

( -- bad matches

select distinct aud.csr\_DEVICE\_NAME, aud.ne\_vlan\_number, aud.xng\_vlan\_number, vlan\_status

from SAM\_CSR\_VLAN\_AUDIT\_WK aud

where match\_code <> ''BOTH''

and aud.csr\_vendor = ''AL''

and (aud.ne\_vlan\_number > 0 or aud.xng\_vlan\_number > 0)

)

,

CNT\_MISMATCH\_VLANs as

( -- bad match cnts

select aud.CSR\_DEVICE\_NAME

, count(1) mismatch\_cnt

from MISMATCH\_VLANs aud

group by aud.csr\_DEVICE\_NAME

)

select area, region, market,clli

, sum(nvl(nv.total\_ne\_csr\_vlan, 0)) ne\_total\_csr\_vlan

, sum(nvl(matched\_l\_csr\_vlan, 0)) matched\_l\_csr\_vlan

, sum(nvl(matched\_nl\_csr\_vlan, 0)) matched\_nl\_csr\_vlan

, sum(nvl(mismatch\_cnt, 0)) mismatched\_csr\_vlan

, round(sum(NVL(matched\_nl\_csr\_vlan, 0)) / decode(sum(NVL(matched\_l\_csr\_vlan, 0)), 0, 1, sum(nvl(matched\_l\_csr\_vlan, 0)))

\* 100, 2) gi\_nl\_2\_total\_vlan\_pct

, round (sum(nvl(matched\_l\_csr\_vlan,0)) / decode (sum(nvl( nv.total\_ne\_csr\_vlan, 0)) - sum(nvl(matched\_nl\_csr\_vlan, 0)), 0 , 1

, sum(nvl(nv.total\_ne\_csr\_vlan,0)) - sum(nvl(matched\_nl\_csr\_vlan,0)))

\* 100 , 2) vlan\_COMPLIANCE

from csr ng

left outer join CNT\_csr\_VLANs nv

on nv.csr\_DEVICE\_NAME = ng.csr\_DEVICE\_NAME

left outer join CNT\_BOTH\_LIVE\_VLANs lmv

on lmv.csr\_DEVICE\_NAME = ng.csr\_DEVICE\_NAME

left outer join CNT\_BOTH\_NONLIVE\_VLANs nlmv

on nlmv.csr\_DEVICE\_NAME = ng.csr\_DEVICE\_NAME

left outer join CNT\_MISMATCH\_VLANs mv

on mv.csr\_DEVICE\_NAME = ng.csr\_DEVICE\_NAME

group by ng.area, ng.region, ng.market, csr\_Vendor, clli';

EXECUTE IMMEDIATE sqlstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: ' || methodname || '(): Can''t generate cllil summary';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END load\_vlan\_csr\_clli\_summary;

PROCEDURE assign\_area\_region\_to\_csr\_vlan

IS

methodname VARCHAR2 (30) := 'assign\_area\_region\_to\_csr\_vlan';

MESSAGE VARCHAR2 (300);

updstmt VARCHAR2 (32767);

processname VARCHAR2 (30) := 'SAM\_GI\_CSR\_AUDIT';

CURSOR area\_region\_dev

IS

WITH domains AS

(SELECT DISTINCT SUBSTR (clli, 1, 6) clli\_6, area, region,

market, leaf\_domain\_inst\_id

FROM clli\_domain\_map\_v

WHERE region <> 'NNO' AND region <> 'OSS')

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

csr\_device\_name, clli\_6

FROM SAM\_CSR\_VLAN\_AUDIT\_WK aud JOIN domains di

ON di.clli\_6 = SUBSTR (aud.csr\_device\_name, 1, 6)

;

CURSOR area\_region\_SAM

IS

SELECT di.area, di.region, di.market, di.leaf\_domain\_inst\_id,

di.csr\_device\_name, di.clli

FROM SAM\_CSR\_VLAN\_AUDIT\_WK aud JOIN SAM\_GI\_AL\_CSR\_AUDIT\_WK di

ON di.csr\_device\_name = aud.csr\_device\_name

where aud.area is null;

BEGIN

updstmt :=

'update SAM\_CSR\_VLAN\_AUDIT\_WK set area = :area

, region = :region, market = :market

, leaf\_domain\_inst\_id = :leaf\_domain\_inst\_id

, clli = :clli\_6

where csr\_device\_name = :csr\_device\_name';

FOR rec IN area\_region\_dev

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli\_6,

rec.csr\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update clli for '

|| rec.csr\_device\_name

|| ' in table: SAM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (methodname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

FOR rec IN area\_region\_SAM

LOOP

BEGIN

EXECUTE IMMEDIATE updstmt

USING rec.area,

rec.region,

rec.market,

rec.leaf\_domain\_inst\_id,

rec.clli,

rec.csr\_device\_name;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update clli for '

|| rec.csr\_device\_name

|| ' in table: SAM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror (methodname,

8000,

SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

ROLLBACK;

RAISE;

END;

END LOOP;

-- area region for put xng\_only based on the area/region of ne\_inst\_id???????????

BEGIN

-- if there still are some devices which cant be mapped to any domain then mark those as unknown

updstmt :=

' update SAM\_CSR\_VLAN\_AUDIT\_WK

set area = ''unknown''

, region = ''unknown''

, market = ''unknown''

, leaf\_domain\_inst\_id = 0

, clli = SUBSTR (csr\_device\_name, 1, 6)

where AREA is null ';

EXECUTE IMMEDIATE updstmt;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

MESSAGE :=

'Error: '

|| methodname

|| '(): Can''t update area/region from to unknown in table: SAM\_CSR\_VLAN\_AUDIT\_WK';

DBMS\_OUTPUT.put\_line (SUBSTR ( MESSAGE

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

ROLLBACK;

RAISE;

END;

END assign\_area\_region\_to\_csr\_vlan;

END sam\_ebh\_audit;

/

--------------------------------------------------------

-- DDL for Package Body SEVONE\_PERFORMANCE

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."SEVONE\_PERFORMANCE" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: sevone\_performance

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 10/9/2013 ramamla 1. Created this package body.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE graniteSevOneAudit IS

sql\_stmt varchar2(32767);

BEGIN

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_SUCCESS',

PROCESSNAME || ' started successfully!',

'N');

-- Tell watchDog the process has started

watchdog.updateprocessstart (processName);

execute immediate ('delete from XNG\_REPORTS.SEVONE\_PERFORMANCE\_DATA where value=0');

commit;

graniteSevOneCsrAudit();

graniteSevOneMplsAudit();

graniteSevOneNgmlsAudit();

graniteSevOne6500Audit();

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_SUCCESS',

PROCESSNAME || ' ran successfully!',

'N');

WATCHDOG.UPDATEPROCESSEND (PROCESSNAME, 'STATUS\_SUCCESS', 'Y');

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.PUT\_LINE (

SUBSTR (

'Error: sevone\_performance.graniteSevOneAudit(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255));

-- mark failed completion of the task

WATCHDOG.UPDATEPROCESSEND (PROCESSNAME, 'STATUS\_FAILURE', 'N');

END graniteSevOneAudit;

PROCEDURE graniteSevOne6500Audit IS

sql\_stmt varchar2(32767);

ERR\_MSG VARCHAR2(1000);

err\_msg\_prefix varchar2(256);

BEGIN

err\_msg\_prefix:='delete from SEVONE\_PARSED\_DATA\_WK';

execute immediate ('delete from XNG\_REPORTS.SEVONE\_PARSED\_DATA\_WK where element\_type=''6500\_Ethernet'' ');

commit;

err\_msg\_prefix:='Insert 6500 into SEVONE\_PARSED\_DATA\_WK';

sql\_stmt := ' INSERT INTO SEVONE\_PARSED\_DATA\_WK columns( DEVICE\_ID,DEVICE\_NAME,BANDWIDTH,PORT\_NAME,VLAN\_NUMBER

,DEVICE\_IP,ELEMENT\_TYPE,TR\_TOKEN,EXTRACT\_DATE)

SELECT DISTINCT SPD.DEVICE\_ID, spd.device\_name,'''' bandwidth

,regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)((\d+\/)+\d+)?'') ,''((\d+\/)+\d+)?$'')port

,(case

when

regexp\_like (object\_name, ''(VLAN-|Vl)(\d+)'')

then

regexp\_substr(regexp\_substr(object\_name, ''(VLAN-|Vl)(\d+)''), ''(\d+)'')

else

regexp\_substr(regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)((\d+\/)+\d+)?(\.(\d+))?'') ,''(\.(\d+))?$''), ''(\d+)'')

end) vlan

,spd.device\_ip

,''6500\_Ethernet''

,NVL(SPD.VALUE, 0) tr\_token

,sysdate

from SEVONE\_PERFORMANCE\_DATA spd

where regexp\_like (device\_name, ''^\w{11}-\w-650'')

order by device\_id, device\_name, device\_ip'

;

/\*

sql\_stmt := ' insert into XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date )

with

paths\_w\_6500 as (

select distinct ei.descr, ei.equip\_inst\_id,epa.PORT\_INST\_ID,epa.port\_hum\_id,slot||''/''||port\_hum\_id xng\_port\_name,

regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

,CPi.CIRC\_PATH\_INST\_ID, CPI.CIRC\_PATH\_HUM\_ID

from vzwnet.equip\_inst ei

,vzwnet.epa

,VZWNET.CIRC\_PATH\_ELEMENT cpe

,vzwnet.circ\_path\_inst cpi

,vzwnet.card\_inst ci

where

ei.descr like ''%-6500-%''

and ei.EQUIP\_INST\_ID = epa.EQUIP\_INST\_ID

and CPE.PORT\_INST\_ID = epa.PORT\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID in (EPA.CIRC\_PATH\_INST\_ID, EPA.NEXT\_PATH\_INST\_ID)

and CI.CARD\_INST\_ID = epa.CARD\_INST\_ID

)

select distinct e.device\_name ,xng\_port\_name, circ\_path\_inst\_id, PORT\_INST\_ID, ''Y'' has\_traffic, extract\_date

--,port\_hum\_id,port\_name, fix2

from

paths\_w\_6500 w

,xng\_reports.SEVONE\_PARSED\_DATA\_WK e

where

parse\_status is null

and tr\_token > 0

and e.ELEMENT\_TYPE = ''6500\_Ethernet''

and w.descr like ''%''||e.device\_name||''%''

and fix2 = port\_name

';\*/

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='delete 6500 from VZW\_SEVONE\_TRAFFIC\_WK';

sql\_stmt := 'delete from XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK where element\_type=''6500'' ';

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='Insert 6500 into VZW\_SEVONE\_TRAFFIC\_WK';

sql\_stmt := ' insert into XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK columns (hostname,port\_name,circ\_path\_inst\_id,port\_inst\_id,has\_traffic,extract\_date,element\_type )

with

paths\_w\_6500 as (

select distinct ei.descr, ei.equip\_inst\_id,epa.PORT\_INST\_ID,epa.port\_hum\_id,slot||''/''||port\_hum\_id xng\_port\_name,

regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

,CPi.CIRC\_PATH\_INST\_ID, CPI.CIRC\_PATH\_HUM\_ID

from vzwnet.equip\_inst ei

,vzwnet.epa

,VZWNET.CIRC\_PATH\_ELEMENT cpe

,vzwnet.circ\_path\_inst cpi

,vzwnet.card\_inst ci

where

ei.descr like ''%-6500-%''

and ei.EQUIP\_INST\_ID = epa.EQUIP\_INST\_ID

and CPE.PORT\_INST\_ID = epa.PORT\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID in (EPA.CIRC\_PATH\_INST\_ID, EPA.NEXT\_PATH\_INST\_ID)

and CI.CARD\_INST\_ID = epa.CARD\_INST\_ID

)

select distinct e.device\_name ,xng\_port\_name, circ\_path\_inst\_id, PORT\_INST\_ID, ''Y'' has\_traffic, extract\_date, ''6500''

--,port\_hum\_id,port\_name, fix2

from

paths\_w\_6500 w

,xng\_reports.SEVONE\_PARSED\_DATA\_WK e

where

parse\_status is null

and e.ELEMENT\_TYPE = ''6500\_Ethernet''

and w.descr like ''%''||e.device\_name||''%''

and fix2 = port\_name

';

execute immediate sql\_stmt;

commit;

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_SUCCESS',

'graniteSevOne6500Audit ran successfully!',

'N');

EXCEPTION

WHEN OTHERS THEN

ERR\_MSG := err\_msg\_prefix || substr(SQLERRM, 1, 150);

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_FAILURE',

'graniteSevOne6500Audit ' || ERR\_MSG,

'N');

END graniteSevOne6500Audit;

PROCEDURE graniteSevOneCsrAudit IS

sql\_stmt varchar2(32767);

ERR\_MSG VARCHAR2(1000);

err\_msg\_prefix varchar2(256);

BEGIN

execute immediate ('truncate table XNG\_REPORTS.SEVONE\_CSR\_PARSED\_DATA\_WK');

err\_msg\_prefix:='Insert into SEVONE\_CSR\_PARSED\_DATA\_WK';

-- CSR both AL and CI

sql\_stmt := '

INSERT INTO XNG\_REPORTS.SEVONE\_CSR\_PARSED\_DATA\_WK columns( DEVICE\_ID,DEVICE\_NAME,BANDWIDTH,PORT\_NAME,VLAN\_NUMBER

,DEVICE\_IP,TR\_TOKEN,EXTRACT\_DATE)

SELECT DISTINCT SPD.DEVICE\_ID, spd.device\_name

, '''' bandwidth

, regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)?((\d+\/)+\d+)?'') ,''((\d+\/)+\d+)?$'')port

,regexp\_substr(regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)?((\d+\/)+\d+)?(\.(\d+))?'') ,''(\.(\d+))?$''),''(\d+)'') vlan

,spd.device\_ip

,SPD.VALUE tr\_token

,sysdate

from SEVONE\_PERFORMANCE\_DATA spd

where regexp\_like (device\_name, ''^\w{8}T\wA-P-(CI|AL)'')

and not regexp\_like (device\_name, ''^\w{8}T\wA-P-AL-7750'') -- exclude SAM NgMLS(7750)??

order by device\_id, device\_name, device\_ip'

;

execute immediate sql\_stmt;

commit;

sql\_stmt := 'Update XNG\_REPORTS.SEVONE\_CSR\_PARSED\_DATA\_WK set element\_type=''CSR AL'' where device\_name like ''%A-P-AL-%'' ';

execute immediate sql\_stmt;

commit;

sql\_stmt := 'Update XNG\_REPORTS.SEVONE\_CSR\_PARSED\_DATA\_WK set element\_type=''CSR CI'' where device\_name like ''%A-P-CI-%'' ';

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='delete SAM CSR from VZW\_SEVONE\_TRAFFIC\_WK';

sql\_stmt := 'delete from XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK where element\_type=''CSR AL'' ';

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='Insert SAM CSR into VZW\_SEVONE\_TRAFFIC\_WK';

-- SAM CSR

sql\_stmt := ' Insert into XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date,element\_type )

with

paths\_w\_sam\_csr as (

select distinct XNE.CSR\_DEVICE\_NAME, XNE.EQUIP\_INST\_ID,epa.PORT\_INST\_ID,epa.port\_hum\_id

,replace(slot,''-'',''/'')||''/''||port\_hum\_id xng\_port\_name

,regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

,CPi.CIRC\_PATH\_INST\_ID, CPI.CIRC\_PATH\_HUM\_ID

from xng\_reports.xng\_csr\_parsed xne

,vzwnet.epa

,VZWNET.CIRC\_PATH\_ELEMENT cpe

,vzwnet.circ\_path\_inst cpi

,vzwnet.card\_inst ci

where

XNE.CSR\_DEVICE\_NAME like ''%-AL-%''

and xne.EQUIP\_INST\_ID = epa.EQUIP\_INST\_ID

and CPE.PORT\_INST\_ID = epa.PORT\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID in (EPA.CIRC\_PATH\_INST\_ID, EPA.NEXT\_PATH\_INST\_ID)

and CI.CARD\_INST\_ID = epa.CARD\_INST\_ID

)

select distinct e.device\_name, port\_hum\_id, circ\_path\_inst\_id, PORT\_INST\_ID, ''Y'' has\_traffic, extract\_date, e.element\_type

--,fix2, xng\_port\_name,port\_name, csr\_device\_name, w.equip\_inst\_id

from

paths\_w\_sam\_csr w

,xng\_reports.SEVONE\_CSR\_PARSED\_DATA\_WK e

where

-- parse\_status is null

e.element\_type=''CSR AL''

-- tr\_token > 0

and e.device\_name = w.csr\_device\_name

and (

port\_name = w.xng\_port\_name

or port\_name = fix2 )

and port\_name <> ''1''

';

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='delete Cisco CSR from VZW\_SEVONE\_TRAFFIC\_WK';

sql\_stmt := 'delete from XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK where element\_type=''CSR CI'' ';

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='Insert Cisco CSR into VZW\_SEVONE\_TRAFFIC\_WK';

-- Cisco CSR

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date, element\_type )

with

paths\_w\_cisco\_csr as (

select distinct XNE.CSR\_DEVICE\_NAME, XNE.EQUIP\_INST\_ID,epa.PORT\_INST\_ID,epa.port\_hum\_id

,CPi.CIRC\_PATH\_INST\_ID, CPI.CIRC\_PATH\_HUM\_ID

from xng\_reports.xng\_csr\_parsed xne

,vzwnet.epa

,VZWNET.CIRC\_PATH\_ELEMENT cpe

,vzwnet.circ\_path\_inst cpi

where

XNE.CSR\_DEVICE\_NAME like ''%-CI-%''

and upper(EPA.BANDWIDTH) like ''%GBPS%''

and xne.EQUIP\_INST\_ID = epa.EQUIP\_INST\_ID

and CPE.PORT\_INST\_ID = epa.PORT\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and CPI.CIRC\_PATH\_INST\_ID in (EPA.CIRC\_PATH\_INST\_ID, EPA.NEXT\_PATH\_INST\_ID)

),

sevone\_match as

(select distinct e.device\_name, port\_hum\_id, circ\_path\_inst\_id, PORT\_INST\_ID, ''Y'' has\_traffic, extract\_date, element\_type

-- port\_name, csr\_device\_name, , w.equip\_inst\_id, , parse\_status

from

paths\_w\_cisco\_csr w

,xng\_reports.SEVONE\_CSR\_PARSED\_DATA\_WK e

where

parse\_status is null

and tr\_token > 0

and port\_name <> ''1''

and e.device\_name = w.csr\_device\_name

and (port\_hum\_id like ''%''||port\_name

or port\_hum\_id like ''%''||port\_name||'' ''||''%'')

) select distinct device\_name, port\_hum\_id, circ\_path\_inst\_id, PORT\_INST\_ID, has\_traffic, extract\_date, element\_type

from sevone\_match

'

;

execute immediate sql\_stmt;

commit;

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_SUCCESS',

'graniteSevOneCsrAudit ran successfully!',

'N');

EXCEPTION

WHEN OTHERS THEN

ERR\_MSG := err\_msg\_prefix || substr(SQLERRM, 1, 150);

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_FAILURE',

'graniteSevOneCsrAudit ' || ERR\_MSG,

'N');

END graniteSevOneCsrAudit;

PROCEDURE graniteSevOneMplsAudit IS

sql\_stmt varchar2(32767);

ERR\_MSG VARCHAR2(1000);

err\_msg\_prefix varchar2(256);

BEGIN

err\_msg\_prefix:='delete from SEVONE\_PARSED\_DATA\_WK';

execute immediate ('delete from XNG\_REPORTS.SEVONE\_PARSED\_DATA\_WK where element\_type=''MPLS'' ');

commit;

err\_msg\_prefix:='Insert MPLS into SEVONE\_PARSED\_DATA\_WK';

-- CSR both AL and CI

-- MPLS / CRS\_GigE

-- CRS devicesare core of MPLS network. MPLS network will have other devices, but all CRS are serving MPLS only

sql\_stmt := '

INSERT INTO XNG\_REPORTS.SEVONE\_PARSED\_DATA\_WK columns( DEVICE\_ID,DEVICE\_NAME,BANDWIDTH,PORT\_NAME,VLAN\_NUMBER

,DEVICE\_IP,ELEMENT\_TYPE,TR\_TOKEN,EXTRACT\_DATE)

SELECT DISTINCT SPD.DEVICE\_ID, spd.device\_name

,regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)((\d+\/)+\d+)?(\.(\d+))?'') ,''^([[:alpha:]]+)'') bandwidth

,regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)((\d+\/)+\d+)?'') ,''((\d+\/)+\d+)?$'')port

,regexp\_substr(regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)((\d+\/)+\d+)?(\.(\d+))?'') ,''(\.(\d+))?$''),''(\d+)'') vlan

,spd.device\_ip

,''MPLS''

,NVL(SPD.VALUE, 0) tr\_token

,sysdate

from SEVONE\_PERFORMANCE\_DATA spd

where regexp\_like (device\_name, ''^\w{8}(7|8)\w(W)+'')

and regexp\_like(object\_name, ''^(GigabitEthernet|TenGigE|HundredGigE)'')

order by device\_id, device\_name, device\_ip'

;

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='delete MPLS from VZW\_SEVONE\_TRAFFIC\_WK';

sql\_stmt := 'delete from XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK where element\_type=''MPLS'' ';

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='Insert MPLS into VZW\_SEVONE\_TRAFFIC\_WK';

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date, element\_type )

with

mpls\_rtrs\_1 as (

select router\_name, port\_inst\_id, port\_hum\_id,

case when circ\_path\_inst\_id is null then next\_path\_inst\_id

else circ\_path\_inst\_id

end circ\_path\_inst\_id

,regexp\_substr(regexp\_substr(port\_access\_id,''([[:alpha:]]+)((\d+\/)+\d+)?'') ,''((\d+\/)+\d+)?$'')port1

,regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(x.port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') port2

from XNG\_MPLS\_RTR\_PORTS\_PARSED x

) --select \* from mpls\_rtrs\_1;

select distinct device\_name, port\_hum\_id, CIRC\_PATH\_INST\_ID, PORT\_INST\_ID, ''Y'' has\_traffic, extract\_date, element\_type

from SEVONE\_PARSED\_DATA\_WK w , mpls\_rtrs\_1

where upper(W.ELEMENT\_TYPE)=''MPLS''

and tr\_token > 0

and router\_name = device\_name

and (port1=port\_name or port2 = port\_name)

'

;

execute immediate sql\_stmt;

commit;

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_SUCCESS',

'graniteSevOneMplsAudit ran successfully!',

'N');

EXCEPTION

WHEN OTHERS THEN

ERR\_MSG := err\_msg\_prefix || substr(SQLERRM, 1, 150);

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_FAILURE',

'graniteSevOneMplsAudit ' || ERR\_MSG,

'N');

END graniteSevOneMplsAudit;

PROCEDURE graniteSevOneNgmlsAudit IS

sql\_stmt varchar2(32767);

ERR\_MSG VARCHAR2(1000);

err\_msg\_prefix varchar2(256);

BEGIN

err\_msg\_prefix:='delete 9010 from SEVONE\_PARSED\_DATA\_WK';

execute immediate ('delete from XNG\_REPORTS.SEVONE\_PARSED\_DATA\_WK where element\_type=''9010'' ');

commit;

err\_msg\_prefix:='Insert 9010 into SEVONE\_PARSED\_DATA\_WK';

-- Load SevOne Cisco 9010/NCM NgMLS data

sql\_stmt := '

INSERT INTO XNG\_REPORTS.SEVONE\_PARSED\_DATA\_WK columns( DEVICE\_ID,DEVICE\_NAME,BANDWIDTH,PORT\_NAME,VLAN\_NUMBER

,DEVICE\_IP,ELEMENT\_TYPE,TR\_TOKEN,EXTRACT\_DATE)

SELECT DISTINCT SPD.DEVICE\_ID, spd.device\_name

,regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)((\d+\/)+\d+)?(\.(\d+))?'') ,''^([[:alpha:]]+)'') bandwidth

,regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)((\d+\/)+\d+)?'') ,''((\d+\/)+\d+)?$'')port

,regexp\_substr(regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)((\d+\/)+\d+)?(\.(\d+))?'') ,''(\.(\d+))?$''),''(\d+)'') vlan

,spd.device\_ip

,''9010''

,NVL(SPD.VALUE, 0) tr\_token

,sysdate

from SEVONE\_PERFORMANCE\_DATA spd

where SPD.DEVICE\_NAME like (''%CI-9010-%'')

and regexp\_like(object\_name, ''^(Giga|Ten)'' )

order by device\_id, device\_name, device\_ip'

;

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='delete 9010 from VZW\_SEVONE\_TRAFFIC\_WK';

sql\_stmt := 'delete from XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK where element\_type=''9010'' ';

execute immediate sql\_stmt;

commit;

-- NCM NGMLS PERFORMANCE

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date,element\_type )

with

paths\_w\_ngmls\_devices as (

select distinct XNE.NGMLS\_DEVICE\_NAME, XNE.EQUIP\_INST\_ID, x.port\_inst\_id, descr,

x.PORT\_HUM\_ID,

regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(x.port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

,cpi.circ\_path\_inst\_id, x.bandwidth

from xng\_reports.xng\_ngMLS\_equip\_wk xne,

vzwnet.epa x,

vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe

where

XNE.VENDOR = ''CISCO''

and x.EQUIP\_INST\_ID = xne.EQUIP\_INST\_ID

and cpi.CIRC\_PATH\_INST\_ID in ( x.CIRC\_PATH\_INST\_ID, x.NEXT\_PATH\_INST\_ID)

and CPE.PORT\_INST\_ID=x.port\_inst\_id

), details as

(select distinct np.circ\_path\_inst\_id,np.port\_inst\_id,e.device\_name,e.port\_name,''Y'' has\_traffic,e.extract\_date,element\_type

, E.BANDWIDTH,port\_hum\_id, np.bandwidth, fix2

from paths\_w\_ngmls\_devices np

,SEVONE\_PARSED\_DATA\_WK e

where

E.ELEMENT\_TYPE = ''9010'' and device\_name is not null

and NGMLS\_DEVICE\_NAME = E.device\_name

and upper(np.BANDWIDTH) like upper(''1 Gbps%'')

and upper(E.BANDWIDTH) = upper(''GigabitEthernet'')

and (fix2 like ''%''||port\_name

or fix2 like ''%''||port\_name||'' ''||''%'' )

union

select distinct np.circ\_path\_inst\_id,np.port\_inst\_id,e.device\_name,e.port\_name,''Y'' has\_traffic,e.extract\_date,element\_type

, E.BANDWIDTH,port\_hum\_id, np.bandwidth, fix2

from paths\_w\_ngmls\_devices np

,SEVONE\_PARSED\_DATA\_WK e

where

E.ELEMENT\_TYPE = ''9010'' and device\_name is not null

and NGMLS\_DEVICE\_NAME = E.device\_name

and upper(np.BANDWIDTH) like upper(''10 Gbps%'')

and upper(E.BANDWIDTH) like upper(''TenGi%'')

and (fix2 like ''%''||port\_name

or fix2 like ''%''||port\_name||'' ''||''%'' )

)select distinct device\_name, port\_hum\_id,circ\_path\_inst\_id, port\_inst\_id, has\_traffic, extract\_date, element\_type from details

';

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='delete 7750 from SEVONE\_PARSED\_DATA\_WK';

execute immediate ('delete from XNG\_REPORTS.SEVONE\_PARSED\_DATA\_WK where element\_type=''7750'' ');

commit;

err\_msg\_prefix:='Insert 7750 into SEVONE\_PARSED\_DATA\_WK';

-- SAM NgMLS

sql\_stmt := '

INSERT INTO SEVONE\_PARSED\_DATA\_WK columns( DEVICE\_ID,DEVICE\_NAME,BANDWIDTH,PORT\_NAME,VLAN\_NUMBER

,DEVICE\_IP,ELEMENT\_TYPE,TR\_TOKEN,EXTRACT\_DATE)

SELECT DISTINCT SPD.DEVICE\_ID, spd.device\_name

, '''' bandwidth

, regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)?((\d+\/)+\d+)?'') ,''((\d+\/)+\d+)?$'')port

,regexp\_substr(regexp\_substr(regexp\_substr(object\_name,''([[:alpha:]]+)?((\d+\/)+\d+)?(\.(\d+))?'') ,''(\.(\d+))?$''),''(\d+)'') vlan

,spd.device\_ip

,''7750''

,NVL(SPD.VALUE, 0) tr\_token

,sysdate

from SEVONE\_PERFORMANCE\_DATA spd

where regexp\_like (device\_name, ''^\w{8}9\dA-P-AL-7750\*'')

order by device\_id, device\_name, device\_ip'

;

execute immediate sql\_stmt;

commit;

err\_msg\_prefix:='delete 7750 from VZW\_SEVONE\_TRAFFIC\_WK';

sql\_stmt := 'delete from XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK where element\_type=''7750'' ';

execute immediate sql\_stmt;

commit;

-- SAM NGMLS PERFORMANCE

sql\_stmt := 'insert into XNG\_REPORTS.VZW\_SEVONE\_TRAFFIC\_WK columns (hostname, port\_name, circ\_path\_inst\_id, port\_inst\_id, has\_traffic,extract\_date,element\_type )

with

paths\_w\_ngmls\_devices as (

select distinct XNE.NGMLS\_DEVICE\_NAME, XNE.EQUIP\_INST\_ID, x.port\_inst\_id, descr,

x.PORT\_HUM\_ID,

regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(regexp\_replace(x.port\_hum\_id, ''/0+'', ''/0''), ''^0+'', ''0''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''(^.\*)(/)0([[:digit:]])(.\*$)'', ''\1\2\3\4''), ''^0([[:digit:]])(.\*$)'', ''\1\2'') fix2

,cpi.circ\_path\_inst\_id, x.bandwidth

from xng\_reports.xng\_ngMLS\_equip\_wk xne,

vzwnet.epa x,

vzwnet.circ\_path\_inst cpi,

vzwnet.circ\_path\_element cpe

where

XNE.VENDOR = ''ALCATEL-LUCENT''

and x.EQUIP\_INST\_ID = xne.EQUIP\_INST\_ID

and cpi.CIRC\_PATH\_INST\_ID in ( x.CIRC\_PATH\_INST\_ID, x.NEXT\_PATH\_INST\_ID)

and CPE.PORT\_INST\_ID=x.port\_inst\_id

)

select distinct device\_name, port\_hum\_id, circ\_path\_inst\_id, port\_inst\_id, ''Y'' has\_traffic, extract\_date, element\_type

-- , port\_name, fix2, np.circ\_path\_inst\_id,np.port\_inst\_id,e.device\_name,e.port\_name,''Y'' has\_traffic,e.extract\_date,e.element\_type

-- , E.BANDWIDTH,port\_hum\_id, np.bandwidth, fix2

from paths\_w\_ngmls\_devices np

,SEVONE\_PARSED\_DATA\_WK e

where

E.ELEMENT\_TYPE = ''7750'' and device\_name is not null

and NGMLS\_DEVICE\_NAME = E.device\_name

and fix2=port\_name '

;

execute immediate sql\_stmt;

commit;

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_SUCCESS',

'graniteSevOneNgmlsAudit ran successfully!',

'N');

EXCEPTION

WHEN OTHERS THEN

ERR\_MSG := err\_msg\_prefix || substr(SQLERRM, 1, 150);

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_FAILURE',

'graniteSevOneNgmlsAudit ' || ERR\_MSG,

'N');

END graniteSevOneNgmlsAudit;

END sevone\_performance;

/

--------------------------------------------------------

-- DDL for Package Body SITE\_PORTAL\_AUDIT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."SITE\_PORTAL\_AUDIT"

IS

--

-- To modify this template, edit file PKGBODY.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package body

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter procedure, function bodies as shown below

PROCEDURE updatesiteportalaudit

IS

sql\_stmt VARCHAR2 (32000);

BEGIN

sql\_stmt := 'truncate table ' || audit\_table\_name;

DBMS\_OUTPUT.put\_line (sql\_stmt);

BEGIN

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not truncate table: '

|| audit\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

END;

BEGIN

sql\_stmt :=

'insert into '

|| audit\_table\_name

|| ' (database\_name, server\_ip, group\_name,

site\_inst\_id, site\_hum\_id, address, city, state\_prov

, post\_code, site\_type, match\_code, match\_status, DOMAIN\_INST\_ID) '

|| ' WITH SITE\_PORTAL\_TMP AS

(select distinct sp.region, sp.database\_name, sp.server\_ip, upper(sp.group\_name) as group\_name

, si.SITE\_INST\_ID, si.SITE\_HUM\_ID, si.ADDRESS,

si.CITY, si.STATE\_PROV, si.POST\_CODE\_1 as post\_code,

si.NUM as SITE\_TYPE,

case when si.site\_inst\_id is not null and not regexp\_like(sp.group\_name, ''.\*-.\*'') and not regexp\_like(sp.group\_name, ''.\*\_.\*'') then ''BOTH''

ELSE ''NE Only''

end match\_code,

case

when si.site\_inst\_id is null and regexp\_like(sp.group\_name, ''.\*-.\*'') then ''Not in GI (Hyphen in Group Name Not Allowed)''

when si.site\_inst\_id is null and regexp\_like(sp.group\_name, ''.\*\_.\*'') then ''Not in GI (Underscore in Group Name Not Allowed)''

when si.site\_inst\_id is null then ''Not in GI''

when regexp\_like(sp.group\_name, ''.\*-.\*'') then ''Hyphen(-) in Group Name Not Allowed''

when regexp\_like(sp.group\_name, ''.\*\_.\*'') then ''Underscore(\_) in Group Name Not Allowed''

end match\_status,

sd.DOMAIN\_INST\_ID as DOMAIN\_INST\_ID

from SITE\_PORTAL\_EXTRACT sp

left outer join vzwnet.site\_inst si

on upper(sp.GROUP\_NAME) = upper(si.SITE\_HUM\_ID)

left join SITE\_PORTAL\_DOMAIN\_MAPPING sd on

sd.SITE\_PORTAL\_DOMAIN\_CODE = sp.region where sp.group\_name is not null)

(select database\_name, server\_ip, group\_name,

site\_inst\_id, site\_hum\_id, address, city, state\_prov,

post\_code, site\_type, match\_code, match\_status,DOMAIN\_INST\_ID

from SITE\_PORTAL\_TMP)

';

DBMS\_OUTPUT.put\_line (sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not truncate table: '

|| audit\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

COMMIT;

BEGIN

sql\_stmt :=

'update '

|| audit\_table\_name

|| ' aud set DUPLICATE = ''Y'' '

|| 'where exists(select 1 from '

|| audit\_table\_name

|| ' i where i.group\_name = aud.group\_name and '

|| '(i.group\_name != aud.group\_name or i.database\_name != aud.database\_name))';

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not update table: '

|| audit\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

COMMIT;

END;

PROCEDURE updatesiteportalauditsummary

IS

sql\_stmt VARCHAR2 (32000);

BEGIN

sql\_stmt := 'truncate table ' || audit\_summary\_table\_name;

DBMS\_OUTPUT.put\_line (sql\_stmt);

BEGIN

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not truncate table: '

|| audit\_summary\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

sql\_stmt :=

'insert into '

|| audit\_summary\_table\_name

|| ' (REGION, AREA, MATCHED\_GI, NO\_MATCH\_IN\_GI, PER\_MATCHED, TOTAL)

(select (select REGION from DOMAINS\_REGIONAL\_REPORTING d where d.domain\_inst\_id = aud.domain\_inst\_id),

(select AREA from DOMAINS\_REGIONAL\_REPORTING d where d.domain\_inst\_id = aud.domain\_inst\_id),

Sum(CASE WHEN MATCH\_CODE = ''BOTH'' Then 1 else 0 end) both,

Sum(CASE WHEN MATCH\_CODE = ''NE Only'' Then 1 else 0 end) NEOnly, 0, count(1) from '

|| audit\_table\_name || ' aud '

|| ' group by aud.domain\_inst\_id)';

--dbms\_output.put\_line(sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not insert into table: '

|| audit\_summary\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

--commit;

BEGIN

sql\_stmt :=

'insert into '

|| audit\_summary\_table\_name

|| ' (AREA, MATCHED\_GI, NO\_MATCH\_IN\_GI, TOTAL) ('

|| 'select AREA, Sum(MATCHED\_GI), '

|| ' Sum(NO\_MATCH\_IN\_GI), '

|| ' Sum(TOTAL) from '

|| audit\_summary\_table\_name

|| ' group by AREA)';

DBMS\_OUTPUT.put\_line (sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

-- When NO\_DATA\_FOUND Then

-- dbms\_output.put\_line(SubStr('Error: No data found to update table: '|| AUDIT\_SUMMARY\_TABLE\_NAME ||

-- 'Error: '|| TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not update table: '

|| audit\_summary\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

sql\_stmt :=

'insert into '

|| audit\_summary\_table\_name

|| ' (MATCHED\_GI, NO\_MATCH\_IN\_GI, TOTAL) values ('

|| '(select Sum(MATCHED\_GI) from '

|| audit\_summary\_table\_name

|| '), (select Sum(NO\_MATCH\_IN\_GI) from '

|| audit\_summary\_table\_name

|| '), (select Sum(TOTAL) from '

|| audit\_summary\_table\_name

|| '))';

DBMS\_OUTPUT.put\_line (sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

-- When NO\_DATA\_FOUND Then

-- dbms\_output.put\_line(SubStr('Error: No data found to update table: '|| AUDIT\_SUMMARY\_TABLE\_NAME ||

-- 'Error: '|| TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not update table: '

|| audit\_summary\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

BEGIN

sql\_stmt :=

'update '

|| audit\_summary\_table\_name

|| ' set PER\_MATCHED = (MATCHED\_GI \* 100.00 / TOTAL)';

-- dbms\_output.put\_line(sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

-- When NO\_DATA\_FOUND Then

-- dbms\_output.put\_line(SubStr('Error: No data found to update table: '|| AUDIT\_SUMMARY\_TABLE\_NAME ||

-- 'Error: '|| TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not update table: '

|| audit\_summary\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

END;

COMMIT;

END;

PROCEDURE restore\_siteportalaudit

IS

sql\_stmt VARCHAR2 (32000);

BEGIN

sql\_stmt :=

'insert into '

|| audit\_table\_name

|| ' (select \* from '

|| prod\_audit\_table\_name

|| ')';

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt :=

'insert into '

|| audit\_summary\_table\_name

|| ' (select \* from '

|| prod\_audit\_summary\_table\_name

|| ')';

EXECUTE IMMEDIATE sql\_stmt;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line

(SUBSTR ( 'Error: Could not restore SP audit table: '

|| audit\_table\_name

|| 'Error: '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

RAISE;

-- dbms\_output.put\_line();

END;

END;

/

--------------------------------------------------------

-- DDL for Package Body T1\_SUMMARY

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."T1\_SUMMARY" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: T1\_Summary

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 4/13/2011 1. Created this package.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE load\_t1\_summary is

cursor T1\_SUMMARY is

select \* from SWITCH\_PATH\_T1\_SUMMARY;

sql\_stmt varchar2(32000);

denominator number;

per\_comp number;

begin

dbms\_output.put\_line('T1\_summary');

sql\_stmt := 'delete from '|| SWITCH\_PATH\_T1\_SUMMARY;

EXECUTE IMMEDIATE sql\_stmt;

sql\_stmt := ' insert into ' || SWITCH\_PATH\_T1\_SUMMARY || '

columns (AREA, REGION, ACTIVE\_NE\_SPANS, ACTIVE\_SPANS\_LIVE\_IN\_XNG,

ACTIVE\_SPANS\_NOT\_FOUND\_IN\_XNG, ACTIVE\_SPANS\_WITH\_EXTRA\_XNG, ACTIVE\_LIVE\_SPANS\_EXEMPT,

ACTIVE\_LIVE\_SPANS\_NOT\_EXEMPT, ACTIVE\_LIVE\_SPANS\_TESTABLE, ACTIVE\_LIVE\_SPANS\_UNTESTABLE )

SELECT

drr.area,

drr.region,

SUM( DECODE( sptt.is\_active\_in\_ne || sptt.is\_backup\_path, ''YN'', 1, 0 ) ) AS active\_ne\_spans,

SUM( DECODE( sptt.is\_active\_in\_ne || sptt.is\_backup\_path || sptt.is\_active\_in\_xng, ''YNY'', 1, 0 ) ) AS active\_spans\_live\_in\_xng,

SUM( DECODE( sptt.is\_active\_in\_ne || sptt.is\_backup\_path || sptt.is\_found\_in\_xng, ''YNN'', 1, 0 ) ) AS active\_spans\_not\_found\_in\_xng,

SUM( DECODE( SIGN( sptt.matching\_live\_xng\_paths - 1 ), 1, 1, 0 ) ) AS active\_spans\_with\_extra\_xng,

SUM( DECODE( sptt.is\_active\_in\_ne || sptt.is\_backup\_path || sptt.is\_active\_in\_xng || sptt.matching\_live\_xng\_paths || sptt.is\_exempt, ''YNY1Y'', 1, 0 ) ) AS active\_live\_spans\_exempt,

SUM( DECODE( sptt.is\_active\_in\_ne || sptt.is\_backup\_path || sptt.is\_active\_in\_xng || sptt.matching\_live\_xng\_paths || sptt.is\_exempt, ''YNY1N'', 1, 0 ) ) AS active\_live\_spans\_not\_exempt,

SUM( DECODE( sptt.is\_active\_in\_ne || sptt.is\_backup\_path || sptt.is\_active\_in\_xng || sptt.matching\_live\_xng\_paths || sptt.is\_testable, ''YNY1Y'', 1, 0 ) ) AS active\_live\_spans\_testable,

SUM( DECODE( sptt.is\_active\_in\_ne || sptt.is\_backup\_path || sptt.is\_active\_in\_xng || sptt.matching\_live\_xng\_paths || sptt.is\_testable, ''YNY1N'', 1, 0 ) ) AS active\_live\_spans\_untestable

FROM

xng\_reports.domains\_regional\_reporting drr

LEFT OUTER JOIN xng\_reports.switch\_path\_t1\_testability sptt ON drr.domain\_inst\_id = sptt.domain\_inst\_id

LEFT OUTER JOIN xng\_reports.ignored\_ne\_for\_reporting infr ON UPPER( sptt.device\_name ) = infr.device\_name

WHERE

drr.area NOT IN ( ''NNO'',''OSS'' )

GROUP BY rollup (drr.area, drr.region)

';

EXECUTE IMMEDIATE sql\_stmt;

commit;

for rec in T1\_SUMMARY

loop

denominator := rec.ACTIVE\_SPANS\_LIVE\_IN\_XNG + rec.ACTIVE\_SPANS\_NOT\_FOUND\_IN\_XNG - rec.ACTIVE\_LIVE\_SPANS\_EXEMPT;

per\_comp := 0 ;

if(denominator <> 0) then

per\_comp := round(rec.ACTIVE\_LIVE\_SPANS\_TESTABLE/denominator, 4)\*100;

end if;

if (rec.area is null and rec.region is null) then

sql\_stmt := ' update '|| SWITCH\_PATH\_T1\_SUMMARY || ' s set s.PERCENTAGE\_T1\_TESTABLE = ' || per\_comp || ', s.ACTIVE\_REPORTED = '|| denominator ||' where s.area is null and region is null';

elsif(rec.region is null) then

sql\_stmt := ' update '|| SWITCH\_PATH\_T1\_SUMMARY || ' s set s.PERCENTAGE\_T1\_TESTABLE = ' || per\_comp || ', s.ACTIVE\_REPORTED = '|| denominator ||' where s.area = '''|| rec.AREA || ''' and s.region is null ';

else

sql\_stmt := ' update '|| SWITCH\_PATH\_T1\_SUMMARY || ' s set s.PERCENTAGE\_T1\_TESTABLE = ' || per\_comp || ', s.ACTIVE\_REPORTED = '|| denominator ||' where s.area = '''|| rec.AREA || ''' and s.region = '''

|| rec.region || '''' ;

end if;

dbms\_output.put\_line(sql\_stmt);

EXECUTE IMMEDIATE sql\_stmt;

end loop;

commit;

update SWITCH\_PATH\_T1\_SUMMARY t1 set T1.PERCENTAGE\_T1\_TESTABLE = 100

where T1.PERCENTAGE\_T1\_TESTABLE=0

and T1.ACTIVE\_NE\_SPANS = T1.ACTIVE\_LIVE\_SPANS\_EXEMPT

;

commit;

end load\_t1\_summary;

END T1\_Summary;

/

--------------------------------------------------------

-- DDL for Package Body TDM\_EBH\_30\_REPORT

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."TDM\_EBH\_30\_REPORT"

AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: TDM\_EBH\_30\_REPORT

PURPOSE:

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 3/8/2011 1. Created this package body.

2,0 6/22/2011 2. Production Release

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PROCEDURE store\_ebh\_data

IS

sqlStmt VARCHAR2 (32767);

table\_name VARCHAR2 (30) := 'ebh\_ckts';

BEGIN

EXECUTE IMMEDIATE 'truncate table ' || table\_name ||' reuse storage';

sqlstmt :=

'

insert into ' || table\_name

|| '

WITH

SEG\_BILL\_DATE as

(

select cas.circ\_inst\_id inst,

to\_date( cas.attr\_value , ''DD-Mon-YYYY'') val

from

vzwnet.circ\_attr\_settings cas,

vzwnet.val\_attr\_name van

where cas.val\_attr\_inst\_id = van.val\_attr\_inst\_id

and van.group\_name = ''Segment Information''

and van.attr\_name = ''Segment Bill Date''

and CAS.ATTR\_VALUE like ''\_\_-\_\_\_-\_\_\_\_''

)

-- old

select dlr.domain\_inst\_id

, ci.Z\_SITE\_ID

, ci.circ\_inst\_id

-- , case when ci.in\_service is null then bxa.IN\_SERVICE\_DATE else ci.in\_service end in\_service

, case when ci.in\_service is null then bxa.IN\_SERVICE\_DATE

when bxa.IN\_SERVICE\_DATE is null then sd.val

else ci.in\_service end in\_service

, ci.RECUR\_COSTS mrc

, CI.CIRC\_HUM\_ID

from vzwnet.circ\_inst ci

left outer join SEG\_BILL\_DATE sd

on sd.inst = CI.CIRC\_INST\_ID

join vzwnet.circ\_domain\_map cdm

on cdm.circ\_INST\_ID = ci.circ\_INST\_ID

join domains\_leaf\_reporting dlr

on dlr.domain\_inst\_id = cdm.DOMAIN\_INST\_ID

left outer join btp\_xng\_audit bxa

on bxa.XNG\_CIRC\_INST\_ID = ci.CIRC\_INST\_ID

join vzwnet.site\_inst si

on CI.Z\_SITE\_ID = SI.SITE\_INST\_ID

and SI.NUM in (''CELL'')

where ci.BANDWIDTH in (''50 Mbps'', ''100 Mbps'', ''150 Mbps'', ''300 Mbps'', ''400 Mbps'')

and vendor not like ''%VERIZON%WIRELESS%''

and vendor not like ''%VZW%''

'

;

EXECUTE IMMEDIATE sqlStmt;

COMMIT;

END;

PROCEDURE store\_TDM\_data

IS

sqlStmt VARCHAR2 (32767);

table\_name VARCHAR2 (30) := 'tdm\_ckts\_to\_ebh\_sites';

BEGIN

EXECUTE IMMEDIATE 'truncate table ' || table\_name ||' reuse storage';

sqlstmt :=

'

insert into ' || table\_name

|| '

WITH

EBH\_SITE\_W\_INS\_DATE as

(

select distinct domain\_inst\_id

, si.Z\_SITE\_ID

, min (in\_service) in\_service

from EBH\_CKTS si

where in\_service is not null

group by domain\_inst\_id, si.Z\_SITE\_ID

)

select domain\_inst\_id

, ci.circ\_inst\_id

, si.z\_site\_id

, status

, ci.RECUR\_COSTS mrc

, si.in\_service ebh\_in\_service

from vzwnet.circ\_inst ci

join EBH\_SITE\_W\_INS\_DATE si

on si.Z\_SITE\_ID = ci.Z\_SITE\_ID

where si.in\_service < trunc(sysdate - 30)

AND ci.BANDWIDTH in (''DS1'', ''DS3'')

AND CI.STATUS not in (''Decommissioned'', ''Cancelled'')

and vendor not like ''%VERIZON%WIRELESS%''

and vendor not like ''%VZW%''

';

EXECUTE IMMEDIATE sqlStmt;

COMMIT;

sqlstmt :=

'

insert into ' || table\_name

|| '

WITH

EBH\_SITE\_WO\_INS\_DATE as

(

select distinct domain\_inst\_id

, si.Z\_SITE\_ID

from EBH\_CKTS si

where in\_service is null

group by domain\_inst\_id, si.Z\_SITE\_ID

)

select domain\_inst\_id

, ci.circ\_inst\_id

, si.z\_site\_id

, status

, ci.RECUR\_COSTS mrc

, null ebh\_in\_service

from vzwnet.circ\_inst ci

join EBH\_SITE\_WO\_INS\_DATE si

on si.Z\_SITE\_ID = ci.Z\_SITE\_ID

where ci.BANDWIDTH in (''DS1'', ''DS3'')

AND CI.STATUS not in (''Decommissioned'', ''Cancelled'')

and vendor not like ''%VERIZON%WIRELESS%''

and vendor not like ''%VZW%''

';

EXECUTE IMMEDIATE sqlStmt;

COMMIT;

END;

PROCEDURE gen\_Reg\_Summary

IS

BEGIN

gen\_ebh\_Reg\_Summ ();

gen\_tdm\_w\_ins\_date\_Reg\_Summ ();

gen\_tdm\_wo\_ins\_date\_Reg\_Summ ();

END;

PROCEDURE gen\_ebh\_Reg\_Summ

IS

sqlStmt VARCHAR2 (32767);

table\_name VARCHAR2 (30) := 'ebh\_tdm\_reg\_summ';

CURSOR ec\_sites

IS

WITH EBH\_SITES AS -- all ebh enabled sites

(SELECT DISTINCT domain\_inst\_id, z\_site\_ID

FROM EBH\_CKTS

WHERE z\_site\_id IS NOT NULL)

SELECT area, region, COUNT (ec\_sites.z\_site\_id) cnt\_ebh\_sites

FROM domains\_leaf\_reporting dlr

JOIN

EBH\_SITES ec\_sites

ON ec\_sites.domain\_inst\_id = dlr.DOMAIN\_INST\_ID

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR ec\_cost

IS

WITH EBH\_CKTS\_COST AS -- EBH ckts terminated on sites

(SELECT domain\_inst\_id, mrc FROM EBH\_CKTS)

SELECT area, region, SUM (eckts.mrc) ebh\_costs

FROM domains\_leaf\_reporting dlr

JOIN

EBH\_CKTS\_COST eckts

ON eckts.domain\_inst\_id = dlr.DOMAIN\_INST\_ID

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR ec\_n\_termination

IS

WITH EBH\_CKTS\_W\_TERM AS -- EBH ckts terminated on sites

(SELECT domain\_inst\_id, circ\_inst\_ID

FROM EBH\_CKTS

WHERE z\_site\_id IS NULL)

SELECT area,

region,

COUNT (t\_eckts.circ\_inst\_id) ebh\_ckts\_n\_terminated

FROM domains\_leaf\_reporting dlr

JOIN

EBH\_CKTS\_W\_TERM t\_eckts

ON t\_eckts.domain\_inst\_id = dlr.DOMAIN\_INST\_ID

AND t\_eckts.circ\_inst\_id IS NOT NULL

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR ec\_n\_ins\_date

IS

WITH EBH\_CKTS\_WO\_INS\_DATE AS -- EBH ckts no IN\_SERVICE date

(SELECT DISTINCT domain\_inst\_id, circ\_inst\_id

FROM EBH\_CKTS

WHERE in\_service IS NULL)

SELECT area,

region,

COUNT (n\_ins\_eckts.circ\_inst\_id) ebh\_ckts\_n\_INS\_date

FROM domains\_leaf\_reporting dlr

LEFT OUTER JOIN

EBH\_CKTS\_WO\_INS\_DATE n\_ins\_eckts

ON n\_ins\_eckts.domain\_inst\_id = dlr.DOMAIN\_INST\_ID

AND n\_ins\_eckts.circ\_inst\_id IS NOT NULL

GROUP BY ROLLUP (dlr.area, dlr.region);

BEGIN

EXECUTE IMMEDIATE 'truncate table ' || table\_name ||' reuse storage';

sqlStmt :=

'

insert into ' || table\_name

|| '

select area, region

, count(eckts.circ\_inst\_id) ebh\_ckts

, 0 ebh\_cost

, 0 ebh\_ckts\_n\_terminated

, 0 ebh\_ckts\_n\_INS\_date

, 0 ebh\_sites

, 0 cnt\_t\_w\_ins\_pd\_ckts

, 0 cost\_t\_w\_ins\_pd\_ckts

, 0 cnt\_t\_w\_ins\_n\_disco\_n\_pd\_ckts

, 0 cost\_t\_w\_ins\_n\_disco\_n\_pd\_ckts

, 0 cnt\_t\_w\_ins\_n\_disco\_ckts

, 0 cost\_t\_w\_ins\_n\_disco\_ckts

, 0 cnt\_t\_wo\_ins\_pd\_ckts

, 0 cost\_t\_wo\_ins\_pd\_ckts

, 0 cnt\_t\_wo\_ins\_n\_disco\_n\_pd\_ckts

, 0 cost\_t\_wo\_ins\_n\_dis\_n\_pd\_ckts

, 0 cnt\_t\_wo\_ins\_n\_disco\_ckts

, 0 cost\_tdm\_n\_disco\_ckts

from domains\_leaf\_reporting dlr

left outer join EBH\_CKTS eckts

on eckts.domain\_inst\_id = dlr.DOMAIN\_INST\_ID

where region <> ''OSS''

group by rollup (dlr.area, dlr.region)

order by dlr.area, dlr.region'

;

EXECUTE IMMEDIATE sqlStmt;

FOR cur IN ec\_sites

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET ebh\_sites = cur.cnt\_ebh\_sites

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN ec\_cost

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET ebh\_cost = cur.ebh\_costs

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN ec\_n\_termination

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET ebh\_ckts\_n\_terminated = cur.ebh\_ckts\_n\_terminated

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN ec\_n\_ins\_date

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET ebh\_ckts\_n\_ins\_date = cur.ebh\_ckts\_n\_ins\_date

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

END;

PROCEDURE gen\_tdm\_w\_ins\_date\_Reg\_Summ

IS

sqlStmt VARCHAR2 (32767);

table\_name VARCHAR2 (30) := 'ebh\_tdm\_reg\_summ';

CURSOR tdm\_n\_pend\_disco\_ckts

IS

WITH N\_PD\_CKTS

AS (SELECT domain\_inst\_id, circ\_inst\_id

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NOT NULL

AND status <> 'Pending Disconnect')

SELECT area, region, COUNT (circ\_inst\_id) cnt\_tdm\_n\_pd\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

N\_PD\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_n\_pend\_disco\_cost

IS

WITH N\_PD\_CKTS

AS (SELECT domain\_inst\_id, mrc

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NOT NULL

AND status <> 'Pending Disconnect')

SELECT area, region, SUM (nvl(mrc,0)) cost\_tdm\_n\_pd\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

N\_PD\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_pend\_disco\_ckts

IS

WITH PD\_CKTS

AS (SELECT domain\_inst\_id, circ\_inst\_id

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NOT NULL

AND status = 'Pending Disconnect')

SELECT area, region, COUNT (circ\_inst\_id) cnt\_tdm\_pd\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

PD\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_pend\_disco\_cost

IS

WITH PD\_CKTS

AS (SELECT domain\_inst\_id, mrc

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NOT NULL

AND status = 'Pending Disconnect')

SELECT area, region, SUM (nvl(mrc,0)) cost\_tdm\_pd\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

PD\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_n\_disco\_ckts

IS

WITH DISCO\_CKTS AS (SELECT domain\_inst\_id, circ\_inst\_id

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NOT NULL)

SELECT area, region, COUNT (circ\_inst\_id) cnt\_tdm\_n\_disco\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

DISCO\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_n\_disco\_cost

IS

WITH DISCO\_CKTS AS (SELECT domain\_inst\_id, mrc

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NOT NULL)

SELECT area, region, SUM (nvl(mrc,0)) cost\_tdm\_n\_disco\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

DISCO\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

BEGIN

FOR cur IN tdm\_pend\_disco\_ckts

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cnt\_t\_pd\_ckts = cur.cnt\_tdm\_pd\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_pend\_disco\_cost

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cost\_t\_pd\_ckts = cur.cost\_tdm\_pd\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_n\_pend\_disco\_ckts

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cnt\_t\_n\_disco\_n\_pd\_ckts = cur.cnt\_tdm\_n\_pd\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_n\_pend\_disco\_cost

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cost\_t\_n\_disco\_n\_pd\_ckts = cur.cost\_tdm\_n\_pd\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_n\_disco\_ckts

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cnt\_t\_n\_disco\_ckts = cur.cnt\_tdm\_n\_disco\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_n\_disco\_cost

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cost\_t\_n\_disco\_ckts = cur.cost\_tdm\_n\_disco\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

END gen\_tdm\_w\_ins\_date\_Reg\_Summ;

PROCEDURE gen\_tdm\_wo\_ins\_date\_Reg\_Summ

IS

sqlStmt VARCHAR2 (32767);

table\_name VARCHAR2 (30) := 'ebh\_tdm\_reg\_summ';

CURSOR tdm\_n\_pend\_disco\_ckts

IS

WITH N\_PD\_CKTS

AS (SELECT domain\_inst\_id, circ\_inst\_id

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NULL

AND status <> 'Pending Disconnect')

SELECT area, region, COUNT (circ\_inst\_id) cnt\_tdm\_n\_pd\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

N\_PD\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_n\_pend\_disco\_cost

IS

WITH N\_PD\_CKTS

AS (SELECT domain\_inst\_id, mrc

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NULL

AND status <> 'Pending Disconnect')

SELECT area, region, SUM (nvl(mrc,0)) cost\_tdm\_n\_pd\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

N\_PD\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_pend\_disco\_ckts

IS

WITH PD\_CKTS

AS (SELECT domain\_inst\_id, circ\_inst\_id

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NULL

AND status = 'Pending Disconnect')

SELECT area, region, COUNT (circ\_inst\_id) cnt\_tdm\_pd\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

PD\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_pend\_disco\_cost

IS

WITH PD\_CKTS

AS (SELECT domain\_inst\_id, mrc

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NULL

AND status = 'Pending Disconnect')

SELECT area, region, SUM (nvl(mrc,0)) cost\_tdm\_pd\_ckts

-- case when SUM (mrc) is null then

-- 0

-- else

-- SUM (mrc)

-- end as cost\_tdm\_pd\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

PD\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_n\_disco\_ckts

IS

WITH DISCO\_CKTS AS (SELECT domain\_inst\_id, circ\_inst\_id

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NULL)

SELECT area, region, COUNT (circ\_inst\_id) cnt\_tdm\_n\_disco\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

DISCO\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

CURSOR tdm\_n\_disco\_cost

IS

WITH DISCO\_CKTS AS (SELECT domain\_inst\_id, mrc

FROM tdm\_ckts\_to\_ebh\_sites t

WHERE ebh\_in\_service IS NULL)

SELECT area, region, SUM (nvl(mrc,0)) cost\_tdm\_n\_disco\_ckts

FROM domains\_leaf\_reporting dlr

JOIN

DISCO\_CKTS dc

ON dc.domain\_inst\_id = dlr.domain\_inst\_id

GROUP BY ROLLUP (dlr.area, dlr.region);

BEGIN

FOR cur IN tdm\_pend\_disco\_ckts

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cnt\_t\_wo\_ins\_pd\_ckts = cur.cnt\_tdm\_pd\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_pend\_disco\_cost

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cost\_t\_wo\_ins\_pd\_ckts = cur.cost\_tdm\_pd\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_n\_pend\_disco\_ckts

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cnt\_t\_wo\_ins\_n\_disco\_n\_pd\_ckts = cur.cnt\_tdm\_n\_pd\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_n\_pend\_disco\_cost

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cost\_t\_wo\_ins\_n\_dis\_n\_pd\_ckts = cur.cost\_tdm\_n\_pd\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_n\_disco\_ckts

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cnt\_t\_wo\_ins\_n\_disco\_ckts = cur.cnt\_tdm\_n\_disco\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

FOR cur IN tdm\_n\_disco\_cost

LOOP

UPDATE ebh\_tdm\_reg\_summ summ

SET cost\_t\_wo\_ins\_n\_disco\_ckts = cur.cost\_tdm\_n\_disco\_ckts

WHERE summ.area IS NULL AND cur.area IS NULL

OR ( summ.area = cur.area

AND summ.region IS NULL

AND cur.region IS NULL)

OR (summ.region = cur.region);

END LOOP;

COMMIT;

END gen\_tdm\_wo\_ins\_date\_Reg\_Summ;

PROCEDURE do\_all

IS

BEGIN

watchdog.updateprocessstart ('TDM\_EBH\_30\_AUDIT');

store\_ebh\_data;

store\_tdm\_data;

gen\_Reg\_Summary;

watchdog.updateprocessend ('TDM\_EBH\_30\_AUDIT', 'STATUS\_SUCCESS', 'Y');

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line (SUBSTR ( 'Error in move\_csr\_data\_from\_wk(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror ('TDM\_EBH\_30\_AUDIT',

4000,

SUBSTR ( 'Error in TDM\_Report.do\_all(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend ('TDM\_EBH\_30\_AUDIT', 'STATUS\_FAILURE', 'N');

--RAISE;

END do\_all;

END TDM\_EBH\_30\_REPORT;

/

--------------------------------------------------------

-- DDL for Package Body VPI\_GI\_AUDIT\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."VPI\_GI\_AUDIT\_PKG" AS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Purpose: Compare VPI and Granite data

\* Process: Compare VPI and Granite data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* procedure that runs audit

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure runAudit is

BEGIN

processName := 'VPI\_GI\_AUDIT';

-- Tell watchDdog the process has started

BEGIN

watchdog.updateprocessstart (processName);

BEGIN

vpiGiMatches();

giOnly();

vpiGiSitesUnique();

-- tell watchDog process completed successfully

watchdog.updateprocessend (processName, 'STATUS\_SUCCESS', 'Y');

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: VPI\_GI\_AUDIT\_PKG(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

raise;

END;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr('Error: VPI\_GI\_AUDIT\_PKG(): '||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

watchdog.updateprocessend (processName, 'STATUS\_FAILURE', 'N');

END;

END runAudit;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Load VPI data with GI matches

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure vpiGiMatches is

v\_cnt number;

v\_process\_name xng\_reports.all\_processes.process\_name%type;

BEGIN

v\_cnt := 0;

XNG\_REPORTS.TRUNCATE\_XNG\_REPORTS\_TABLE('VPI\_GI\_AUDIT\_WK');

insert into xng\_reports.vpi\_gi\_audit\_wk

select \* from

(with pslc as (select si.site\_inst\_id, max(sas.attr\_value) ps\_loc\_code\_uda

from vzwnet.site\_inst si,

vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

where si.site\_inst\_id = sas.site\_inst\_id

and van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id

and van.group\_name = 'Site Information'

and van.attr\_name = 'PeopleSoft Location Code #'

and sas.attr\_value is not null

and sas.attr\_value <> '0'

group by si.site\_inst\_id, si.site\_hum\_id),

locat as (select si.site\_inst\_id, max(sas.attr\_value) gi\_loc\_cat

from vzwnet.site\_inst si,

vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

where si.site\_inst\_id = sas.site\_inst\_id

and van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id

and van.group\_name = 'Site Realignment'

and van.attr\_name = 'Location Category'

and sas.attr\_value is not null

and sas.attr\_value <> '0'

group by si.site\_inst\_id, si.site\_hum\_id),

srt as (select si.site\_inst\_id, max(sas.attr\_value) gi\_ring\_type

from vzwnet.site\_inst si,

vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

where si.site\_inst\_id = sas.site\_inst\_id

and van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id

and van.group\_name = 'Siterra'

and van.attr\_name = 'Search Ring Type'

and sas.attr\_value is not null

and sas.attr\_value <> '0'

group by si.site\_inst\_id, si.site\_hum\_id),

vzw\_sites as (select distinct territory, market\_territory, sub\_market, site\_inst\_id, site\_hum\_id,

latitude, longitude, city, gi\_site\_type, state\_prov, county, post\_code\_1, ps\_loc\_code\_uda,

base\_num, status, nauru\_site\_id, gi\_loc\_cat

from (select nvl(drr.territory,'Unknown') territory, nvl(drr.market\_territory, 'Unknown') market\_territory,

nvl(drr.sub\_market,'Unknown') sub\_market, si.site\_inst\_id, site\_hum\_id,

latitude, longitude, city, state\_prov, county, post\_code\_1, base\_num,

num gi\_site\_type, si.status, pslc.ps\_loc\_code\_uda, nxsm.nauru\_site\_id, nxsm.nwf\_id,

nvl(locat.gi\_loc\_cat,srt.gi\_ring\_type) gi\_loc\_cat

from vzwnet.site\_inst si, vzwnet.site\_domain\_map sdm,

xng\_reports.domains\_regional\_reporting\_new drr, pslc, locat, srt,

xng\_apps.nauru\_xng\_site\_map nxsm

where si.site\_inst\_id = sdm.site\_inst\_id (+)

and sdm.domain\_inst\_id = drr.domain\_inst\_id (+)

and drr.territory <> 'OSS'

and si.site\_inst\_id = pslc.site\_inst\_id (+)

and si.site\_inst\_id = locat.site\_inst\_id (+)

and si.site\_inst\_id = srt.site\_inst\_id (+)

and si.site\_inst\_id = nxsm.site\_inst\_id(+))),

vpi as (select territory, market, sub\_market, morphology, cma\_no, cma\_name, uace, uace\_name, nwid, enodeb\_grp, site\_name,

county, state, latitude, longitude, market\_name, mkt\_id, pslc, s\_c\_count, top\_cma\_uace, extract\_date

from xng\_reports.vpi\_data),

matches as

(select distinct vpi.territory, vpi.market, vpi.sub\_market, vpi.morphology, vpi.cma\_no, vpi.cma\_name, vpi.uace,

vpi.uace\_name, vpi.nwid, vpi.enodeb\_grp, vpi.site\_name, vpi.county, vpi.state, vpi.latitude,

vpi.longitude, vpi.market\_name, vpi.mkt\_id, vpi.pslc, vpi.s\_c\_count, vpi.top\_cma\_uace, vpi.extract\_date, vs.territory gi\_territory, vs.market\_territory gi\_market\_territory,

vs.sub\_market gi\_sub\_market, vs.site\_inst\_id gi\_site\_id, vs.site\_hum\_id gi\_site\_name, vs.ps\_loc\_code\_uda gi\_pslc,

vs.county gi\_county, vs.state\_prov gi\_state\_prov, vs.gi\_site\_type, vs.status gi\_status, vs.latitude gi\_lat,

vs.longitude gi\_long, nauru\_site\_id gi\_nauru\_id, gi\_loc\_cat

from vpi, vzw\_sites vs

where vpi.nwid is not null and

vs.nauru\_site\_id is not null and

upper(vpi.nwid) = upper(vs.nauru\_site\_id)),

vpi\_all as

(select match\_type, territory, market, sub\_market, morphology, cma\_no, cma\_name, uace, uace\_name, nwid, enodeb\_grp, site\_name,

county, state, latitude, longitude, market\_name, mkt\_id, pslc, s\_c\_count, top\_cma\_uace, extract\_date,

gi\_site\_id, gi\_site\_name, gi\_pslc, gi\_county, gi\_state\_prov, gi\_site\_type, gi\_status, gi\_lat, gi\_long, gi\_nauru\_id, gi\_loc\_cat,

row\_number() over (partition by territory, market, sub\_market, territory, market, sub\_market, morphology, cma\_no, cma\_name, uace, uace\_name, nwid, enodeb\_grp, site\_name,

county, state, latitude, longitude, market\_name, mkt\_id, pslc, s\_c\_count, top\_cma\_uace

order by match\_type) as rankr

from (select distinct territory, market, sub\_market, morphology, cma\_no, cma\_name, uace, uace\_name, nwid, enodeb\_grp, site\_name,

county, state, latitude, longitude, market\_name, mkt\_id, pslc, s\_c\_count, top\_cma\_uace, extract\_date, gi\_site\_id, gi\_site\_name, gi\_pslc, gi\_county,

gi\_state\_prov, gi\_site\_type, gi\_status, gi\_lat, gi\_long, gi\_nauru\_id, gi\_loc\_cat, 1 as match\_type

from matches

union

select territory, market, sub\_market, morphology, cma\_no, cma\_name, uace, uace\_name, nwid, enodeb\_grp, site\_name,

county, state, latitude, longitude, market\_name, mkt\_id, pslc, s\_c\_count, top\_cma\_uace, extract\_date, null, null, null, null, null, null, null, null,

null, null, null, 2 as match\_type

from vpi vu

where not exists ( select 1

from matches m

where m.territory = vu.territory

and m.market = vu.market

and m.sub\_market = vu.sub\_market

and m.morphology = vu.morphology

and m.cma\_no = vu.cma\_no

and m.cma\_name = vu.cma\_name

and m.uace = vu.uace

and m.uace\_name = vu.uace\_name

and m.nwid = vu.nwid

and m.enodeb\_grp = vu.enodeb\_grp

and m.site\_name = vu.site\_name

and m.county = vu.county

and m.state = vu.state

and m.latitude = vu.latitude

and m.longitude = vu.longitude

and m.market\_name = vu.market\_name

and m.mkt\_id = vu.mkt\_id

and m.pslc = vu.pslc

and m.s\_c\_count = vu.s\_c\_count

and m.top\_cma\_uace = vu.top\_cma\_uace

and m.extract\_date = vu.extract\_date))),

df\_segs as (select site\_id, gi\_df\_vendor, min(gi\_df\_status) gi\_df\_status

from (select df.site\_id, df.gi\_df\_vendor,

case

when vds.root\_status = 'P' then 'Planned/Awarded'

else 'Active'

end gi\_df\_status

from (select z\_site\_id site\_id, status, vendor gi\_df\_vendor

from vzwnet.circ\_inst

where type = 'DARK FIBER'

union

select a\_site\_id site\_id, status, vendor gi\_df\_vendor

from vzwnet.circ\_inst

where type = 'DARK FIBER') df,

vzwnet.val\_deploy\_status vds

where df.status = vds.status\_name

and vds.root\_status in ('A','P'))

group by site\_id, gi\_df\_vendor),

df\_segs\_unq as (select site\_id, gi\_df\_vendor, gi\_df\_status from df\_segs where gi\_df\_status = 'Active'

union

select site\_id, gi\_df\_vendor, gi\_df\_status from df\_segs dp where gi\_df\_status = 'Planned/Awarded'

and not exists (select 1

from df\_segs da

where gi\_df\_status = 'Active'

and da.site\_id = dp.site\_id)),

mw\_segs as (select site\_id, gi\_mw\_vendor, min(gi\_mw\_status) gi\_mw\_status

from (select df.site\_id, df.gi\_mw\_vendor,

case

when vds.root\_status = 'P' then 'Planned/Awarded'

else 'Active'

end gi\_mw\_status

from (select z\_site\_id site\_id, status, vendor gi\_mw\_vendor

from vzwnet.circ\_inst

where type = 'MICROWAVE'

union

select a\_site\_id site\_id, status, vendor gi\_mw\_vendor

from vzwnet.circ\_inst

where type = 'MICROWAVE') df,

vzwnet.val\_deploy\_status vds

where df.status = vds.status\_name

and vds.root\_status in ('A','P'))

group by site\_id, gi\_mw\_vendor),

mw\_segs\_unq as (select site\_id, gi\_mw\_vendor, gi\_mw\_status from mw\_segs where gi\_mw\_status = 'Active'

union

select site\_id, gi\_mw\_vendor, gi\_mw\_status from mw\_segs dp where gi\_mw\_status = 'Planned/Awarded'

and not exists (select 1

from mw\_segs da

where gi\_mw\_status = 'Active'

and da.site\_id = dp.site\_id)),

ls\_segs as (select site\_id, gi\_ls\_vendor, min(gi\_ls\_status) gi\_ls\_status

from (select df.site\_id, df.gi\_ls\_vendor,

case

when vds.root\_status = 'P' then 'Planned/Awarded'

else 'Active'

end gi\_ls\_status

from (select z\_site\_id site\_id, status, vendor gi\_ls\_vendor

from vzwnet.circ\_inst

where type = 'LEASED SERVICE'

union

select a\_site\_id site\_id, status, vendor gi\_ls\_vendor

from vzwnet.circ\_inst

where type = 'LEASED SERVICE') df,

vzwnet.val\_deploy\_status vds

where df.status = vds.status\_name

and vds.root\_status in ('A','P'))

group by site\_id, gi\_ls\_vendor),

ls\_segs\_unq as (select site\_id, gi\_ls\_vendor, gi\_ls\_status from ls\_segs where gi\_ls\_status = 'Active'

union

select site\_id, gi\_ls\_vendor, gi\_ls\_status from ls\_segs dp where gi\_ls\_status = 'Planned/Awarded'

and not exists (select 1

from ls\_segs da

where gi\_ls\_status = 'Active'

and da.site\_id = dp.site\_id))

select distinct territory, market, sub\_market, morphology, cma\_no, cma\_name, uace, uace\_name, nwid, enodeb\_grp, site\_name,

county, state, latitude, longitude, market\_name, mkt\_id, pslc, s\_c\_count, top\_cma\_uace, extract\_date, gi\_site\_id,

gi\_site\_name, gi\_site\_type, gi\_status, gi\_pslc, gi\_county, gi\_state\_prov, gi\_lat, gi\_long, gi\_nauru\_id,

gi\_loc\_cat, trunc(sysdate) gi\_extract\_date, df.gi\_df\_vendor, df.gi\_df\_status, mw.gi\_mw\_vendor, mw.gi\_mw\_status,

ls.gi\_ls\_vendor, ls.gi\_ls\_status

from vpi\_all va, df\_segs\_unq df, mw\_segs\_unq mw, ls\_segs\_unq ls

where va.gi\_site\_id = df.site\_id (+)

and va.gi\_site\_id = mw.site\_id (+)

and va.gi\_site\_id = ls.site\_id (+)

and rankr = 1);

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

xng\_reports.WATCHDOG.logerror(v\_process\_name,4000,SubStr('Error in xng\_reports.VPI\_GI\_AUDIT\_PKG.vpiGiAudit stored procedure: '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

xng\_reports.WATCHDOG.updateprocessend('VPI\_GI\_AUDIT','STATUS\_FAILURE','N');

--raise;

END vpiGiMatches;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Load GI sites that do not VPI data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure giOnly is

v\_cnt number;

v\_process\_name xng\_reports.all\_processes.process\_name%type;

BEGIN

v\_cnt := 0;

insert into xng\_reports.vpi\_gi\_audit\_wk

select \* from (

with pslc as (select si.site\_inst\_id, max(sas.attr\_value) ps\_loc\_code\_uda

from vzwnet.site\_inst si,

vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

where si.site\_inst\_id = sas.site\_inst\_id

and van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id

and van.group\_name = 'Site Information'

and van.attr\_name = 'PeopleSoft Location Code #'

and sas.attr\_value is not null

and sas.attr\_value <> '0'

group by si.site\_inst\_id, si.site\_hum\_id),

locat as (select si.site\_inst\_id, max(sas.attr\_value) gi\_loc\_cat

from vzwnet.site\_inst si,

vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

where si.site\_inst\_id = sas.site\_inst\_id

and van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id

and van.group\_name = 'Site Realignment'

and van.attr\_name = 'Location Category'

and sas.attr\_value is not null

and sas.attr\_value <> '0'

group by si.site\_inst\_id, si.site\_hum\_id),

srt as (select si.site\_inst\_id, max(sas.attr\_value) gi\_ring\_type

from vzwnet.site\_inst si,

vzwnet.site\_attr\_settings sas,

vzwnet.val\_attr\_name van

where si.site\_inst\_id = sas.site\_inst\_id

and van.val\_attr\_inst\_id = sas.val\_attr\_inst\_id

and van.group\_name = 'Siterra'

and van.attr\_name = 'Search Ring Type'

and sas.attr\_value is not null

and sas.attr\_value <> '0'

group by si.site\_inst\_id, si.site\_hum\_id),

vzw\_sites as (select distinct territory, market\_territory, sub\_market, site\_inst\_id, site\_hum\_id,

latitude, longitude, city, gi\_site\_type, state\_prov, county, post\_code\_1, ps\_loc\_code\_uda,

base\_num, status, nauru\_site\_id, gi\_loc\_cat

from (select nvl(drr.territory,'Unknown') territory, nvl(drr.market\_territory, 'Unknown') market\_territory,

nvl(drr.sub\_market,'Unknown') sub\_market, si.site\_inst\_id, site\_hum\_id,

latitude, longitude, city, state\_prov, county, post\_code\_1, base\_num,

num gi\_site\_type, si.status, pslc.ps\_loc\_code\_uda, nxsm.nauru\_site\_id, nxsm.nwf\_id,

nvl(locat.gi\_loc\_cat,srt.gi\_ring\_type) gi\_loc\_cat

from vzwnet.site\_inst si, vzwnet.site\_domain\_map sdm,

xng\_reports.domains\_regional\_reporting\_new drr, pslc, locat, srt,

xng\_apps.nauru\_xng\_site\_map nxsm

where si.site\_inst\_id = sdm.site\_inst\_id (+)

and sdm.domain\_inst\_id = drr.domain\_inst\_id (+)

and drr.territory <> 'OSS'

and si.site\_inst\_id = pslc.site\_inst\_id (+)

and si.site\_inst\_id = locat.site\_inst\_id (+)

and si.site\_inst\_id = srt.site\_inst\_id (+)

and si.site\_inst\_id = nxsm.site\_inst\_id(+))),

df\_segs as (select site\_id, gi\_df\_vendor, min(gi\_df\_status) gi\_df\_status

from (select df.site\_id, df.gi\_df\_vendor,

case

when vds.root\_status = 'P' then 'Planned/Awarded'

else 'Active'

end gi\_df\_status

from (select z\_site\_id site\_id, status, vendor gi\_df\_vendor

from vzwnet.circ\_inst

where type = 'DARK FIBER'

union

select a\_site\_id site\_id, status, vendor gi\_df\_vendor

from vzwnet.circ\_inst

where type = 'DARK FIBER') df,

vzwnet.val\_deploy\_status vds

where df.status = vds.status\_name

and vds.root\_status in ('A','P'))

group by site\_id, gi\_df\_vendor),

df\_segs\_unq as (select site\_id, gi\_df\_vendor, gi\_df\_status from df\_segs where gi\_df\_status = 'Active'

union

select site\_id, gi\_df\_vendor, gi\_df\_status from df\_segs dp where gi\_df\_status = 'Planned/Awarded'

and not exists (select 1

from df\_segs da

where gi\_df\_status = 'Active'

and da.site\_id = dp.site\_id)),

mw\_segs as (select site\_id, gi\_mw\_vendor, min(gi\_mw\_status) gi\_mw\_status

from (select df.site\_id, df.gi\_mw\_vendor,

case

when vds.root\_status = 'P' then 'Planned/Awarded'

else 'Active'

end gi\_mw\_status

from (select z\_site\_id site\_id, status, vendor gi\_mw\_vendor

from vzwnet.circ\_inst

where type = 'MICROWAVE'

union

select a\_site\_id site\_id, status, vendor gi\_mw\_vendor

from vzwnet.circ\_inst

where type = 'MICROWAVE') df,

vzwnet.val\_deploy\_status vds

where df.status = vds.status\_name

and vds.root\_status in ('A','P'))

group by site\_id, gi\_mw\_vendor),

mw\_segs\_unq as (select site\_id, gi\_mw\_vendor, gi\_mw\_status from mw\_segs where gi\_mw\_status = 'Active'

union

select site\_id, gi\_mw\_vendor, gi\_mw\_status from mw\_segs dp where gi\_mw\_status = 'Planned/Awarded'

and not exists (select 1

from mw\_segs da

where gi\_mw\_status = 'Active'

and da.site\_id = dp.site\_id)),

ls\_segs as (select site\_id, gi\_ls\_vendor, min(gi\_ls\_status) gi\_ls\_status

from (select df.site\_id, df.gi\_ls\_vendor,

case

when vds.root\_status = 'P' then 'Planned/Awarded'

else 'Active'

end gi\_ls\_status

from (select z\_site\_id site\_id, status, vendor gi\_ls\_vendor

from vzwnet.circ\_inst

where type = 'LEASED SERVICE'

union

select a\_site\_id site\_id, status, vendor gi\_ls\_vendor

from vzwnet.circ\_inst

where type = 'LEASED SERVICE') df,

vzwnet.val\_deploy\_status vds

where df.status = vds.status\_name

and vds.root\_status in ('A','P'))

group by site\_id, gi\_ls\_vendor),

ls\_segs\_unq as (select site\_id, gi\_ls\_vendor, gi\_ls\_status from ls\_segs where gi\_ls\_status = 'Active'

union

select site\_id, gi\_ls\_vendor, gi\_ls\_status from ls\_segs dp where gi\_ls\_status = 'Planned/Awarded'

and not exists (select 1

from ls\_segs da

where gi\_ls\_status = 'Active'

and da.site\_id = dp.site\_id)),

sites as (select distinct territory, market\_territory, sub\_market, site\_inst\_id, site\_hum\_id,

latitude, longitude, city, gi\_site\_type, state\_prov, county, post\_code\_1, ps\_loc\_code\_uda,

base\_num, status, nauru\_site\_id, gi\_loc\_cat, df.gi\_df\_vendor, df.gi\_df\_status,

mw.gi\_mw\_vendor, mw.gi\_mw\_status, ls.gi\_ls\_vendor, ls.gi\_ls\_status

from vzw\_sites vs, df\_segs\_unq df, mw\_segs\_unq mw, ls\_segs\_unq ls

where vs.site\_inst\_id = df.site\_id (+)

and vs.site\_inst\_id = mw.site\_id (+)

and vs.site\_inst\_id = ls.site\_id (+))

select distinct territory, market\_territory,

sub\_market, null as morphology, null as cma\_no, null as cma\_name, null as uace,

null as uace\_name, null as nwid, null as enodeb\_grp, null as site\_name, null as county, null as state, null as latitude,

null as longitude, null as market\_name, null as mkt\_id, null as pslc, null as s\_c\_count, null as top\_cma\_uace, null as extract\_date,

s.site\_inst\_id gi\_site\_id, s.site\_hum\_id gi\_site\_name, s.gi\_site\_type, s.status gi\_status, s.ps\_loc\_code\_uda gi\_pslc,

s.county gi\_county, s.state\_prov gi\_state\_prov, s.latitude gi\_lat, s.longitude gi\_long, s.nauru\_site\_id gi\_nauru\_id,

s.gi\_loc\_cat, trunc(sysdate) as gi\_extract\_date, s.gi\_df\_vendor, s.gi\_df\_status, s.gi\_mw\_vendor, s.gi\_mw\_status,

s.gi\_ls\_vendor, s.gi\_ls\_status

from sites s

where not exists ( select 1

from xng\_reports.vpi\_gi\_audit\_wk m

where m.gi\_site\_id = s.site\_inst\_id)

);

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

xng\_reports.WATCHDOG.logerror(v\_process\_name,4000,SubStr('Error in xng\_reports.VPI\_GI\_AUDIT\_PKG.giOnly stored procedure: '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

xng\_reports.WATCHDOG.updateprocessend('VPI\_GI\_AUDIT','STATUS\_FAILURE','N');

--raise;

END giOnly;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Load VPI GI audit with fields with duplicates combined in semi-column delimited

\* lists so GI sites are unique

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Procedure vpiGiSitesUnique is

v\_cnt number;

v\_process\_name xng\_reports.all\_processes.process\_name%type;

BEGIN

v\_cnt := 0;

XNG\_REPORTS.TRUNCATE\_XNG\_REPORTS\_TABLE('VPI\_GI\_AUDIT\_UNQ\_WK');

insert into xng\_reports.vpi\_gi\_audit\_unq\_wk

with df\_seg\_list as (select gi\_site\_id, rtrim(ltrim(n1||n2||n3||n4||n5||n6||n7||n8||n9||n10)) as gi\_df\_vendors

from (select gi\_site\_id,

max(decode (order\_no,1, gi\_df\_vendor, null)) n1,

max(decode (order\_no,2, '; ' ||gi\_df\_vendor, null)) n2,

max(decode (order\_no,3, '; ' ||gi\_df\_vendor, null)) n3,

max(decode (order\_no,4, '; ' ||gi\_df\_vendor, null)) n4,

max(decode (order\_no,5, '; ' ||gi\_df\_vendor, null)) n5,

max(decode (order\_no,6, '; ' ||gi\_df\_vendor, null)) n6,

max(decode (order\_no,7, '; ' ||gi\_df\_vendor, null)) n7,

max(decode (order\_no,8, '; ' ||gi\_df\_vendor, null)) n8,

max(decode (order\_no,9, '; ' ||gi\_df\_vendor, null)) n9,

max(decode (order\_no,10, '; ' ||gi\_df\_vendor, null)) n10

from (select gi\_site\_id, gi\_df\_vendor, row\_number() over (partition by gi\_site\_id

order by gi\_df\_vendor desc) as order\_no

from xng\_reports.vpi\_gi\_audit\_wk

where gi\_df\_vendor is not null

group by gi\_site\_id, gi\_df\_vendor)

group by gi\_site\_id)),

mw\_seg\_list as (select gi\_site\_id, rtrim(ltrim(n1||n2||n3||n4||n5||n6||n7||n8||n9||n10)) as gi\_mw\_vendors

from (select gi\_site\_id,

max(decode (order\_no,1, gi\_mw\_vendor, null)) n1,

max(decode (order\_no,2, '; ' ||gi\_mw\_vendor, null)) n2,

max(decode (order\_no,3, '; ' ||gi\_mw\_vendor, null)) n3,

max(decode (order\_no,4, '; ' ||gi\_mw\_vendor, null)) n4,

max(decode (order\_no,5, '; ' ||gi\_mw\_vendor, null)) n5,

max(decode (order\_no,6, '; ' ||gi\_mw\_vendor, null)) n6,

max(decode (order\_no,7, '; ' ||gi\_mw\_vendor, null)) n7,

max(decode (order\_no,8, '; ' ||gi\_mw\_vendor, null)) n8,

max(decode (order\_no,9, '; ' ||gi\_mw\_vendor, null)) n9,

max(decode (order\_no,10, '; ' ||gi\_mw\_vendor, null)) n10

from (select gi\_site\_id, gi\_mw\_vendor, row\_number() over (partition by gi\_site\_id

order by gi\_mw\_vendor desc) as order\_no

from xng\_reports.vpi\_gi\_audit\_wk

where gi\_mw\_vendor is not null

group by gi\_site\_id, gi\_mw\_vendor)

group by gi\_site\_id)),

ls\_seg\_list as (select gi\_site\_id, rtrim(ltrim(n1||n2||n3||n4||n5||n6||n7||n8||n9||n10)) as gi\_ls\_vendors

from (select gi\_site\_id,

max(decode (order\_no,1, gi\_ls\_vendor, null)) n1,

max(decode (order\_no,2, '; ' ||gi\_ls\_vendor, null)) n2,

max(decode (order\_no,3, '; ' ||gi\_ls\_vendor, null)) n3,

max(decode (order\_no,4, '; ' ||gi\_ls\_vendor, null)) n4,

max(decode (order\_no,5, '; ' ||gi\_ls\_vendor, null)) n5,

max(decode (order\_no,6, '; ' ||gi\_ls\_vendor, null)) n6,

max(decode (order\_no,7, '; ' ||gi\_ls\_vendor, null)) n7,

max(decode (order\_no,8, '; ' ||gi\_ls\_vendor, null)) n8,

max(decode (order\_no,9, '; ' ||gi\_ls\_vendor, null)) n9,

max(decode (order\_no,10, '; ' ||gi\_ls\_vendor, null)) n10

from (select gi\_site\_id, gi\_ls\_vendor, row\_number() over (partition by gi\_site\_id

order by gi\_ls\_vendor desc) as order\_no

from xng\_reports.vpi\_gi\_audit\_wk

where gi\_ls\_vendor is not null

group by gi\_site\_id, gi\_ls\_vendor)

group by gi\_site\_id)),

terr\_list as (select gi\_site\_id, rtrim(ltrim(n1||n2||n3||n4||n5||n6||n7||n8||n9||n10)) as terrs

from (select gi\_site\_id,

max(decode (order\_no,1, territory, null)) n1,

max(decode (order\_no,2, '; ' ||territory, null)) n2,

max(decode (order\_no,3, '; ' ||territory, null)) n3,

max(decode (order\_no,4, '; ' ||territory, null)) n4,

max(decode (order\_no,5, '; ' ||territory, null)) n5,

max(decode (order\_no,6, '; ' ||territory, null)) n6,

max(decode (order\_no,7, '; ' ||territory, null)) n7,

max(decode (order\_no,8, '; ' ||territory, null)) n8,

max(decode (order\_no,9, '; ' ||territory, null)) n9,

max(decode (order\_no,10, '; ' ||territory, null)) n10

from (select gi\_site\_id, territory, row\_number() over (partition by gi\_site\_id

order by territory desc) as order\_no

from xng\_reports.vpi\_gi\_audit\_wk

group by gi\_site\_id, territory)

group by gi\_site\_id)),

mkt\_list as (select gi\_site\_id, rtrim(ltrim(n1||n2||n3||n4||n5||n6||n7||n8||n9||n10)) as markets

from (select gi\_site\_id,

max(decode (order\_no,1, market, null)) n1,

max(decode (order\_no,2, '; ' ||market, null)) n2,

max(decode (order\_no,3, '; ' ||market, null)) n3,

max(decode (order\_no,4, '; ' ||market, null)) n4,

max(decode (order\_no,5, '; ' ||market, null)) n5,

max(decode (order\_no,6, '; ' ||market, null)) n6,

max(decode (order\_no,7, '; ' ||market, null)) n7,

max(decode (order\_no,8, '; ' ||market, null)) n8,

max(decode (order\_no,9, '; ' ||market, null)) n9,

max(decode (order\_no,10, '; ' ||market, null)) n10

from (select gi\_site\_id, market, row\_number() over (partition by gi\_site\_id

order by market desc) as order\_no

from xng\_reports.vpi\_gi\_audit\_wk

group by gi\_site\_id, market)

group by gi\_site\_id)),

sub\_list as (select gi\_site\_id, rtrim(ltrim(n1||n2||n3||n4||n5||n6||n7||n8||n9||n10)) as sub\_mkts

from (select gi\_site\_id,

max(decode (order\_no,1, sub\_market, null)) n1,

max(decode (order\_no,2, '; ' ||sub\_market, null)) n2,

max(decode (order\_no,3, '; ' ||sub\_market, null)) n3,

max(decode (order\_no,4, '; ' ||sub\_market, null)) n4,

max(decode (order\_no,5, '; ' ||sub\_market, null)) n5,

max(decode (order\_no,6, '; ' ||sub\_market, null)) n6,

max(decode (order\_no,7, '; ' ||sub\_market, null)) n7,

max(decode (order\_no,8, '; ' ||sub\_market, null)) n8,

max(decode (order\_no,9, '; ' ||sub\_market, null)) n9,

max(decode (order\_no,10, '; ' ||sub\_market, null)) n10

from (select gi\_site\_id, sub\_market, row\_number() over (partition by gi\_site\_id

order by sub\_market desc) as order\_no

from xng\_reports.vpi\_gi\_audit\_wk

group by gi\_site\_id, sub\_market)

group by gi\_site\_id))

select distinct nvl(tl.terrs, vum.territory) terrs, nvl(ml.markets, vum.market) markets,

nvl(sl.sub\_mkts, vum.sub\_market) sub\_mkts, morphology,

cma\_no, cma\_name, uace, uace\_name, nwid, enodeb\_grp, site\_name,

county, state, latitude, longitude, market\_name, mkt\_id, pslc, s\_c\_count,

top\_cma\_uace, vpi\_extract\_date, vum.gi\_site\_id, gi\_site\_name, gi\_site\_type, gi\_site\_status,

gi\_loc\_cat, gi\_extract\_date, gi\_df\_vendors, gi\_df\_status, gi\_mw\_vendors, gi\_mw\_status,

gi\_ls\_vendors, gi\_ls\_status

from xng\_reports.vpi\_gi\_audit\_wk vum, df\_seg\_list dslm, mw\_seg\_list mslm, ls\_seg\_list lslm,

terr\_list tl, mkt\_list ml, sub\_list sl

where vum.gi\_site\_id = dslm.gi\_site\_id(+)

and vum.gi\_site\_id = mslm.gi\_site\_id(+)

and vum.gi\_site\_id = lslm.gi\_site\_id(+)

and vum.gi\_site\_id = tl.gi\_site\_id (+)

and vum.gi\_site\_id = ml.gi\_site\_id (+)

and vum.gi\_site\_id = sl.gi\_site\_id(+);

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

xng\_reports.WATCHDOG.logerror(v\_process\_name,4000,SubStr('Error in xng\_reports.VPI\_GI\_AUDIT\_PKG.giOnly stored procedure: '

||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255),'Y');

xng\_reports.WATCHDOG.updateprocessend('VPI\_GI\_AUDIT','STATUS\_FAILURE','N');

--raise;

END vpiGiSitesUnique;

END;

/

--------------------------------------------------------

-- DDL for Package Body VZ\_REPORTS\_UTILS

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."VZ\_REPORTS\_UTILS" AS

pv\_XML\_DISPLAYED\_ALLOWED BOOLEAN := VZ\_CONSTANTS.VS\_DISPLAY\_DATA\_IN\_XML\_ALLOWED;

/\*\*\*\*\*\*\*\*\*\*\*\*\*

$Log: VZ\_REPORTS\_UTILS.PKB,v $

Revision 1.2 2004/03/01 20:55:24 c\_anderj

update after initial testing on EA @ scorpion.ea

\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* FUNCTION authorize return boolean is

begin

owa\_Sec.set\_protection\_realm('WNEA');

return true;

end;

\*/

FUNCTION iff(

tf IN BOOLEAN,

yes IN VARCHAR2,

no IN VARCHAR2 := NULL

) RETURN VARCHAR2

IS

BEGIN

IF ( tf ) THEN

RETURN yes;

ELSE

RETURN no;

END IF;

END iff;

FUNCTION GETNAME(p\_userID varchar2) RETURN varchar2 IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: GETNAME

PURPOSE: gets the full name based on userid

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 08/28/01 Anon 1. Created this function.

PARAMETERS:

INPUT:

OUTPUT:

RETURNED VALUE: NUMBER

CALLED BY:

CALLS:

EXAMPLE USE: NUMBER := getname(nvl(userid, '.'))

ASSUMPTIONS:

LIMITATIONS:

ALGORITHM:

NOTES:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

vs\_fullName varchar2(50);

BEGIN

-- select nvl(FULL\_NAME, p\_userId)

-- into vs\_fullName

-- from xperadmin.user\_inst

-- where user\_name = rtrim(ltrim(p\_userID));

-- RETURN vs\_fullName;

--

--

-- EXCEPTION

-- WHEN NO\_DATA\_FOUND THEN

-- vs\_fullName := p\_userId;

-- RETURN vs\_fullName;

-- WHEN OTHERS THEN

-- vs\_fullName := p\_userId;

RETURN vs\_fullName;

END GETNAME;

FUNCTION fs\_pickMonth return varchar2

-- generate a pick list of months, defaulting to current month.

IS

retBuf long; -- will hold the select options string

vd\_today date;

vd\_nextMonth date;

vs\_monthMON varchar2(3); -- month as JAN

vs\_monthMM varchar2(2); -- month as 01

BEGIN

vd\_today := trunc(sysdate);

vs\_monthMON := to\_char(vd\_today, 'MON'); -- month as JAN

vs\_monthMM := to\_char(vd\_today, 'MM'); -- month as 01

----------------Open form with current month selected ----------------------------------------------

retBuf := htf.formSelectOpen( cname => 'p\_dateMonth', nsize => '1' );

retBuf := retBuf || htf.FORMSELECTOPTION(vs\_monthMON, cselected => 'TRUE', cattributes => ' value='||vs\_monthMM||'');

vd\_nextMonth := vd\_today;

FOR i IN REVERSE 1..11 LOOP

vd\_nextMonth := add\_months(vd\_nextMonth , 1);

vs\_monthMON := to\_char(vd\_nextMonth, 'MON'); -- month as JAN

vs\_monthMM := to\_char(vd\_nextMonth, 'MM'); -- month as 01

retBuf := retBuf || htf.FORMSELECTOPTION(vs\_monthMON, cattributes => ' value='||vs\_monthMM||'');

END LOOP;

retBuf := retBuf || htf.formSelectClose;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

Null;

WHEN OTHERS THEN

Null;

return retBuf ;

END fs\_pickMonth;

procedure embedscript(p\_link varchar2, p\_imageLocation varchar2)

is

BEGIN

htp.print( '<DIV CLASS="jsbrand" ID="jsbrand" STYLE="position:absolute;top:1;visibility:hide;" zIndex="1000" ALIGN="right">' );

htp.anchor( p\_link , htf.img(p\_imageLocation, calt => 'Main Page', cattributes => ' BORDER="0"'), cattributes => ' TARGET="\_top"');

htp.print( '</DIV>' );

htp.print( '<SCRIPT language="Javascript1.2">' );

htp.print( '<!-- ' );

htp.print('// you must keep the following lines on when you use this');

htp.print('// original idea from the Geocities Watermark');

htp.print('// ? Nicolas - http://www.javascript-page.com');

htp.print('var window\_says = "Go to Main Page";') ;

htp.print('var image\_width = 65;' );

htp.print('var image\_height = 31;');

htp.print('var left\_from\_corner = 5;' );

htp.print('var up\_from\_corner = 0;' );

htp.print('var JH = 0;' );

htp.print('var JW = 0;' );

htp.print('var JX = 0;');

htp.print('var JY = 0;' );

htp.print('var left = image\_width + left\_from\_corner + 17;' );

htp.print('var up = image\_height + up\_from\_corner + 15;');

htp.print('if(navigator.appName == "Netscape") {');

htp.print('var wm = document.jsbrand;' );

htp.print('}');

htp.print('if (navigator.appVersion.indexOf("MSIE") != -1){' );

htp.print('var wm = document.all.jsbrand;');

htp.print('}');

htp.print('wm.onmouseover = msover' );

htp.print('wm.onmouseout = msout' );

htp.print('function watermark() {' );

htp.print(' if(navigator.appName == "Netscape") {' );

htp.print(' JH = window.innerHeight');

htp.print(' JW = window.innerWidth' );

htp.print(' JX = window.pageXOffset' );

htp.print(' JY = window.pageYOffset');

htp.print(' wm.visibility = "hide"' );

htp.print(' wm.top = (JH+JY-up)' );

htp.print(' wm.left = (JW+JX-left)' );

htp.print(' wm.visibility= "show"' );

htp.print(' }');

htp.print(' if (navigator.appVersion.indexOf("MSIE") != -1){' );

htp.print(' wm.style.display = "none";');

htp.print(' JH = document.body.clientHeight;' );

htp.print(' JW = document.body.clientWidth;' );

htp.print(' JX = document.body.scrollLeft;' );

htp.print(' JY = document.body.scrollTop;' );

htp.print(' wm.style.top = (JH+JY-up);' );

htp.print(' wm.style.left =(JW+JX-left);' );

htp.print(' wm.style.display = "";' );

htp.print(' ' );

htp.print(' }' );

htp.print('}');

htp.print('function msover(){' );

htp.print(' window.status = window\_says' );

htp.print(' return true;' );

htp.print('}' );

htp.print('function msout(){' );

htp.print(' window.status = ""' );

htp.print(' return true;' );

htp.print('}' );

htp.print('setInterval("watermark()",100);' );

htp.print('//-->' );

htp.print( '</SCRIPT>' );

END;

---------------------------

procedure setlinkstyle

is

BEGIN

htp.print('<STYLE>');

htp.print('<!--');

htp.print('

TD {

FONT-FAMILY: Arial, sans-serif

}

TD.mylink {

BACKGROUND-COLOR:"#f0f0f0"; TEXT-ALIGN: left

}

TD.mylink A:link {

COLOR: black; BACKGROUND-COLOR: #f0f0f0; TEXT-DECORATION: none

}

TD.mylink A:visited {

COLOR: black; BACKGROUND-COLOR: #f0f0f0; TEXT-DECORATION: underline

}

TD.mylink A:hover {

COLOR: black; BACKGROUND-COLOR: #cccccc; TEXT-DECORATION: none

}

TD.mylink A:active {

COLOR: black; BACKGROUND-COLOR: #cccccc; TEXT-DECORATION: underline

}');

htp.print('-->');

htp.print('</STYLE>');

END setlinkstyle;

------------------------------------

PROCEDURE VZ\_RPT\_FOOTER(man\_url IN VARCHAR2) IS

tmpVar NUMBER;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: VZ\_RPT\_FOOTER

PURPOSE: footer for xpercom reports. Gives contact email id and the link

to manual page for the report

REVISIONS:

Ver Date Author Description

--------- ---------- ------------- ------------------------------------

1.0 07-23-03 Vlad Gladyshev . Created this procedure.

EXAMPLE USE: VZ\_RPT\_FOOTER('/docs/ea\_tools/manuaal.doc');

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BEGIN

tmpVar := 0;

htp.LINE;

htp.ADDRESS(htf.sup('&copy;')

|| 'Verizon Wireless');

htp.print(htf.anchor (man\_url, 'Documentation'));

htp.print('<BR>');

/\*htp.MAILTO('wnea-support@scorpion.ddc.vzwcorp.com&subject=OnLine Reports',

'Support');\*/

htp.BODYCLOSE;

htp.HTMLCLOSE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

Null;

WHEN OTHERS THEN

Null;

END;

------------------------------------------------------------------------------

PROCEDURE VZ\_RPT\_FOOTER IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: VZ\_RPT\_FOOTER

PURPOSE: footer for xpercom reports. Gives date, contact email id.

REVISIONS:

Ver Date Author Description

--------- ---------- ------------- ------------------------------------

1.0 06-29-01 anon 1. Created this procedure.

EXAMPLE USE: VZ\_RPT\_FOOTER;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BEGIN

htp.LINE;

htp.ADDRESS('Run Date: '|| to\_char(SYSDATE, 'Mon-DD-yyyy:hh:mi:ss')

|| ' . ' || htf.sup('&copy;')

|| 'Verizon Wireless');

/\*htp.MAILTO('wnea-support@scorpion.ddc.vzwcorp.com&subject=OnLine Reports',

'Support');\*/

htp.BODYCLOSE;

htp.HTMLCLOSE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

Null;

WHEN OTHERS THEN

Null;

END;

------------------------------------------------------------------------------

PROCEDURE selectDateRange

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: selectDateRange

PURPOSE: uses owa\_util.choose\_date to create form elements for selecting

day, month and year.

Limitation: displays a date of 1 to 31 for every month.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BEGIN

htp.htmlOpen;

htp.headOpen;

htp.title('Select Date..');

htp.nl;

htp.headClose;

htp.bodyOpen( cattributes => ' TOPMARGIN="40" BGCOLOR="#F1F1F1" TEXT="#000000"' );

htp.formopen('VZ\_REPORTS.pass\_dates');

htp.centerOpen;

owa\_util.choose\_date('p\_date1', trunc(sysdate) - 7 );

htp.p('<BR>');

htp.para;

htp.para;

htp.para;

htp.para;

htp.print('<h3> thru </h3>');

owa\_util.choose\_date('p\_date2', trunc(sysdate));

htp.para;

htp.para;

htp.para;

htp.para;

htp.centerClose;

htp.centerOpen;

htp.tablerowopen;

htp.tableData(htf.FormSubmit(cValue => 'Select'));

htp.p('<TD>');

htp.print('<INPUT TYPE="RESET" NAME="BUTTON" VALUE="Clear">');

htp.p('</TD>');

htp.formclose;

htp.TableRowClose;

htp.tableClose;

htp.centerClose;

htp.para;

htp.para;

htp.para;

htp.para;

htp.para;

htp.para;

htp.para;

htp.para;

htp.centerOpen;

htp.tableopen('border');

htp.tablerowopen;

htp.p('<td width="329" height="121" bgcolor="#FFFFCC">');

htp.p(' <p align="left"> <i>Pick a Date Range to report on</p><BR>

Default is one week. </i> </td>');

htp.TableRowClose;

htp.tableClose;

htp.centerClose;

htp.bodyClose;

htp.htmlClose;

END selectDateRange;

procedure setXLTab(p\_tabName varchar2 default 'sheet1')

is

BEGIN

htp.p('<HTML xmlns:x="urn:schemas-microsoft-com:office:excel">');

htp.p('<!--[if gte mso 9]><xml>');

htp.p('<x:ExcelWorkbook>');

htp.p('<x:ExcelWorksheets>');

htp.p('<x:ExcelWorksheet>');

htp.p('<x:Name>' || p\_tabName || '</x:Name>');

htp.p('<x:WorksheetOptions>');

htp.p('<x:ProtectContents>False</x:ProtectContents>');

htp.p('<x:ProtectObjects>False</x:ProtectObjects>');

htp.p('<x:ProtectScenarios>False</x:ProtectScenarios>');

htp.p('</x:WorksheetOptions>');

htp.p('</x:ExcelWorksheet>');

htp.p('</x:ExcelWorksheets>');

htp.p('</x:ExcelWorkbook>');

htp.p('</xml><![endif]-->');

END;

procedure setTitleStyle

IS

BEGIN

htp.print('<STYLE>');

htp.print('<!--');

htp.print('

p.title{

COLOR:#9999FF;

FILTER: DropShadow(Color=#223344, OffX=1, OffY=1, Positive=1);

FONT-SIZE: 25px;

FONT-WEIGHT: 900;

HEIGHT: 15px;

}');

htp.p('DIV.page{

margin-left:1.00in;

margin-right:1.00in;

border:solid 1px; }');

--added microsoft office number format style

--this will limit data that is in a date format from being

--represented as a date in excel reporting.

htp.print('.x124{mso-number-format:"\@";}');

htp.print('-->');

htp.print('</STYLE>');

END setTitleStyle;

procedure setreportStyle

as

--need to normailze all styles

begin

--style for div.box is from http://www.plone.org

htp.print( '<STYLE>body { font-style: normal; font-variant: normal; font-weight: normal; font-size:

0.7em; font-family: Verdana, Helvetica, Arial, sans-serif;

background-color: White; background-repeat: repeat;

background-attachment: scroll; color: Black;

scrollbar-base-color: #DEE7EC; scrollbar-highlight-color:

#DEE7EC; scrollbar-track-color: #F7F9FA;

scrollbar-darkshadow-color: #F7F9FA; scrollbar-3dlight-color:

#8CACBB; scrollbar-shadow-color: #8CACBB;

scrollbar-arrow-color: #436976; margin: 0; padding: 0;

background-position: 0% 50% }

.table { border-collapse: collapse; border-spacing: 0px; border: 2px groove #ffffff }

.td { border-spacing: 0px; border: 2px groove #ffffff }

select { font-style: normal; font-variant: normal; font-weight: normal; font-size: 100%;

font-family: Arial, Verdana, Helvetica, sans-serif;

vertical-align: top; border: 1px solid #8cacbb; background-color:#DEE7EC; }

div.box { border-style: none; border-width: medium; margin-left: 0em; margin-right: 0em;

margin-top: 0em; margin-bottom: 2em; padding: 0 }

div.box h5 { background-color: #DEE7EC; background-repeat: repeat; background-attachment:

scroll; color: Black; display: inline; font-size: 1em; height:

1em; border-left: 1px solid #8CACBB;

border-right: 1px solid #8CACBB; border-top: 1px solid #8CACBB;

border-bottom: 1px none #8CACBB }

input { font-style: normal; font-variant: normal; font-weight: normal; font-size: 100%;

font-family: Verdana, Helvetica, Arial, sans-serif; color:

Black; background-color: white; vertical-align: middle;

border: 1px solid #8cacbb; margin-bottom: 1px; padding: 0.1em }

hr {

clear: both;

height: 1px;

color: #8CACBB;

background-color: transparent;}

p.title {

COLOR:#cdcdcd;

FILTER: DropShadow(Color=223344, OffX=1, OffY=1, Positive=1);

FONT-SIZE: 25px;

FONT-WEIGHT: 900;

HEIGHT: 15px;}

</STYLE>');

end;

procedure print\_xml\_tag(

tag\_name in varchar2,

tag\_value in varchar2,

tag\_attr in varchar2 default null ) is

xml\_str varchar2(5000);

begin

if tag\_attr is null then

xml\_str := '<' || tag\_name || '>';

else

xml\_str := '<' || tag\_name || ' ' || tag\_attr || ' >';

end if;

xml\_str := xml\_str || tag\_value;

xml\_str := xml\_str || '</' || tag\_name || '>';

htp.print(xml\_str);

end print\_xml\_tag;

FUNCTION xdate( vs\_dateStr IN VARCHAR2 ) RETURN DATE IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: xdate

PURPOSE: converts a varchar to date

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 01-16-02 anon

(mm-dd-yy) 1. Created this function.

PARAMETERS:

INPUT: varchar2

RETURNED VALUE: DATE if I/P is a valid date; NULL if input is not recognized as a valid date.

LIMITATIONS: closely tied with Oracle's date processing.

NOTES: return value of NULL can be used to flag unrecognized date strings.

created for use in date last tested, to handle date stored as varchar2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

rtn\_date DATE := NULL; -- a string not recognized as date is set to NULL.

vs\_errTxt varchar2(200);

--vc\_str eareports.site\_Attr\_settings.attr\_value%TYPE := rtrim(ltrim(vs\_dateStr));

vc\_str varchar2(250) := rtrim(ltrim(vs\_dateStr));

BEGIN rtn\_date := TO\_DATE(vc\_str, 'MM-DD-YY');

RETURN (rtn\_date);

EXCEPTION WHEN OTHERS THEN

BEGIN rtn\_date := TO\_DATE (vc\_str, 'DD-MON-YY');

RETURN (rtn\_date);

EXCEPTION WHEN OTHERS THEN

BEGIN rtn\_date := NULL; RETURN (rtn\_date); END;END;

END xdate;

FUNCTION chopNPC( v VARCHAR2 )

RETURN VARCHAR2 IS

result VARCHAR2(32000);

BEGIN

result := RTRIM(LTRIM(TRANSLATE(v,CHR(13)||CHR(8)||CHR(10),' ')));

WHILE (INSTR(result,' ') > 0) LOOP

result := REPLACE(result,' ',' ');

END LOOP;

RETURN result;

END chopNPC;

-------------------------------------------------------------

procedure html\_header(p\_displayType varchar2 default 'excel', p\_title varchar2 default '')

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: html\_header

PURPOSE: sets mime type

prints html tags

sets display style

displays title

PARAMETERS:

INPUT:

p\_display\_type excel or html

p\_title title to be displayed

CALLED BY: print\_query

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

pbl\_displayXL BOOLEAN := FALSE;

pv\_browser VZ\_CONSTANTS.VS\_BROWSER%TYPE := VZ\_CONSTANTS.VS\_BROWSER;

BEGIN

--excel download is supported only on Windows.

pv\_browser := owa\_util.get\_cgi\_env('HTTP\_USER\_AGENT');

pbl\_displayXL := (p\_displayType = 'excel' and (instr(pv\_browser, 'Windows' ) > 0));

IF(pbl\_displayXL) THEN

owa\_util.MIME\_header ('application/vnd.ms-excel');

END IF;

htp.htmlOpen;

htp.headOpen;

IF(pbl\_displayXL) THEN

--htp.linkRel('stylesheet', '/includes/excellDisplay.css');

VZ\_REPORTS\_UTILS.SETXLTAB(p\_title ||trunc(sysdate));

END IF;

htp.print('<STYLE>');

htp.print('<!--');

htp.print('

p.title{

COLOR:#9999FF;

FILTER: DropShadow(Color=#223344, OffX=1, OffY=1, Positive=1);

FONT-SIZE: 25px;

FONT-WEIGHT: 900;

HEIGHT: 15px;

}');

htp.print('-->');

htp.print('</STYLE>');

htp.title(p\_title);

htp.headClose;

htp.p('<DIV CLASS="page">');

htp.bodyOpen( VZ\_CONSTANTS.vs\_src || 'backdrop.gif' );

--title is not centered to make it readable in Excel

htp.print( htf.paragraph(calign => 'left', cattributes => ' class=title') || p\_title);

htp.p('</p>');

htp.TABLEOPEN( cattributes => ' border=".1" cellPadding="2" cellSpacing="1" valign="TOP"');

EXCEPTION

WHEN OTHERS THEN

htp.p(SQLERRM);

END ;

----------------------------------------------------------------

PROCEDURE write\_formatted\_stream( p\_column\_data\_inout IN OUT NOCOPY VARCHAR2, p\_data\_type varchar2 default null)

IS

-- column data is read in and modified

-- create logic to display results

begin

htp.TABLEDATA(nvl(to\_char(p\_column\_data\_inout), '<BR>'));--, cattributes => ' bgColor="#ddffdd" ');

end;

----------------------------------------------------------------

procedure print\_query( p\_sql in varchar2, p\_write\_type varchar2 default 'excel',

p\_title varchar2 default '',

p\_isHeader boolean default TRUE )

is

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: print\_query

PURPOSE: uses dynamic sql to execute passed query string with

variable number of arguments

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 Nov 11 -03 Anon 1. Created this procedure.

PARAMETERS:

INPUT:

p\_sql sql string that needs to be executed

p\_write\_type display type, html, excel.

p\_title title to be displayed- this is handled by html\_header

p\_isHeader html header to be diplyed or not.

LIMITATION: no bind is used and all columns are treated as varchar2.

this may need to be modified if inserts are required.

some modification required to handle output as xml

CALLS: write\_formatted\_stream where the column values are written.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--columns retrieved are read as varchar.

v\_col\_value varchar2(1000);

-- return value of executing the cursor.holds number of

--rows only for INSERT, DELETE or UPDATE statements and undefined for rest.

v\_status integer;

v\_colCnt number default 0; -- number of columns in select list

v\_cnt number default 0; -- used in fetch loop; may not be needed

v\_recTbl dbms\_sql.desc\_tab; -- query is described for select list

v\_curHandle number default dbms\_sql.open\_cursor; --cursor handle to sql

v\_bgColor varchar2(10) := 'color1'; -- will be used to alternate colors

--get execution time

lt\_start\_time PLS\_INTEGER default dbms\_utility.get\_time;

lt\_execTime PLS\_INTEGER;

begin

--parse the input query

dbms\_sql.parse( v\_curHandle, p\_sql, dbms\_sql.native );

-- fill table l\_recTbl with description of columns

dbms\_sql.describe\_columns( v\_curHandle, v\_colCnt, v\_recTbl );

-- records are for diplay and so no binding is used.

for i in 1 .. v\_colCnt

loop

/\* dbms\_sql.define\_column defines a column to be selected from the given cursor. This

procedure is only used with SELECT cursors.

The column being defined is identified by its relative position in the SELECT list of

the statement in the given cursor. The type of the COLUMN value determines the type

of the column being defined. Columns are defined by their position in the select list\*/

dbms\_sql.define\_column( v\_curHandle, i, v\_col\_value, 1000 );

end loop;

v\_status := dbms\_sql.execute(v\_curHandle);

--print html header; also has the mime type set.

if (p\_isHeader) then

html\_header(p\_write\_type, p\_title);

end if;

--display column headings

htp.tableRowOpen ( cattributes => ' bgColor="#cccccc" borderColorDark="#fcfcfc" borderColorLight="#999999" ' );

for i in 1 .. v\_colCnt loop

--dbms\_sql.column\_value returns the value of the cursor element at position i

dbms\_sql.column\_value( v\_curHandle, i, v\_col\_value );

htp.TABLEDATA( cvalue => v\_recTbl(i).col\_name , cattributes => 'FILTER="ALL"');

end loop;

--end display of column headers

htp.TableRowClose;

while ( dbms\_sql.fetch\_rows(v\_curHandle) > 0 )

loop

v\_cnt := v\_cnt+1; -- this could possibly be obtained from return value of fetch rows.

if (v\_bgColor = 'color1') then --change background color of alternate rows

-- highlight with #FFFF00 color, when mouse is moved over a record and restore background when moved out.

htp.p('<TR bgColor="#FFFFCC" borderColorDark="#fcfcfc" borderColorLight="#FFFFCC" onMouseOver="this.bgColor= ''#FFFF00'';" onMouseOut="this.bgColor= ''#FFFFCC'' ;">');

v\_bgColor := 'color2';

else

htp.p('<TR bgColor="#f0f0f0" borderColorDark="#fcfcfc" borderColorLight="#f0f0f0" onMouseOver="this.bgColor= ''#FFFF00'';" onMouseOut="this.bgColor= ''#f0f0f0'';">');

v\_bgColor := 'color1';

end if;

--loop through all columns

for i in 1 .. v\_colCnt loop

/\* dbms\_sql.column\_value returns the value of the cursor element for a given position in a

given cursor. This procedure is used to access the data fetched by calling FETCH\_ROWS.\*/

dbms\_sql.column\_value( v\_curHandle, i, v\_col\_value );

write\_formatted\_stream(v\_col\_value);

end loop;

htp.TableRowClose;

end loop;

dbms\_sql.close\_cursor(v\_curHandle);

htp.tableClose;

--htp.p('</center>');

--get time to execute

-- lt\_execTime := to\_char((dbms\_utility.get\_time - lt\_start\_time)/1000, '99999.999');

-- htp.p('<B>'|| v\_cnt || ' Rows returned in '|| lt\_execTime || ' seconds.</B>');

if (p\_isHeader) then

VZ\_REPORTS\_UTILS.VZ\_RPT\_FOOTER;

end if;

exception

when others then

dbms\_sql.close\_cursor( v\_curHandle );

--replace it with a error handler

htp.p(sqlerrm);

raise;

end ;

----------------------------------------------------------------

PROCEDURE SET\_IN\_LIST(p\_varcharTbl IN VZ\_CONSTANTS.site\_tbl\_ty,

p\_in\_col varchar2,

p\_sql\_inout IN OUT long,

p\_colName varchar2,

p\_selectionTbl IN OUT VZ\_CONSTANTS.site\_tbl\_ty

) IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: SET\_IN\_LIST

PURPOSE: overloaded to handle various scenarios

build an IN clause to the select statement.

and return the built clause for diplay in report header.

Date: 11-12-03

Author: ANON

NOTES: p\_varcharTbl IN VZ\_CONSTANTS.site\_tbl\_ty is tabla with elements of in list

p\_inClause is sql in clause; and column in

p\_sql\_inout the in list clause will be appended to the original sql

USAGE: SET\_IN\_LIST(sitenum\_in,' and CIRC\_PATH\_INST.status in ( ' ,v\_sql);

v\_sql is the original sql.to this is appended- in this case

and circ\_path\_inst.status in (' foo ', ' bar ')

sitenum\_in is a table having the elements for the in list

this could be put in a utils package but is left here for convenience of

specialized manipulation

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--SET\_IN\_LIST(sitenum\_in,' and CIRC\_PATH\_INST.status in ( ' ,v\_sql);

vn\_listCnt integer;

-- v\_selectionString varchar2(200) default ' '; -- the selection made by user

v\_selectionString long default ' '; -- the selection made by user

begin

vn\_listCnt := p\_varcharTbl.COUNT; -- get number of elements in the passed array. This can never be 0

IF ( vn\_listCnt = 1 ) AND (p\_varcharTbl(1) = '0') THEN

--the first element in the selection is ALL which has a value of '0'

--if this is in anyother position there is no harm.

vz\_constants.v\_all := vz\_constants.v\_all || '0';

ELSE

--append in clause; and column\_name in (

p\_sql\_inout := p\_sql\_inout || ' and ' || p\_in\_col || ' in (' ;

--the last element in the list needs to be followed with );

FOR I IN 1 .. vn\_listCnt - 1

LOOP

p\_sql\_inout := p\_sql\_inout || '''' ||p\_varcharTbl(i)|| ''''||' , ';

v\_selectionString := v\_selectionString || p\_varcharTbl(i) || ';';

END LOOP;

--close the in list with );

p\_sql\_inout := p\_sql\_inout || '''' || p\_varcharTbl(vn\_listCnt) || ''''||' )';

v\_selectionString := v\_selectionString || p\_varcharTbl(vn\_listCnt) || ';';

p\_selectionTbl(p\_selectionTbl.count + 1) := v\_selectionString;

END IF;

EXCEPTION

WHEN OTHERS THEN

htp.p(sqlerrm);

raise;

END ;

function fn\_this\_caller return varchar2

is

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: fn\_this\_caller

PURPOSE: returns caller\_type-owner.name

e.g., PACKAGE-WNEA.vz\_report

Note: when caller type is an Anonymous block v\_caller\_t is NULL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

v\_owner varchar2(50);

v\_name varchar2(50);

vn\_lineno number;

v\_type varchar2(50);

v\_caller\_t varchar2(50);

begin

owa\_util.who\_called\_me( v\_owner, v\_name, vn\_lineno, v\_caller\_t);

return nvl(v\_caller\_t||'-'||v\_owner || '.' || v\_name, 'Anonymous');

end;

FUNCTION xml\_formatter(original\_sql IN VARCHAR2) RETURN VARCHAR2 is

replacement\_sql VARCHAR2(32767);

begin

replacement\_sql := replace(original\_sql, '&','&amp;');

replacement\_sql := replace(replacement\_sql, '<','&lt;');

return replacement\_sql;

end;

END VZ\_REPORTS\_UTILS;

/

--------------------------------------------------------

-- DDL for Package Body WATCHDOG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."WATCHDOG"

IS

FUNCTION compare\_table\_columns (p\_process\_id number, p\_schema varchar2, p\_wk\_table varchar2, p\_app\_table varchar2) RETURN number

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PURPOSE: compare the columns in one table to another to make sure they match

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

v\_err\_cnt number;

BEGIN

v\_err\_cnt := 0;

select sum(err\_cnt) as err\_cnt

into v\_err\_cnt

from --cks to see column counts are the same

(select count(\*) as err\_cnt

from (select p\_process\_id as process\_id, table\_name, count(\*) as col\_cnt

from all\_tab\_columns

where owner = p\_schema

and table\_name = p\_wk\_table

group by table\_name) a,

(select p\_process\_id as process\_id, table\_name, count(\*) as col\_cnt

from all\_tab\_columns

where owner = p\_schema

and table\_name = p\_app\_table

group by table\_name) b

where a.process\_id = b.process\_id

and a.col\_cnt <> b.col\_cnt

union

--cks to see if columns type and length are the same

select count(\*) as err\_cnt

from (select p\_process\_id as process\_id, column\_id, column\_name||data\_type||data\_length as col\_data

from all\_tab\_columns

where owner = p\_schema

and table\_name = p\_wk\_table) a,

(select p\_process\_id as process\_id, column\_id, column\_name||data\_type||data\_length as col\_data

from all\_tab\_columns

where owner = p\_schema

and table\_name = p\_app\_table) b

where a.process\_id = b.process\_id

and a.column\_id = b.column\_id

and a.col\_data <> b.col\_data);

return v\_err\_cnt;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

WHEN OTHERS THEN

-- Consider logging the error and then re-raise

RAISE;

END compare\_table\_columns;

PROCEDURE updateprocessstart (p\_processname IN VARCHAR2)

IS

v\_process xng\_reports.all\_processes.process\_name%TYPE;

BEGIN

v\_process := p\_processname;

UPDATE xng\_reports.all\_processes

SET end\_time = NULL,

exec\_status = 2,

is\_ready = 'N',

start\_time = SYSDATE

WHERE process\_name = v\_process;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END updateprocessstart;

PROCEDURE updateprocessend (

p\_processname IN VARCHAR2,

p\_executionstatus IN VARCHAR2,

p\_isready IN VARCHAR2

)

IS

v\_process xng\_reports.all\_processes.process\_name%TYPE;

v\_exec\_status xng\_reports.all\_processes.exec\_status%TYPE;

v\_isready xng\_reports.all\_processes.is\_ready%TYPE;

BEGIN

v\_process := p\_processname;

v\_isready := p\_isready;

CASE p\_executionstatus

WHEN 'STATUS\_FAILURE'

THEN

v\_exec\_status := 0;

WHEN 'STATUS\_SUCCESS\_WITH\_WARNINGS'

THEN

v\_exec\_status := 1;

WHEN 'STATUS\_TRANSIENT'

THEN

v\_exec\_status := 2;

WHEN 'STATUS\_SUCCESS'

THEN

v\_exec\_status := 3;

ELSE

v\_exec\_status := 0;

END CASE;

UPDATE xng\_reports.all\_processes

SET end\_time = SYSDATE,

exec\_status = v\_exec\_status,

is\_ready = v\_isready

WHERE process\_name = v\_process;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

watchdog.logerror (v\_process,

4000,

SUBSTR ( 'Error in updateprocessend(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend (v\_process, 0, 'N');

--RAISE;

END updateprocessend;

PROCEDURE updateextractdate (

p\_processname IN VARCHAR2,

p\_fileextractdate IN VARCHAR2

)

IS

v\_process xng\_reports.all\_processes.process\_name%TYPE;

v\_file\_extract\_date DATE;

BEGIN

v\_process := p\_processname;

v\_file\_extract\_date := TO\_DATE (p\_fileextractdate, 'mm/dd/yyyy');

UPDATE xng\_reports.all\_processes

SET last\_file\_extract\_date = v\_file\_extract\_date

WHERE process\_name = v\_process;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END updateextractdate;

PROCEDURE logerror (

p\_processname IN VARCHAR2,

p\_errorcode IN VARCHAR2,

p\_errormessage IN VARCHAR2,

p\_emailnotify IN VARCHAR2

)

IS

v\_process xng\_reports.all\_processes.process\_name%TYPE;

v\_process\_id xng\_reports.wd\_error\_details.process\_id%TYPE;

v\_eid xng\_reports.wd\_error\_details.error\_id%TYPE;

v\_msg xng\_reports.wd\_error\_details.error\_msg%TYPE;

v\_notify xng\_reports.wd\_error\_details.notify%TYPE;

BEGIN

v\_process := p\_processname;

v\_msg := p\_errormessage;

v\_notify := p\_emailnotify;

SELECT process\_id

INTO v\_process\_id

FROM xng\_reports.all\_processes

WHERE process\_name = v\_process;

SELECT MIN (eid)

INTO v\_eid

FROM (SELECT ec.error\_id eid

FROM xng\_reports.wd\_error\_code ec

WHERE ec.ERROR\_CODE = p\_errorcode OR ec.ERROR\_CODE = 'UNKNOWN');

INSERT INTO wd\_error\_details

(process\_id, error\_id, error\_msg, notify

)

VALUES (v\_process\_id, v\_eid, v\_msg, v\_notify

);

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END logerror;

PROCEDURE populate\_last\_successful\_date

IS

BEGIN

UPDATE xng\_reports.all\_processes p

SET p.last\_successful\_exec\_date = p.last\_file\_extract\_date

WHERE p.last\_file\_extract\_date IS NOT NULL

AND p.is\_ready = 'Y'

AND p.RUN\_CRON = 'Y';

COMMIT;

UPDATE xng\_reports.all\_processes p

SET p.last\_successful\_exec\_date = p.end\_time

WHERE p.last\_file\_extract\_date IS NULL

AND p.is\_ready = 'Y'

AND p.RUN\_CRON = 'Y';

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END populate\_last\_successful\_date;

PROCEDURE populate\_last\_success\_dt\_proc(

p\_processname IN VARCHAR2

)

IS

BEGIN

UPDATE xng\_reports.all\_processes p

SET p.last\_successful\_exec\_date = p.last\_file\_extract\_date

WHERE p.last\_file\_extract\_date IS NOT NULL

AND p.is\_ready = 'Y'

AND p.PROCESS\_NAME=p\_processname;

COMMIT;

UPDATE xng\_reports.all\_processes p

SET p.last\_successful\_exec\_date = p.end\_time

WHERE p.last\_file\_extract\_date IS NULL

AND p.is\_ready = 'Y'

AND p.PROCESS\_NAME=p\_processname;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END populate\_last\_success\_dt\_proc;

PROCEDURE retore\_is\_ready\_flag

IS

BEGIN

UPDATE xng\_reports.all\_processes p

SET p.is\_ready = 'N'

WHERE p.process\_type = 'D'

AND p.RUN\_CRON = 'Y';

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END retore\_is\_ready\_flag;

PROCEDURE update\_copy\_status (

p\_processname IN VARCHAR2,

p\_executionstatus IN VARCHAR2,

p\_isready IN VARCHAR2

)

IS

v\_process xng\_reports.all\_processes.process\_name%TYPE;

v\_exec\_status xng\_reports.all\_processes.exec\_status%TYPE;

v\_isready xng\_reports.all\_processes.is\_ready%TYPE;

BEGIN

v\_process := p\_processname;

v\_isready := p\_isready;

CASE p\_executionstatus

WHEN 'STATUS\_FAILURE'

THEN

v\_exec\_status := 0;

WHEN 'STATUS\_SUCCESS\_WITH\_WARNINGS'

THEN

v\_exec\_status := 1;

WHEN 'STATUS\_TRANSIENT'

THEN

v\_exec\_status := 2;

WHEN 'STATUS\_SUCCESS'

THEN

v\_exec\_status := 3;

ELSE

v\_exec\_status := 0;

END CASE;

UPDATE xng\_reports.all\_processes

SET exec\_status = v\_exec\_status,

is\_ready = v\_isready

WHERE process\_name = v\_process;

COMMIT;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put (SQLCODE);

DBMS\_OUTPUT.put (SQLERRM);

RAISE;

END update\_copy\_status;

PROCEDURE copy\_table\_prd\_to\_wk (p\_process\_name in varchar2)

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PURPOSE: truncate specific wk table and insert values from prd tables

determined by copy\_wk\_tables table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

v\_process\_id copy\_wk\_tables.process\_id%type;

v\_process\_name all\_processes.process\_name%type;

v\_seq\_no copy\_wk\_tables.seq\_no%type;

v\_schema copy\_wk\_tables.schema\_name%type;

v\_wk\_table copy\_wk\_tables.wk\_table\_name%type;

v\_prd\_table copy\_wk\_tables.prd\_table\_name%type;

v\_where\_clause copy\_wk\_tables.where\_clause%type;

v\_err\_cnt number;

v\_run\_err varchar2(255);

sql\_str VARCHAR2(32767);

TYPE RowCntTyp IS REF CURSOR;

row\_cnt\_cv RowCntTyp;

v\_row\_cnt number;

cursor detail\_cursor is

select process\_id, process\_name, seq\_no, schema\_name, wk\_table\_name, prd\_table\_name, where\_clause

from copy\_wk\_tables\_ready\_v

where process\_name = p\_process\_name

order by process\_id, seq\_no;

BEGIN

v\_err\_cnt := 1;

v\_run\_err := ' ';

v\_row\_cnt := 0;

OPEN detail\_cursor;

LOOP

FETCH detail\_cursor

INTO v\_process\_id, v\_process\_name, v\_seq\_no, v\_schema, v\_wk\_table, v\_prd\_table, v\_where\_clause;

EXIT WHEN detail\_cursor%NOTFOUND;

--check make sure there are records in the wk table before copying

v\_row\_cnt := 0;

sql\_str := 'select count (\*) as row\_cnt from '||v\_schema||'.'||v\_wk\_table;

if v\_where\_clause is not null then

sql\_str := sql\_str||' '||v\_where\_clause;

end if;

OPEN row\_cnt\_cv FOR sql\_str;

LOOP

FETCH row\_cnt\_cv INTO v\_row\_cnt;

EXIT WHEN row\_cnt\_cv%NOTFOUND;

END LOOP;

CLOSE row\_cnt\_cv;

if v\_row\_cnt > 0 then

v\_err\_cnt := 1;

v\_run\_err := ' ';

v\_err\_cnt := compare\_table\_columns(v\_process\_id, v\_schema, v\_wk\_table, v\_prd\_table);

if v\_err\_cnt = 0 then

BEGIN

if v\_where\_clause is null then

sql\_str :=

'truncate table '||v\_schema||'.'||v\_wk\_table||' reuse storage';

else

sql\_str :=

'delete from '||v\_schema||'.'||v\_wk\_table||' '||v\_where\_clause;

end if;

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

if v\_where\_clause is null then

sql\_str :=

'insert into '||v\_schema||'.'||v\_wk\_table||' (select \* from '||v\_schema||'.'||v\_prd\_table||')';

else

sql\_str :=

'insert into '||v\_schema||'.'||v\_wk\_table||' (select \* from '||v\_schema||'.'||v\_prd\_table||' '||v\_where\_clause||')';

end if;

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line (SUBSTR (v\_process\_name||': '||TO\_CHAR (SQLCODE)|| ': '|| SQLERRM, 1, 255));

v\_run\_err := SUBSTR ('Error in COPY\_TABLE\_PRD\_TO\_WK; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': wk\_table = '|| v\_wk\_table ||' / prd\_table = '|| v\_prd\_table||'; '||TO\_CHAR (SQLCODE)|| ': '|| SQLERRM, 1, 2000);

watchdog.logerror (v\_process\_name, 4000, v\_run\_err,'Y');

update\_copy\_status (v\_process\_name, 'STATUS\_FAILURE', 'N');

END;

else

v\_run\_err := SUBSTR('Error in COPY\_TABLE\_PRD\_TO\_WK; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': Columns in '|| v\_prd\_table ||'do not match columns in '|| v\_wk\_table, 1, 2000);

watchdog.logerror (v\_process\_name, 4001, v\_run\_err,'Y');

update\_copy\_status (v\_process\_name, 'STATUS\_FAILURE', 'N');

end if;

else

v\_run\_err := SUBSTR('Error in COPY\_TABLE\_PRD\_TO\_WK; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': No records in '|| v\_prd\_table ||' so not copied into '|| v\_wk\_table, 1, 2000);

watchdog.logerror (v\_process\_name, 4002, v\_run\_err,'Y');

end if;

END LOOP;

CLOSE detail\_cursor;

COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

-- raise;

END copy\_table\_prd\_to\_wk;

PROCEDURE copy\_table\_wk\_to\_prd (p\_process\_name in varchar2)

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PURPOSE: truncate specific prd table and insert values from wk table

determined by copy\_wk\_tables table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

v\_process\_id copy\_wk\_tables.process\_id%type;

v\_process\_name all\_processes.process\_name%type;

v\_seq\_no copy\_wk\_tables.seq\_no%type;

v\_schema copy\_wk\_tables.schema\_name%type;

v\_wk\_table copy\_wk\_tables.wk\_table\_name%type;

v\_prd\_table copy\_wk\_tables.prd\_table\_name%type;

v\_where\_clause copy\_wk\_tables.where\_clause%type;

v\_err\_cnt number;

v\_run\_err varchar2(255);

sql\_str VARCHAR2(32767);

TYPE RowCntTyp IS REF CURSOR;

row\_cnt\_cv RowCntTyp;

v\_row\_cnt number;

cursor detail\_cursor is

select process\_id, process\_name, seq\_no, schema\_name, wk\_table\_name, prd\_table\_name, where\_clause

from copy\_wk\_tables\_ready\_v

where process\_name = p\_process\_name

order by process\_id, seq\_no;

BEGIN

v\_err\_cnt := 1;

v\_run\_err := ' ';

v\_row\_cnt := 0;

OPEN detail\_cursor;

LOOP

FETCH detail\_cursor

INTO v\_process\_id, v\_process\_name, v\_seq\_no, v\_schema, v\_wk\_table, v\_prd\_table, v\_where\_clause;

EXIT WHEN detail\_cursor%NOTFOUND;

--check make sure there are records in the wk table before copying

v\_row\_cnt := 0;

sql\_str := 'select count (\*) as row\_cnt from '||v\_schema||'.'||v\_wk\_table;

if v\_where\_clause is not null then

sql\_str := sql\_str||' '||v\_where\_clause;

end if;

OPEN row\_cnt\_cv FOR sql\_str;

LOOP

FETCH row\_cnt\_cv INTO v\_row\_cnt;

EXIT WHEN row\_cnt\_cv%NOTFOUND;

END LOOP;

CLOSE row\_cnt\_cv;

if v\_row\_cnt > 0 then

v\_err\_cnt := 1;

v\_run\_err := ' ';

v\_err\_cnt := compare\_table\_columns(v\_process\_id, v\_schema, v\_wk\_table, v\_prd\_table);

if v\_err\_cnt = 0 then

BEGIN

if v\_where\_clause is null then

sql\_str :=

'truncate table '||v\_schema||'.'||v\_prd\_table||' reuse storage';

else

sql\_str :=

'delete from '||v\_schema||'.'||v\_prd\_table||' '||v\_where\_clause;

end if;

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

if v\_where\_clause is null then

sql\_str :=

'insert into '||v\_schema||'.'||v\_prd\_table||' (select \* from '||v\_schema||'.'||v\_wk\_table||')';

else

sql\_str :=

'insert into '||v\_schema||'.'||v\_prd\_table||' (select \* from '||v\_schema||'.'||v\_wk\_table||' '||v\_where\_clause||')';

end if;

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

sql\_str :=

'update xng\_reports.copy\_wk\_tables

set last\_copy\_date = sysdate

where process\_id = '''||v\_process\_id||'''

and seq\_no = '''||v\_seq\_no||'''';

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line (SUBSTR (v\_process\_name||': '||TO\_CHAR (SQLCODE)|| ': '|| SQLERRM, 1, 255));

v\_run\_err := SUBSTR ('Error in COPY\_TABLE\_WK\_TO\_PRD; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': wk\_table = '|| v\_wk\_table ||' / prd\_table = '|| v\_prd\_table||'; '||TO\_CHAR (SQLCODE)|| ': '|| SQLERRM, 1, 2000);

watchdog.logerror (v\_process\_name, 4000, v\_run\_err,'Y');

update\_copy\_status (v\_process\_name, 'STATUS\_FAILURE', 'N');

END;

else

v\_run\_err := SUBSTR('Error in COPY\_TABLE\_WK\_TO\_PRD; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': Columns in '|| v\_wk\_table ||'do not match columns in '|| v\_prd\_table, 1, 2000);

watchdog.logerror (v\_process\_name, 4001, v\_run\_err,'Y');

update\_copy\_status (v\_process\_name, 'STATUS\_FAILURE', 'N');

end if;

else

v\_run\_err := SUBSTR('Error in COPY\_TABLE\_WK\_TO\_PRD; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': No records in '|| v\_wk\_table ||' so not copied into '|| v\_prd\_table, 1, 2000);

watchdog.logerror (v\_process\_name, 4002, v\_run\_err,'Y');

end if;

END LOOP;

CLOSE detail\_cursor;

COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

-- raise;

END copy\_table\_wk\_to\_prd;

PROCEDURE copy\_wk\_tables\_process

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PURPOSE: truncate prd tables and insert values from wrk tables then truncate

wk tables for all processes in copy\_wk\_tables table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

v\_process\_id copy\_wk\_tables.process\_id%type;

v\_process\_name all\_processes.process\_name%type;

v\_seq\_no copy\_wk\_tables.seq\_no%type;

v\_schema copy\_wk\_tables.schema\_name%type;

v\_wk\_table copy\_wk\_tables.wk\_table\_name%type;

v\_prd\_table copy\_wk\_tables.prd\_table\_name%type;

v\_where\_clause copy\_wk\_tables.where\_clause%type;

v\_err\_cnt number;

v\_run\_err varchar2(255);

sql\_str VARCHAR2(32767);

TYPE RowCntTyp IS REF CURSOR;

row\_cnt\_cv RowCntTyp;

v\_row\_cnt number;

cursor detail\_cursor is

select process\_id, process\_name, seq\_no, schema\_name, wk\_table\_name, prd\_table\_name, where\_clause

from copy\_wk\_tables\_ready\_v

order by process\_id, seq\_no;

BEGIN

v\_err\_cnt := 1;

v\_run\_err := ' ';

v\_row\_cnt := 0;

OPEN detail\_cursor;

LOOP

FETCH detail\_cursor

INTO v\_process\_id, v\_process\_name, v\_seq\_no, v\_schema, v\_wk\_table, v\_prd\_table, v\_where\_clause;

EXIT WHEN detail\_cursor%NOTFOUND;

--check make sure there are records in the wk table before copying

v\_row\_cnt := 0;

sql\_str := 'select count (\*) as row\_cnt from '||v\_schema||'.'||v\_wk\_table;

if v\_where\_clause is not null then

sql\_str := sql\_str||' '||v\_where\_clause;

end if;

OPEN row\_cnt\_cv FOR sql\_str;

LOOP

FETCH row\_cnt\_cv INTO v\_row\_cnt;

EXIT WHEN row\_cnt\_cv%NOTFOUND;

END LOOP;

CLOSE row\_cnt\_cv;

if v\_row\_cnt > 0 then

v\_err\_cnt := 1;

v\_run\_err := ' ';

v\_err\_cnt := compare\_table\_columns(v\_process\_id, v\_schema, v\_wk\_table, v\_prd\_table);

if v\_err\_cnt = 0 then

BEGIN

if v\_where\_clause is null then

sql\_str :=

'truncate table '||v\_schema||'.'||v\_prd\_table||' reuse storage';

else

sql\_str :=

'delete from '||v\_schema||'.'||v\_prd\_table||' '||v\_where\_clause;

end if;

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

if v\_where\_clause is null then

sql\_str :=

'insert into '||v\_schema||'.'||v\_prd\_table||' (select \* from '||v\_schema||'.'||v\_wk\_table||')';

else

sql\_str :=

'insert into '||v\_schema||'.'||v\_prd\_table||' (select \* from '||v\_schema||'.'||v\_wk\_table||' '||v\_where\_clause||')';

end if;

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

sql\_str :=

'update xng\_reports.copy\_wk\_tables

set last\_copy\_date = sysdate

where process\_id = '''||v\_process\_id||'''

and seq\_no = '''||v\_seq\_no||'''';

EXECUTE IMMEDIATE sql\_str;

COMMIT WORK;

if v\_where\_clause is null then

sql\_str :=

'truncate table '||v\_schema||'.'||v\_wk\_table||' reuse storage';

else

sql\_str :=

'delete from '||v\_schema||'.'||v\_wk\_table||' '||v\_where\_clause;

end if;

-- EXECUTE IMMEDIATE sql\_str;

-- COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line (SUBSTR (v\_process\_name||': '||TO\_CHAR (SQLCODE)|| ': '|| SQLERRM, 1, 255));

v\_run\_err := SUBSTR ('Error in COPY\_WK\_TABLES\_PROCESS; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': wk\_table = '|| v\_wk\_table ||' / prd\_table = '|| v\_prd\_table||'; '||TO\_CHAR (SQLCODE)|| ': '|| SQLERRM, 1, 2000);

watchdog.logerror (v\_process\_name, 4000, v\_run\_err,'Y');

update\_copy\_status (v\_process\_name, 'STATUS\_FAILURE', 'N');

END;

else

v\_run\_err := SUBSTR('Error in COPY\_WK\_TABLES\_PROCESS; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': Columns in '|| v\_wk\_table ||'do not match columns in '|| v\_prd\_table, 1, 2000);

watchdog.logerror (v\_process\_name, 4001, v\_run\_err,'Y');

update\_copy\_status (v\_process\_name, 'STATUS\_FAILURE', 'N');

end if;

else

v\_run\_err := SUBSTR('Error in COPY\_TABLE\_WK\_TO\_PRD; '|| v\_process\_name ||'- seq\_no '||v\_seq\_no||': No records in '|| v\_wk\_table ||' so not copied into '|| v\_prd\_table, 1, 2000);

watchdog.logerror (v\_process\_name, 4002, v\_run\_err,'Y');

end if;

END LOOP;

CLOSE detail\_cursor;

COMMIT WORK;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(SQLCODE);

dbms\_output.put(SQLERRM);

-- raise;

END copy\_wk\_tables\_process;

PROCEDURE all\_processes\_wrapper

IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PURPOSE: truncate prd tables and replace with wrk tables for all processes

in copy\_wk\_tables table and reset dates in all\_prcesses table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BEGIN

updateprocessstart ('BACKHAUL\_AUDIT\_COMPLETE');

copy\_wk\_tables\_process;

updateprocessend ('BACKHAUL\_AUDIT\_COMPLETE', 'STATUS\_SUCCESS', 'Y');

populate\_last\_successful\_date;

retore\_is\_ready\_flag;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.put\_line (SUBSTR ( 'Error in move\_csr\_data\_from\_wk(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

)

);

watchdog.logerror ('BACKHAUL\_AUDIT\_COMPLETE',

4000,

SUBSTR ( 'Error in restore\_all\_process(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255

),

'Y'

);

watchdog.updateprocessend ('BACKHAUL\_AUDIT\_COMPLETE', 0, 'N');

--RAISE;

END all\_processes\_wrapper;

END WATCHDOG;

/

--------------------------------------------------------

-- DDL for Package Body XNGSTD\_AUDIT\_BATCH\_PKG

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."XNGSTD\_AUDIT\_BATCH\_PKG"

AS

PROCEDURE XNGSTD\_AUDIT\_BATCH\_PROC

IS

PROCESSNAME VARCHAR2 (100);

BEGIN

processName := 'XNGSTD\_AUDIT';

-- Tell watchDog the process has started

watchdog.updateprocessstart (processName);

BEGIN

DBMS\_OUTPUT.put\_line ('START XNGSTD\_AUDIT\_BATCH\_PROC');

XNGSTD\_AUDIT\_SUMMARY\_PROC;

DBMS\_OUTPUT.put\_line ('DONE XNGSTD\_AUDIT\_SUMMARY\_PROC');

XNGSTD\_AUDIT\_SITE\_PROC;

DBMS\_OUTPUT.put\_line ('DONE XNGSTD\_AUDIT\_SITE\_PROC');

XNGSTD\_AUDIT\_EQUIP\_PROC;

DBMS\_OUTPUT.put\_line ('DONE XNGSTD\_AUDIT\_EQUIP\_PROC');

XNGSTD\_AUDIT\_PATH\_PROC;

DBMS\_OUTPUT.put\_line ('DONE XNGSTD\_AUDIT\_PATH\_PROC');

XNGSTD\_AUDIT\_SEGMENT\_PROC;

DBMS\_OUTPUT.put\_line ('DONE XNGSTD\_AUDIT\_SEGMENT\_PROC');

XNGSTD\_AUDIT\_POST\_PROC;

DBMS\_OUTPUT.put\_line ('DONE XNGSTD\_AUDIT\_BATCH\_PROC');

-- mark successful completion of the task

WATCHDOG.LOGERROR (PROCESSNAME,

'STATUS\_SUCCESS',

PROCESSNAME || ' ran successfully!',

'N');

WATCHDOG.UPDATEPROCESSEND (PROCESSNAME, 'STATUS\_SUCCESS', 'Y');

WATCHDOG.POPULATE\_LAST\_SUCCESS\_DT\_PROC (PROCESSNAME);

DBMS\_OUTPUT.PUT\_LINE (

'XNGSTD\_AUDIT\_BATCH\_PKG.XNGSTD\_AUDIT\_BATCH\_PROC() ran successfully!');

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.PUT\_LINE (

SUBSTR (

'Error: XNGSTD\_AUDIT\_BATCH\_PKG.XNGSTD\_AUDIT\_BATCH\_PROC(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255));

RAISE;

END;

EXCEPTION

WHEN OTHERS

THEN

DBMS\_OUTPUT.PUT\_LINE (

SUBSTR (

'Error: XNGSTD\_AUDIT\_BATCH\_PKG.XNGSTD\_AUDIT\_BATCH\_PROC(): '

|| TO\_CHAR (SQLCODE)

|| ': '

|| SQLERRM,

1,

255));

-- mark failed completion of the task

WATCHDOG.UPDATEPROCESSEND (PROCESSNAME, 'STATUS\_FAILURE', 'N');

END;

PROCEDURE XNGSTD\_AUDIT\_SUMMARY\_PROC

IS

BEGIN

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SUM\_BK (AREA,

REGION,

SITE\_FAULT\_PCT,

EQUIP\_FAULT\_PCT,

PATH\_FAULT\_PCT,

SEG\_FAULT\_PCT,

OVERALL\_FAULT\_PCT,

CREATION\_DATE)

SELECT AREA,

REGION,

SITE\_FAULT\_PCT,

EQUIP\_FAULT\_PCT,

PATH\_FAULT\_PCT,

SEG\_FAULT\_PCT,

OVERALL\_FAULT\_PCT,

CREATION\_DATE

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SUM;

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_SUM;

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SUM (AREA, REGION, CREATION\_DATE)

WITH AREA\_REGION

AS (SELECT \*

FROM XNG\_REPORTS.VZW\_NETWORK\_ORG

WHERE MARKET\_DOMAIN\_NAME IN

(SELECT DOMAIN\_NAME

FROM XNG\_REPORTS.XNGSTD\_DOMAIN\_REGION))

SELECT AREA, REGION, SYSDATE

FROM (SELECT DISTINCT AREA, REGION, 1 P FROM AREA\_REGION

UNION

SELECT DISTINCT

AREA || ' Area Subtotal',

'Total Compliance' REGION,

2 P

FROM AREA\_REGION)

ORDER BY AREA, P, REGION;

COMMIT;

END;

PROCEDURE XNGSTD\_AUDIT\_SITE\_PROC

IS

sqlStmt VARCHAR2 (32000);

BEGIN

--. FIND EACH FAULT IN DETAIL

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_DETAIL;

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_DETAIL (AREA,

REGION,

VW\_TYPE,

SITE\_INST\_ID,

SITE\_NAME,

DOMAIN\_NAME,

CATEGORY,

STATE,

NAME\_NULL,

SITEID\_NULL,

LATITUDE\_NULL,

LONGITUDE\_NULL,

STREET\_NULL,

-- CITY\_NULL,

-- STATE\_NULL,

POSTALCODE\_NULL,

FLOOR\_NULL,

COUNTY\_NULL,

COUNTRY\_NULL,

DIRECTIONS\_NULL,

FIREPHONE\_FAULT,

POLICEPHONE\_FAULT,

POWERPHONE\_FAULT,

POWERACCT\_NULL,

POWERCOMPANY\_NULL)

SELECT area,

region,

VW\_TYPE,

site\_inst\_id,

sitename,

domainname,

category,

state,

nameNullCnt,

siteidNullCnt,

latitudeNullCnt,

longitudeNullCnt,

streetNullCnt,

-- cityNullCnt,

-- stateNullCnt,

postalcodeNullCnt,

floorNullCnt,

countyNullCnt,

countryNullCnt,

directionsNullCnt,

firePhoneInvalidCnt,

policePhoneInvalidCnt,

powerPhoneInvalidCnt,

powerAcctNumNullCnt,

powerCompanyNullCnt

FROM (SELECT d.area,

d.region,

'ALL' VW\_TYPE, -- 'ALLTEL', --'NONTEL', --

v.domainname,

v.name sitename,

v.site\_inst\_id,

v.state,

v.TYPE category,

NVL2 (v.name, 0, 1) nameNullCnt,

NVL2 (v.site\_inst\_id, 0, 1) siteidNullCnt,

NVL2 (v.latitude, 0, 1) latitudeNullCnt,

NVL2 (v.longitude, 0, 1) longitudeNullCnt,

NVL2 (v.street, 0, 1) streetNullCnt,

-- NVL2 (v.city, 0, 1) cityNullCnt,

-- NVL2 (v.state, 0, 1) stateNullCnt,

NVL2 (v.postalcode, 0, 1) postalcodeNullCnt,

NVL2 (v.FLOOR, 0, 1) floorNullCnt,

NVL2 (v.county, 0, 1) countyNullCnt,

NVL2 (v.country, 0, 1) countryNullCnt,

NVL2 (v.directions, 0, 1) directionsNullCnt,

CASE

WHEN (v.TYPE = 'ENHANCER' OR v.TYPE = 'MICROCELL')

THEN

0

ELSE

isValidPhoneNum (fire\_number)

END

firePhoneInvalidCnt,

CASE

WHEN (v.TYPE = 'ENHANCER' OR v.TYPE = 'MICROCELL')

THEN

0

ELSE

isValidPhoneNum (local\_police\_number)

END

policePhoneInvalidCnt,

CASE

WHEN (v.TYPE = 'ENHANCER' OR v.TYPE = 'MICROCELL')

THEN

0

ELSE

isValidPhoneNum (power\_phone\_number)

END

powerPhoneInvalidCnt,

CASE

WHEN (v.TYPE = 'ENHANCER' OR v.TYPE = 'MICROCELL')

THEN

0

ELSE

NVL2 (power\_account\_number, 0, 1)

END

powerAcctNumNullCnt,

CASE

WHEN (v.TYPE = 'ENHANCER' OR v.TYPE = 'MICROCELL')

THEN

0

ELSE

NVL2 (power\_company, 0, 1)

END

powerCompanyNullCnt,

0

FROM XNG\_REPORTS.XNGSTD\_SUM\_SITE\_ALL\_VW v, -- XNGSTD\_SUM\_SITE\_ALLTEL\_VW v, -- XNGSTD\_SUM\_SITE\_NONTEL\_VW v, --

XNG\_REPORTS.XNGSTD\_MARKET\_DOMAIN\_VW d,

XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES A

WHERE V.DOMAINNAME = D.MARKET\_DOMAIN\_NAME

AND ( ( V.DOMAINNAME = 'NNO\_DOMAIN'

AND A.NAME = 'nno.site.types'

AND A.VALUE LIKE '%,' || V.TYPE || ',%')

OR ( V.DOMAINNAME <> 'NNO\_DOMAIN'

AND A.NAME = 'site.types'

AND A.VALUE LIKE '%,' || V.TYPE || ',%')))

WHERE nameNullCnt = 1

OR siteidNullCnt = 1

OR latitudeNullCnt = 1

OR longitudeNullCnt = 1

OR streetNullCnt = 1

-- OR cityNullCnt = 1

-- OR stateNullCnt = 1

OR postalcodeNullCnt = 1

OR floorNullCnt = 1

OR countyNullCnt = 1

OR countryNullCnt = 1

OR directionsNullCnt = 1

OR firePhoneInvalidCnt = 1

OR policePhoneInvalidCnt = 1

OR powerPhoneInvalidCnt = 1

OR powerAcctNumNullCnt = 1

OR powerCompanyNullCnt = 1;

-- COUNT EACH FAULT PER AREA, REGION

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT;

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT (AREA,

REGION,

VW\_TYPE,

NAME\_NULL\_CNT,

SITEID\_NULL\_CNT,

LATITUDE\_NULL\_CNT,

LONGITUDE\_NULL\_CNT,

STREET\_NULL\_CNT,

-- CITY\_NULL\_CNT,

-- STATE\_NULL\_CNT,

POSTALCODE\_NULL\_CNT,

FLOOR\_NULL\_CNT,

COUNTY\_NULL\_CNT,

COUNTRY\_NULL\_CNT,

DIRECTIONS\_NULL\_CNT,

FIREPHONE\_FAULT\_CNT,

POLICE\_FAULT\_CNT,

POWERPHONE\_FAULT\_CNT,

POWERACCT\_NULL\_CNT,

POWERCOMPANY\_NULL\_CNT,

TOTAL\_FAULT\_REC)

SELECT DR.AREA,

DR.REGION,

NVL (S.VW\_TYPE, 'ALL'),

SUM (NVL (NAME\_NULL, 0)),

SUM (NVL (SITEID\_NULL, 0)),

SUM (NVL (LATITUDE\_NULL, 0)),

SUM (NVL (LONGITUDE\_NULL, 0)),

SUM (NVL (STREET\_NULL, 0)),

-- SUM (NVL(CITY\_NULL,0)),

-- SUM (NVL(STATE\_NULL,0)),

SUM (NVL (POSTALCODE\_NULL, 0)),

SUM (NVL (FLOOR\_NULL, 0)),

SUM (NVL (COUNTY\_NULL, 0)),

SUM (NVL (COUNTRY\_NULL, 0)),

SUM (NVL (DIRECTIONS\_NULL, 0)),

SUM (NVL (FIREPHONE\_FAULT, 0)),

SUM (NVL (POLICEPHONE\_FAULT, 0)),

SUM (NVL (POWERPHONE\_FAULT, 0)),

SUM (NVL (POWERACCT\_NULL, 0)),

SUM (NVL (POWERCOMPANY\_NULL, 0)),

COUNT (S.NAME\_NULL) -- NUM OF FAULT REC

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_DETAIL S,

XNG\_REPORTS.XNGSTD\_MARKET\_DOMAIN\_VW DR

WHERE DR.AREA = S.AREA(+)

AND DR.REGION = S.REGION(+)

AND DR.MARKET\_DOMAIN\_NAME = S.DOMAIN\_NAME(+)

GROUP BY S.VW\_TYPE, DR.AREA, DR.REGION

ORDER BY DR.AREA, DR.REGION;

-- for region has no fault at all, still need record in SITE cnt summary

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT (AREA,

REGION,

VW\_TYPE,

NAME\_NULL\_CNT,

SITEID\_NULL\_CNT,

LATITUDE\_NULL\_CNT,

LONGITUDE\_NULL\_CNT,

STREET\_NULL\_CNT,

-- CITY\_NULL\_CNT,

-- STATE\_NULL\_CNT,

POSTALCODE\_NULL\_CNT,

FLOOR\_NULL\_CNT,

COUNTY\_NULL\_CNT,

COUNTRY\_NULL\_CNT,

DIRECTIONS\_NULL\_CNT,

FIREPHONE\_FAULT\_CNT,

POLICE\_FAULT\_CNT,

POWERPHONE\_FAULT\_CNT,

POWERACCT\_NULL\_CNT,

POWERCOMPANY\_NULL\_CNT,

TOTAL\_FAULT\_REC)

SELECT AREA,

REGION,

'ALL',

0 NAME\_NULL\_CNT,

0 SITEID\_NULL\_CNT,

0 LATITUDE\_NULL\_CNT,

0 LONGITUDE\_NULL\_CNT,

0 STREET\_NULL\_CNT,

-- 0 CITY\_NULL\_CNT,

-- 0 STATE\_NULL\_CNT,

0 POSTALCODE\_NULL\_CNT,

0 FLOOR\_NULL\_CNT,

0 COUNTY\_NULL\_CNT,

0 COUNTRY\_NULL\_CNT,

0 DIRECTIONS\_NULL\_CNT,

0 FIREPHONE\_FAULT\_CNT,

0 POLICE\_FAULT\_CNT,

0 POWERPHONE\_FAULT\_CNT,

0 POWERACCT\_NULL\_CNT,

0 POWERCOMPANY\_NULL\_CNT,

0 TOTAL\_FAULT\_REC

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

WHERE NOT EXISTS

(SELECT 1

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT B

WHERE A.AREA = B.AREA

AND A.REGION = B.REGION

AND B.VW\_TYPE = 'ALL')

AND A.REGION <> 'Total Compliance';

-- COUNT TOTAL RECORDS PER AREA, REGION

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT A

SET A.TOTAL\_REC =

(SELECT CNT

FROM ( SELECT D.AREA, D.REGION, COUNT (\*) CNT

FROM XNG\_REPORTS.XNGSTD\_SUM\_SITE\_ALL\_VW v,

XNG\_REPORTS.XNGSTD\_MARKET\_DOMAIN\_VW D,

XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES A

WHERE V.DOMAINNAME = D.MARKET\_DOMAIN\_NAME

AND ( ( V.DOMAINNAME = 'NNO\_DOMAIN'

AND A.NAME = 'nno.site.types'

AND A.VALUE LIKE

'%,' || V.TYPE || ',%')

OR ( V.DOMAINNAME <> 'NNO\_DOMAIN'

AND A.NAME = 'site.types'

AND A.VALUE LIKE

'%,' || V.TYPE || ',%'))

GROUP BY D.AREA, D.REGION) B

WHERE A.AREA = B.AREA AND A.REGION = B.REGION)

WHERE A.VW\_TYPE = 'ALL';

-- ADD all fault count to TOTAL\_FAULT\_CNT

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT a

SET TOTAL\_FAULT\_CNT =

(SELECT NAME\_NULL\_CNT

+ SITEID\_NULL\_CNT

+ LATITUDE\_NULL\_CNT

+ LONGITUDE\_NULL\_CNT

+ STREET\_NULL\_CNT

-- + CITY\_NULL\_CNT -- not counted in XngSiteAudit.java

-- + STATE\_NULL\_CNT -- not counted in XngSiteAudit.java

+ POSTALCODE\_NULL\_CNT

+ FLOOR\_NULL\_CNT

+ COUNTY\_NULL\_CNT

+ COUNTRY\_NULL\_CNT

+ DIRECTIONS\_NULL\_CNT

+ FIREPHONE\_FAULT\_CNT

+ POLICE\_FAULT\_CNT

+ POWERPHONE\_FAULT\_CNT

+ POWERACCT\_NULL\_CNT

+ POWERCOMPANY\_NULL\_CNT

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT b

WHERE a.AREA = B.AREA

AND A.REGION = B.REGION

AND A.VW\_TYPE = B.VW\_TYPE);

-- ADD entries for subtotal of per AREA

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT (AREA,

REGION,

VW\_TYPE,

NAME\_NULL\_CNT,

SITEID\_NULL\_CNT,

LATITUDE\_NULL\_CNT,

LONGITUDE\_NULL\_CNT,

STREET\_NULL\_CNT,

-- CITY\_NULL\_CNT,

-- STATE\_NULL\_CNT,

POSTALCODE\_NULL\_CNT,

FLOOR\_NULL\_CNT,

COUNTY\_NULL\_CNT,

COUNTRY\_NULL\_CNT,

DIRECTIONS\_NULL\_CNT,

FIREPHONE\_FAULT\_CNT,

POLICE\_FAULT\_CNT,

POWERPHONE\_FAULT\_CNT,

POWERACCT\_NULL\_CNT,

POWERCOMPANY\_NULL\_CNT,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT,

TOTAL\_REC)

SELECT AREA,

NULL REGION,

VW\_TYPE,

SUM (NVL (NAME\_NULL\_CNT, 0)),

SUM (NVL (SITEID\_NULL\_CNT, 0)),

SUM (NVL (LATITUDE\_NULL\_CNT, 0)),

SUM (NVL (LONGITUDE\_NULL\_CNT, 0)),

SUM (NVL (STREET\_NULL\_CNT, 0)),

-- SUM (NVL(CITY\_NULL\_CNT,0)),

-- SUM (NVL(STATE\_NULL\_CNT,0)),

SUM (NVL (POSTALCODE\_NULL\_CNT, 0)),

SUM (NVL (FLOOR\_NULL\_CNT, 0)),

SUM (NVL (COUNTY\_NULL\_CNT, 0)),

SUM (NVL (COUNTRY\_NULL\_CNT, 0)),

SUM (NVL (DIRECTIONS\_NULL\_CNT, 0)),

SUM (NVL (FIREPHONE\_FAULT\_CNT, 0)),

SUM (NVL (POLICE\_FAULT\_CNT, 0)),

SUM (NVL (POWERPHONE\_FAULT\_CNT, 0)),

SUM (NVL (POWERACCT\_NULL\_CNT, 0)),

SUM (NVL (POWERCOMPANY\_NULL\_CNT, 0)),

SUM (NVL (TOTAL\_FAULT\_REC, 0)),

SUM (NVL (TOTAL\_FAULT\_CNT, 0)),

SUM (NVL (TOTAL\_REC, 0))

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT

GROUP BY VW\_TYPE, AREA

ORDER BY VW\_TYPE, AREA;

-- calculate SITE fault percentage per area, region

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET SITE\_FAULT\_PCT =

(SELECT TO\_CHAR (

1.00000 - (TOTAL\_FAULT\_CNT / (TOTAL\_REC \* 16)),

'FM9.99990')

FROM XNGSTD\_FAULT\_SITE\_CNT B

WHERE B.REGION IS NOT NULL

AND A.AREA = B.AREA

AND A.REGION = B.REGION

AND TOTAL\_REC > 0

AND B.VW\_TYPE = 'ALL')

WHERE REGION <> 'Total Compliance';

-- calculate SITE fault percentage for subtotal per area

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET SITE\_FAULT\_PCT =

(SELECT TO\_CHAR (

1.00000 - (TOTAL\_FAULT\_CNT / (TOTAL\_REC \* 16)),

'FM9.99990')

FROM XNGSTD\_FAULT\_SITE\_CNT B

WHERE B.VW\_TYPE = 'ALL'

AND TOTAL\_REC > 0

AND B.REGION IS NULL

AND A.AREA = B.AREA || ' Area Subtotal')

WHERE REGION = 'Total Compliance';

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET SITE\_FAULT\_PCT =

(SELECT TO\_CHAR (1.00000, 'FM9.99990') FROM DUAL)

WHERE SITE\_FAULT\_PCT IS NULL;

COMMIT;

END;

PROCEDURE XNGSTD\_AUDIT\_EQUIP\_PROC

IS

sqlStmt VARCHAR2 (32000);

BEGIN

-- ADD ALL FAULT EQUIPMENT DETAIL RECORDS

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_DETAIL;

INSERT INTO XNGSTD\_FAULT\_EQUIP\_DETAIL (VW\_TYPE,

DOMAIN\_NAME,

AREA,

REGION,

DESCR,

EQUIP\_INST\_ID,

SITE\_INST\_ID,

SITE\_HUM\_ID,

SITE\_STATE,

CATEGORY,

DESCR\_FAULT,

EQ\_CLASS\_FAULT,

FAULT\_CNT)

WITH EQPM\_FAULT

AS (SELECT B.DOMAIN\_NAME,

B.DESCR,

B.EQUIP\_INST\_ID,

B.SITE\_INST\_ID,

B.SITE\_HUM\_ID,

B.SITE\_STATE,

B.CATEGORY,

NVL2 (

B.DESCR,

NVL2 (

A.VALUE,

(CASE

WHEN REGEXP\_LIKE (

B.DESCR,

'^(\w+ )\*'

|| REPLACE (A.VALUE, '\\', '\')

|| '(( )\*\[.\*\])?') -- EquipmentValidator.init()

THEN -- matches pattern, ok

CASE

WHEN REGEXP\_LIKE (

B.DESCR,

'^.+]([^]])+$')

THEN

2

ELSE

0 -- but not have any char after last right bracket ]

END

ELSE

2 -- DESCR(equipName) has ID\_bad\_format

END),

0), -- no pattern for category, ok

1) -- DESCR(equipName) is empty ID\_missing

DESCR\_FAULT,

(CASE

WHEN B.EQ\_CLASS <> 'C' THEN NVL2 (B.MODEL, 0, 1)

ELSE 0

END)

EQ\_CLASS\_FAULT

FROM XNG\_REPORTS.XNGSTD\_SUM\_EQUIP\_ALL\_VW B,

XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES A

WHERE B.CATEGORY <> 'SMART JACK'

AND 'equipment.types.'

|| REPLACE (B.CATEGORY, ' ', '\_')

|| '.pattern' = A.NAME(+))

SELECT 'ALL',

V.DOMAIN\_NAME,

D.AREA,

D.REGION,

V.DESCR,

V.EQUIP\_INST\_ID,

V.SITE\_INST\_ID,

V.SITE\_HUM\_ID,

V.SITE\_STATE,

V.CATEGORY,

DESCR\_FAULT,

EQ\_CLASS\_FAULT,

( (CASE WHEN (DESCR\_FAULT = 2) THEN 1 ELSE DESCR\_FAULT END)

+ EQ\_CLASS\_FAULT)

FAULT\_CNT

FROM EQPM\_FAULT V, XNG\_REPORTS.XNGSTD\_MARKET\_DOMAIN\_VW D

WHERE (DESCR\_FAULT <> 0 OR V.EQ\_CLASS\_FAULT <> 0)

AND V.DOMAIN\_NAME = D.MARKET\_DOMAIN\_NAME;

--56,809 RECORDS

-- ADDING ALL COUNT PER AREA,REGION

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_CNT;

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_CNT (VW\_TYPE,

AREA,

REGION,

DESCR\_FAULT\_CNT,

EQ\_CLASS\_FAULT\_CNT,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT)

SELECT VW\_TYPE,

AREA,

REGION,

SUM (

(CASE WHEN (DESCR\_FAULT = 2) THEN 1 ELSE DESCR\_FAULT END))

DESCR\_FAULT\_CNT,

SUM (EQ\_CLASS\_FAULT) EQ\_CLASS\_FAULT\_CNT,

COUNT (1),

SUM (FAULT\_CNT) TOTAL\_FAULT\_CNT

FROM XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_DETAIL

GROUP BY VW\_TYPE, AREA, REGION;

-- for region has no fault at all, still need record in EQUIPMENT cnt summary

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_CNT (VW\_TYPE,

AREA,

REGION,

DESCR\_FAULT\_CNT,

EQ\_CLASS\_FAULT\_CNT,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT)

SELECT 'ALL',

AREA,

REGION,

0 DESCR\_FAULT\_CNT,

0 EQ\_CLASS\_FAULT\_CNT,

0,

0 TOTAL\_FAULT\_CNT

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

WHERE NOT EXISTS

(SELECT 1

FROM XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_DETAIL B

WHERE A.AREA = B.AREA

AND A.REGION = B.REGION

AND B.VW\_TYPE = 'ALL')

AND A.REGION <> 'Total Compliance';

-- TOTAL RECORD COUNT (INCLUDE NON-FAULT) PER REGION

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_CNT A

SET TOTAL\_REC =

(SELECT CNT

FROM ( SELECT AREA, REGION, COUNT (\*) CNT

FROM XNG\_REPORTS.XNGSTD\_SUM\_EQUIP\_ALL\_VW V,

XNG\_REPORTS.XNGSTD\_MARKET\_DOMAIN\_VW D

WHERE V.DOMAIN\_NAME = D.MARKET\_DOMAIN\_NAME

AND V.CATEGORY <> 'SMART JACK'

GROUP BY AREA, REGION) B

WHERE B.AREA = A.AREA AND B.REGION = A.REGION)

WHERE A.VW\_TYPE = 'ALL';

-- ADD ENTRY PER AREA

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_CNT (VW\_TYPE,

AREA,

REGION,

TOTAL\_REC,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT,

DESCR\_FAULT\_CNT,

EQ\_CLASS\_FAULT\_CNT)

SELECT VW\_TYPE,

AREA,

NULL REGION,

SUM (TOTAL\_REC),

SUM (TOTAL\_FAULT\_REC),

SUM (TOTAL\_FAULT\_CNT),

SUM (DESCR\_FAULT\_CNT),

SUM (EQ\_CLASS\_FAULT\_CNT)

FROM XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_CNT

GROUP BY VW\_TYPE, AREA;

-- calculate EQUIP fault percentage per AREA, REGION

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET EQUIP\_FAULT\_PCT =

(SELECT TO\_CHAR (

1.00000 - (TOTAL\_FAULT\_CNT / (TOTAL\_REC \* 2)),

'FM9.99990')

FROM XNGSTD\_FAULT\_EQUIP\_CNT B

WHERE B.REGION IS NOT NULL

AND A.AREA = B.AREA

AND A.REGION = B.REGION

AND B.VW\_TYPE = 'ALL')

WHERE REGION <> 'Total Compliance';

-- calculate EQUIP fault percentage for subtotal per AREA

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET EQUIP\_FAULT\_PCT =

(SELECT TO\_CHAR (

1.00000 - (TOTAL\_FAULT\_CNT / (TOTAL\_REC \* 2)),

'FM9.99990')

FROM XNGSTD\_FAULT\_EQUIP\_CNT B

WHERE B.VW\_TYPE = 'ALL'

AND B.REGION IS NULL

AND A.AREA = B.AREA || ' Area Subtotal')

WHERE REGION = 'Total Compliance';

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET EQUIP\_FAULT\_PCT =

(SELECT TO\_CHAR (1.00000, 'FM9.99990') FROM DUAL)

WHERE EQUIP\_FAULT\_PCT IS NULL;

COMMIT;

END;

PROCEDURE XNGSTD\_AUDIT\_PATH\_PROC

IS

sqlStmt VARCHAR2 (32000);

BEGIN

DELETE FROM XNG\_REPORTS.XNGSTD\_SWITCH\_VENDORS;

INSERT INTO XNG\_REPORTS.XNGSTD\_SWITCH\_VENDORS (VENDOR, SWITCH\_ID)

SELECT DISTINCT VENDOR, SWITCH\_ID FROM XNGSTD\_SWITCH\_VENDOR\_VW;

COMMIT;

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_DETAIL;

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_DETAIL (VW\_TYPE,

AREA,

REGION,

CIRC\_PATH\_HUM\_ID,

CIRC\_PATH\_REV\_NBR,

CATEGORY,

SUBCATEGORY,

PATTERN,

DOMAIN\_NAME,

DOMAIN\_INST\_ID,

BANDWIDTH,

STATUS,

A\_SIDE\_SITE\_ID,

Z\_SIDE\_SITE\_ID,

NBR\_CHANNELS,

NBR\_CHAN\_ASSIGNED,

TOPOLOGY,

IS\_TRUNK\_GROUP,

CIRC\_PATH\_INST\_ID,

A\_SIDE\_SITE\_STATE,

DOM\_IP,

PATH\_NUM\_ID\_FAULT,

ASIDE\_FAULT,

ZSIDE\_FAULT,

DOM\_IP\_FAULT,

TOTAL\_FAULT)

WITH PATH\_SUBCATEGORY

AS (SELECT CIRC\_PATH\_HUM\_ID,

CIRC\_PATH\_REV\_NBR,

TYPE\_NAME,

DOMAIN\_NAME,

DOMAIN\_INST\_ID,

BANDWIDTH,

STATUS,

A\_SIDE\_SITE\_ID,

Z\_SIDE\_SITE\_ID,

NBR\_CHANNELS,

NBR\_CHAN\_ASSIGNED,

TOPOLOGY,

IS\_TRUNK\_GROUP,

CIRC\_PATH\_INST\_ID,

A\_SIDE\_SITE\_STATE,

DOM\_IP,

XNGSTD\_AUDIT\_BATCH\_PKG.GET\_SUBCATEGORY (

CIRC\_PATH\_HUM\_ID,

DOM\_IP,

TYPE\_NAME)

SUBCATEGORY

FROM XNG\_REPORTS.XNGSTD\_SUM\_PATH\_ALL\_VW),

PATH\_PATTERN

AS (SELECT PS.\*,

(CASE

WHEN ( (SUBCATEGORY IS NOT NULL)

AND (SUBCATEGORY <> 'BADSWITCH'))

THEN

SUBCATEGORY

ELSE

TYPE\_NAME

END)

CATEGORY,

(CASE

WHEN (SUBCATEGORY = 'BADSWITCH')

THEN

NULL

WHEN (SUBCATEGORY IS NOT NULL)

THEN

XNGSTD\_AUDIT\_BATCH\_PKG.GET\_PATH\_PATTERN (

SUBCATEGORY)

ELSE

XNGSTD\_AUDIT\_BATCH\_PKG.GET\_PATH\_PATTERN (

TYPE\_NAME)

END)

PATTERN

FROM PATH\_SUBCATEGORY PS),

PATH\_FAULT

AS (SELECT A.\*,

(CASE

WHEN (A.CIRC\_PATH\_HUM\_ID IS NULL)

THEN

1

WHEN ( (A.SUBCATEGORY IS NULL)

AND (PATTERN IS NULL))

THEN

0

WHEN (SUBCATEGORY = 'BADSWITCH')

THEN

4

WHEN ( SUBCATEGORY LIKE '%CDMA%'

AND ( A.CIRC\_PATH\_HUM\_ID LIKE

'%CHTNSC01%'

OR A.CIRC\_PATH\_HUM\_ID LIKE

'%RLGHNC01%'))

THEN

(CASE

WHEN ( (PATTERN IS NULL)

OR REGEXP\_LIKE (A.CIRC\_PATH\_HUM\_ID,

PATTERN))

THEN

0 -- matched

ELSE

XNGSTD\_AUDIT\_BATCH\_PKG.CHECK\_ID (

A.CIRC\_PATH\_HUM\_ID,

XNGSTD\_AUDIT\_BATCH\_PKG.GET\_PATH\_PATTERN (

'CDMANORTEL'))

END)

ELSE

XNGSTD\_AUDIT\_BATCH\_PKG.CHECK\_ID (

A.CIRC\_PATH\_HUM\_ID,

PATTERN)

END)

PATH\_NUM\_ID\_FAULT,

NVL2 (A\_SIDE\_SITE\_ID, 0, 1) ASIDE\_FAULT,

NVL2 (Z\_SIDE\_SITE\_ID, 0, 1) ZSIDE\_FAULT,

(CASE

WHEN ( DOM\_IP IS NOT NULL

AND DOM\_IP = 'not present')

THEN

1

ELSE

0

END)

DOM\_IP\_FAULT

FROM PATH\_PATTERN A)

SELECT 'ALL',

D.AREA,

D.REGION,

CIRC\_PATH\_HUM\_ID,

CIRC\_PATH\_REV\_NBR,

CATEGORY,

B.SUBCATEGORY,

B.PATTERN,

B.DOMAIN\_NAME,

DOMAIN\_INST\_ID,

BANDWIDTH,

STATUS,

A\_SIDE\_SITE\_ID,

Z\_SIDE\_SITE\_ID,

NBR\_CHANNELS,

NBR\_CHAN\_ASSIGNED,

TOPOLOGY,

IS\_TRUNK\_GROUP,

CIRC\_PATH\_INST\_ID,

A\_SIDE\_SITE\_STATE,

DOM\_IP,

PATH\_NUM\_ID\_FAULT,

ASIDE\_FAULT,

ZSIDE\_FAULT,

DOM\_IP\_FAULT,

( (CASE WHEN PATH\_NUM\_ID\_FAULT = 0 THEN 0 ELSE 1 END)

+ ASIDE\_FAULT

+ ZSIDE\_FAULT

+ DOM\_IP\_FAULT)

TOTAL\_FAULT

FROM PATH\_FAULT B, XNG\_REPORTS.XNGSTD\_MARKET\_DOMAIN\_VW D

WHERE ( PATH\_NUM\_ID\_FAULT

+ ASIDE\_FAULT

+ ZSIDE\_FAULT

+ DOM\_IP\_FAULT) > 0

AND B.DOMAIN\_NAME = D.MARKET\_DOMAIN\_NAME;

COMMIT;

-- POPULATE DATA FOR PATH SUMMARY TABLE

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT;

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT (VW\_TYPE,

AREA,

REGION,

PATH\_NUM\_ID\_FAULT\_CNT,

ASIDE\_FAULT\_CNT,

ZSIDE\_FAULT\_CNT,

DOM\_IP\_FAULT\_CNT,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT)

SELECT VW\_TYPE,

AREA,

REGION,

SUM ( (CASE WHEN (PATH\_NUM\_ID\_FAULT <> 0) THEN 1 ELSE 0 END))

PATH\_NUM\_ID\_FAULT\_CNT,

SUM (ASIDE\_FAULT) ASIDE\_FAULT\_CNT,

SUM (ZSIDE\_FAULT) ZSIDE\_FAULT\_CNT,

SUM (DOM\_IP\_FAULT) DOM\_IP\_FAULT\_CNT,

COUNT (1),

SUM (TOTAL\_FAULT) TOTAL\_FAULT\_CNT

FROM XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_DETAIL

GROUP BY VW\_TYPE, AREA, REGION

ORDER BY VW\_TYPE, AREA, REGION;

-- for region has no fault at all, still need record in PATH cnt summary

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT (VW\_TYPE,

AREA,

REGION,

PATH\_NUM\_ID\_FAULT\_CNT,

ASIDE\_FAULT\_CNT,

ZSIDE\_FAULT\_CNT,

DOM\_IP\_FAULT\_CNT,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT)

SELECT 'ALL',

AREA,

REGION,

0 PATH\_NUM\_ID\_FAULT\_CNT,

0 ASIDE\_FAULT\_CNT,

0 ZSIDE\_FAULT\_CNT,

0 DOM\_IP\_FAULT\_CNT,

0 TOTAL\_FAULT\_REC,

0 TOTAL\_FAULT\_CNT

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

WHERE NOT EXISTS

(SELECT 1

FROM XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT B

WHERE A.AREA = B.AREA

AND A.REGION = B.REGION

AND B.VW\_TYPE = 'ALL')

AND A.REGION <> 'Total Compliance';

-- TOTAL RECORD COUNT, FAULT OR NOT PER AREA/REGION

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT A

SET TOTAL\_REC =

(SELECT CNT

FROM ( SELECT AREA, REGION, COUNT (\*) CNT

FROM XNG\_REPORTS.XNGSTD\_SUM\_PATH\_ALL\_VW V,

XNG\_REPORTS.XNGSTD\_MARKET\_DOMAIN\_VW D

WHERE V.DOMAIN\_NAME = D.MARKET\_DOMAIN\_NAME

GROUP BY AREA, REGION) B

WHERE B.AREA = A.AREA AND B.REGION = A.REGION)

WHERE A.VW\_TYPE = 'ALL';

-- PATH SUMMARIZE COUNT PER AREA

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT (VW\_TYPE,

AREA,

REGION,

TOTAL\_REC,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT,

PATH\_NUM\_ID\_FAULT\_CNT,

ASIDE\_FAULT\_CNT,

ZSIDE\_FAULT\_CNT,

DOM\_IP\_FAULT\_CNT)

SELECT VW\_TYPE,

AREA,

NULL REGION,

SUM (TOTAL\_REC),

SUM (TOTAL\_FAULT\_REC),

SUM (TOTAL\_FAULT\_CNT),

SUM (PATH\_NUM\_ID\_FAULT\_CNT),

SUM (ASIDE\_FAULT\_CNT),

SUM (ZSIDE\_FAULT\_CNT),

SUM (DOM\_IP\_FAULT\_CNT)

FROM XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT

GROUP BY VW\_TYPE, AREA;

-- calculate PATH fault percentage per AREA, REGION

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET PATH\_FAULT\_PCT =

(SELECT TO\_CHAR (

1.00000 - (TOTAL\_FAULT\_CNT / (TOTAL\_REC \* 4)),

'FM9.99990')

FROM XNGSTD\_FAULT\_PATH\_CNT B

WHERE B.REGION IS NOT NULL

AND A.AREA = B.AREA

AND A.REGION = B.REGION

AND B.VW\_TYPE = 'ALL')

WHERE REGION <> 'Total Compliance';

-- SUM PATH PERCENTAGE calculate PATH fault percentage for subtotal per AREA

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET PATH\_FAULT\_PCT =

(SELECT TO\_CHAR (

1.00000 - (TOTAL\_FAULT\_CNT / (TOTAL\_REC \* 4)),

'FM9.99990')

FROM XNGSTD\_FAULT\_PATH\_CNT B

WHERE B.VW\_TYPE = 'ALL'

AND B.REGION IS NULL

AND A.AREA = B.AREA || ' Area Subtotal')

WHERE REGION = 'Total Compliance';

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET PATH\_FAULT\_PCT =

(SELECT TO\_CHAR (1.00000, 'FM9.99990') FROM DUAL)

WHERE PATH\_FAULT\_PCT IS NULL;

END;

PROCEDURE XNGSTD\_AUDIT\_SEGMENT\_PROC

IS

sqlStmt VARCHAR2 (32000);

BEGIN

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_DETAIL;

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_DETAIL (

VW\_TYPE,

AREA,

REGION,

CIRC\_HUM\_ID,

ASIDESITESTATE,

DOMAIN\_NAME,

CIRC\_INST\_ID,

XNG\_TYPE,

BILL\_DATE\_MONTH,

BILL\_DATE\_YEAR,

DISCONNECT\_PON,

DISCONNECT\_ORDER\_DATE,

DISCONNECT\_ORDER\_NUMBER,

DISCONNECT\_DATE,

PON\_FAULT,

ORDER\_DATE\_FAULT,

ORDER\_NUMBER\_FAULT,

DATE\_FAULT,

TOTAL\_FAULT)

WITH SEGMENT\_FAULT

AS (SELECT V.\*,

NVL2 (DISCONNECT\_PON, 0, 1) PON\_FAULT,

NVL2 (DISCONNECT\_ORDER\_DATE, 0, 1) ORDER\_DATE\_FAULT,

NVL2 (DISCONNECT\_ORDER\_NUMBER, 0, 1)

ORDER\_NUMBER\_FAULT,

NVL2 (DISCONNECT\_DATE, 0, 1) DATE\_FAULT

FROM XNG\_REPORTS.XNGSTD\_SUM\_SGMT\_ALL\_VW V)

SELECT 'ALL',

D.AREA,

D.REGION,

CIRC\_HUM\_ID,

ASIDESITESTATE,

A.DOMAIN\_NAME,

CIRC\_INST\_ID,

XNG\_TYPE,

BILL\_DATE\_MONTH,

BILL\_DATE\_YEAR,

DISCONNECT\_PON,

DISCONNECT\_ORDER\_DATE,

DISCONNECT\_ORDER\_NUMBER,

DISCONNECT\_DATE,

PON\_FAULT,

ORDER\_DATE\_FAULT,

ORDER\_NUMBER\_FAULT,

DATE\_FAULT,

( PON\_FAULT

+ ORDER\_DATE\_FAULT

+ ORDER\_NUMBER\_FAULT

+ DATE\_FAULT)

TOTAL\_FAULT

FROM SEGMENT\_FAULT A, XNG\_REPORTS.XNGSTD\_REGION\_DOMAIN\_VW D

WHERE ( PON\_FAULT

+ ORDER\_DATE\_FAULT

+ ORDER\_NUMBER\_FAULT

+ DATE\_FAULT) > 0

AND A.DOMAIN\_NAME = D.REGION\_DOMAIN\_NAME;

DELETE FROM XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT;

-- SUM COUNT PER REGION

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT (

VW\_TYPE,

AREA,

REGION,

PON\_FAULT\_CNT,

ORDER\_DATE\_FAULT\_CNT,

ORDER\_NUMBER\_FAULT\_CNT,

DATE\_FAULT\_CNT,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT,

TOTAL\_REC)

SELECT VW\_TYPE,

AREA,

REGION,

SUM (PON\_FAULT) PON\_FAULT\_CNT,

SUM (ORDER\_DATE\_FAULT) ORDER\_DATE\_FAULT\_CNT,

SUM (ORDER\_NUMBER\_FAULT) ORDER\_NUMBER\_FAULT\_CNT,

SUM (DATE\_FAULT) DATE\_FAULT\_CNT,

COUNT (1) TOTAL\_FAULT\_REC,

SUM (TOTAL\_FAULT) TOTAL\_FAULT\_CNT,

0 TOTAL\_REC

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_DETAIL

GROUP BY VW\_TYPE, AREA, REGION

ORDER BY VW\_TYPE, AREA, REGION;

-- for region has no fault at all, still need record in SEGMENT cnt summary

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT (

VW\_TYPE,

AREA,

REGION,

PON\_FAULT\_CNT,

ORDER\_DATE\_FAULT\_CNT,

ORDER\_NUMBER\_FAULT\_CNT,

DATE\_FAULT\_CNT,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT,

TOTAL\_REC)

SELECT 'ALL',

AREA,

REGION,

0 PON\_FAULT\_CNT,

0 ORDER\_DATE\_FAULT\_CNT,

0 ORDER\_NUMBER\_FAULT\_CNT,

0 DATE\_FAULT\_CNT,

0 TOTAL\_FAULT\_REC,

0 TOTAL\_FAULT\_CNT,

0 TOTAL\_REC

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

WHERE NOT EXISTS

(SELECT 1

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT B

WHERE A.AREA = B.AREA

AND A.REGION = B.REGION

AND B.VW\_TYPE = 'ALL')

AND A.REGION <> 'Total Compliance';

-- COUNT TOTAL RECORD

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT A

SET TOTAL\_REC =

(SELECT CNT

FROM ( SELECT AREA, REGION, COUNT (\*) CNT

FROM XNG\_REPORTS.XNGSTD\_SUM\_SGMT\_ALL\_VW V,

XNG\_REPORTS.XNGSTD\_REGION\_DOMAIN\_VW D

WHERE V.DOMAIN\_NAME = D.REGION\_DOMAIN\_NAME

GROUP BY AREA, REGION) B

WHERE B.AREA = A.AREA AND B.REGION = A.REGION)

WHERE A.VW\_TYPE = 'ALL';

-- SUM COUNT PER AREA

INSERT INTO XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT (

VW\_TYPE,

AREA,

REGION,

TOTAL\_REC,

TOTAL\_FAULT\_REC,

TOTAL\_FAULT\_CNT,

PON\_FAULT\_CNT,

ORDER\_DATE\_FAULT\_CNT,

ORDER\_NUMBER\_FAULT\_CNT,

DATE\_FAULT\_CNT)

SELECT VW\_TYPE,

AREA,

NULL REGION,

SUM (TOTAL\_REC),

SUM (TOTAL\_FAULT\_REC),

SUM (TOTAL\_FAULT\_CNT),

SUM (PON\_FAULT\_CNT),

SUM (ORDER\_DATE\_FAULT\_CNT),

SUM (ORDER\_NUMBER\_FAULT\_CNT),

SUM (DATE\_FAULT\_CNT)

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT

GROUP BY VW\_TYPE, AREA;

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT A

SET TOTAL\_REC = 0

WHERE TOTAL\_REC IS NULL;

-- calculate SEGMENT fault percentage per AREA, REGION

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET SEG\_FAULT\_PCT =

(SELECT TO\_CHAR (

1.00000 - (TOTAL\_FAULT\_CNT / (TOTAL\_REC \* 4)),

'FM9.99990')

FROM XNGSTD\_FAULT\_SEGMENT\_CNT B

WHERE B.REGION IS NOT NULL

AND A.AREA = B.AREA

AND A.REGION = B.REGION

AND TOTAL\_REC IS NOT NULL

AND TOTAL\_REC > 0

AND B.VW\_TYPE = 'ALL')

WHERE REGION <> 'Total Compliance';

-- SUM SEGMENT PERCENTAGE

-- calculate PATH fault percentage for subtotal per AREA

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET SEG\_FAULT\_PCT =

(SELECT TO\_CHAR (

1.00000 - (TOTAL\_FAULT\_CNT / (TOTAL\_REC \* 4)),

'FM9.99990')

FROM XNGSTD\_FAULT\_SEGMENT\_CNT B

WHERE B.VW\_TYPE = 'ALL'

AND TOTAL\_REC IS NOT NULL

AND TOTAL\_REC > 0

AND B.REGION IS NULL

AND A.AREA = B.AREA || ' Area Subtotal')

WHERE REGION = 'Total Compliance';

--NO FAULT, MEAN 100 PCT OK

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM A

SET SEG\_FAULT\_PCT =

(SELECT TO\_CHAR (1.00000, 'FM9.99990') FROM DUAL)

WHERE SEG\_FAULT\_PCT IS NULL;

COMMIT;

END;

PROCEDURE XNGSTD\_AUDIT\_POST\_PROC

IS

sqlStmt VARCHAR2 (32000);

BEGIN

-- execute at last

UPDATE XNG\_REPORTS.XNGSTD\_FAULT\_SUM S

SET OVERALL\_FAULT\_PCT =

(WITH TOTAL\_FAULT

AS (SELECT S.AREA,

S.REGION,

( NVL (A.TOTAL\_FAULT\_CNT, 0)

+ NVL (B.TOTAL\_FAULT\_CNT, 0)

+ NVL (C.TOTAL\_FAULT\_CNT, 0)

+ NVL (D.TOTAL\_FAULT\_CNT, 0))

/ ( NVL (A.TOTAL\_REC, 0) \* 16

+ NVL (B.TOTAL\_REC, 0) \* 2

+ NVL (C.TOTAL\_REC, 0) \* 4

+ NVL (D.TOTAL\_REC, 0) \* 4)

TOTAL\_PCT

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SUM S,

XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT A,

XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_CNT B,

XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT C,

XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT D

WHERE S.AREA = A.AREA(+)

AND S.REGION = A.REGION(+)

AND S.AREA = B.AREA(+)

AND S.REGION = B.REGION(+)

AND S.AREA = C.AREA(+)

AND S.REGION = C.REGION(+)

AND S.AREA = D.AREA(+)

AND S.REGION = D.REGION(+)

AND S.REGION <> 'Total Compliance'

UNION

SELECT S.AREA,

S.REGION,

( NVL (A.TOTAL\_FAULT\_CNT, 0)

+ NVL (B.TOTAL\_FAULT\_CNT, 0)

+ NVL (C.TOTAL\_FAULT\_CNT, 0)

+ NVL (D.TOTAL\_FAULT\_CNT, 0))

/ ( NVL (A.TOTAL\_REC, 0) \* 16

+ NVL (B.TOTAL\_REC, 0) \* 2

+ NVL (C.TOTAL\_REC, 0) \* 4

+ NVL (D.TOTAL\_REC, 0) \* 4)

TOTAL\_PCT

FROM XNG\_REPORTS.XNGSTD\_FAULT\_SUM S,

XNG\_REPORTS.XNGSTD\_FAULT\_SITE\_CNT A,

XNG\_REPORTS.XNGSTD\_FAULT\_EQUIP\_CNT B,

XNG\_REPORTS.XNGSTD\_FAULT\_PATH\_CNT C,

XNG\_REPORTS.XNGSTD\_FAULT\_SEGMENT\_CNT D

WHERE S.AREA = (A.AREA(+) || ' Area Subtotal')

AND (A.REGION IS NULL)

AND S.AREA = (B.AREA(+) || ' Area Subtotal')

AND (B.REGION IS NULL)

AND S.AREA = (C.AREA(+) || ' Area Subtotal')

AND (C.REGION IS NULL)

AND S.AREA = (D.AREA(+) || ' Area Subtotal')

AND (D.REGION IS NULL)

AND S.REGION = 'Total Compliance')

SELECT TO\_CHAR ( (1.00000 - TOTAL\_PCT), 'FM9.99990')

FROM TOTAL\_FAULT T

WHERE S.AREA = T.AREA AND S.REGION = T.REGION);

COMMIT;

END;

FUNCTION GET\_SUBCATEGORY (p\_CIRC\_PATH\_HUM\_ID VARCHAR2,

p\_DOM\_IP VARCHAR2,

p\_type\_name VARCHAR2)

RETURN VARCHAR2

IS

v\_switchName VARCHAR2 (100) := NULL;

v\_vendor VARCHAR2 (100) := NULL;

v\_subcategory VARCHAR2 (100) := NULL;

v\_pattern VARCHAR2 (100) := NULL;

v\_pathTypes VARCHAR2 (2000) := NULL;

v\_typeName VARCHAR2 (100) := NULL;

BEGIN

v\_subcategory := NULL;

IF (p\_CIRC\_PATH\_HUM\_ID IS NULL)

THEN

RETURN v\_subcategory;

END IF;

BEGIN

SELECT REGEXP\_REPLACE (P.VALUE, '[ ]', '')

INTO v\_pathTypes

FROM XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES P

WHERE P.NAME = 'path.types';

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_pathTypes := '';

END;

v\_typeName := REGEXP\_REPLACE (p\_type\_name, '[ ]', '');

IF ( v\_pathTypes = ''

OR NOT REGEXP\_LIKE (',' || v\_pathTypes || ',',

',' || v\_typeName || ',')) -- path.types not contains p\_type\_name

THEN

RETURN v\_subcategory;

END IF;

BEGIN

SELECT P.VALUE

INTO v\_pattern

FROM XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES P

WHERE P.NAME = 'path.types.' || v\_typeName || '.pattern';

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_pattern := '';

END;

IF (v\_pattern = '') -- path.types.typename.pattern not existin in property

THEN

RETURN v\_subcategory;

END IF;

IF ( (p\_DOM\_IP IS NOT NULL AND p\_DOM\_IP NOT LIKE '%not present%')

AND (p\_type\_name = 'EVDO' OR p\_type\_name = '3G'))

THEN -- XngPathAudit.getResults()

RETURN p\_type\_name || 'NORTEL'; -- EVDO/X3G.getSubcategoryForNortel()->SubcategoryValidator.getSubcategory(id,category,dom\_ip\_node)

END IF;

IF (NOT ( p\_type\_name = 'AMPS'

OR p\_type\_name = 'CDMA'

OR p\_type\_name = 'CDMALUCENT'

OR p\_type\_name = 'CDMAMOTOROLA'

OR p\_type\_name = 'CDMANORTEL'

OR p\_type\_name = 'CDMAOTHER'

OR p\_type\_name = 'CELL'

OR p\_type\_name = 'CELLLUCENT'

OR p\_type\_name = 'CELLMOTOROLA'

OR p\_type\_name = 'CELLNORTEL'

OR p\_type\_name = 'CELLOTHER'

OR p\_type\_name = 'EVDO'

OR p\_type\_name = 'EVDOLUCENT'

OR p\_type\_name = 'EVDOMOTOROLA'

OR p\_type\_name = 'EVDONORTEL'

OR p\_type\_name = 'EVDOOTHER'

OR p\_type\_name = 'IPBH'

OR p\_type\_name = 'IPBHLUCENT'

OR p\_type\_name = 'IPBHMOTOROLA'

OR p\_type\_name = 'IPBHNORTEL'

OR p\_type\_name = 'IPBHOTHER'

OR p\_type\_name = '3G'

OR p\_type\_name = '3GLUCENT'

OR p\_type\_name = '3GMOTOROLA'

OR p\_type\_name = '3GNORTEL'

OR p\_type\_name = '3GOTHER'))

THEN -- only classes name with above type/category extends from SubcategoryValidator.java, others extends from PatternValidator =>PathValidator.getSubcategory(id) return null

RETURN NULL;

END IF;

--SubcategoryValidator.getSubcategory(id,category)

IF (NOT REGEXP\_LIKE (p\_CIRC\_PATH\_HUM\_ID, '^(\w+ )?(\w+)-(.\*)$')) --not exactly match, not just contains

THEN -- p\_CIRC\_PATH\_HUM\_ID not match general pattern

IF (p\_type\_name = 'EVDO')

THEN

v\_subcategory := 'EVDOOTHER';

ELSE

v\_subcategory := NULL;

END IF;

RETURN v\_subcategory;

END IF;

SELECT REGEXP\_REPLACE (p\_CIRC\_PATH\_HUM\_ID, '^(\w+ )?(\w+)-(.\*)$', '\2')

INTO v\_switchName

FROM DUAL;

BEGIN

SELECT DISTINCT B.VENDOR

INTO v\_vendor

FROM XNG\_REPORTS.XNGSTD\_SWITCH\_VENDORS B

WHERE B.SWITCH\_ID = v\_switchName;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_vendor := NULL;

END;

IF (v\_vendor IS NOT NULL)

THEN

v\_subcategory := p\_type\_name || v\_vendor;

ELSE

IF (p\_type\_name = 'EVDO')

THEN

IF (LENGTH (v\_switchName) > 6)

THEN

v\_switchName := SUBSTR (v\_switchName, 1, 6);

BEGIN

SELECT DISTINCT B.VENDOR

INTO v\_vendor

FROM XNG\_REPORTS.XNGSTD\_SWITCH\_VENDORS B

WHERE B.SWITCH\_ID = v\_switchName;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_vendor := NULL;

END;

END IF;

IF (v\_vendor IS NOT NULL)

THEN

v\_subcategory := p\_type\_name || v\_vendor;

ELSE

v\_subcategory := 'EVDOOTHER';

END IF;

ELSE --non 'EVDO'

v\_subcategory := 'BADSWITCH';

END IF;

END IF;

RETURN v\_subcategory;

END;

FUNCTION GET\_PATH\_PATTERN (p\_Category VARCHAR2)

RETURN VARCHAR2

IS

v\_pattern VARCHAR2 (100) := NULL;

BEGIN

--refer to PathValidator.init(String bundleName, String category)

IF ( (p\_Category IS NOT NULL))

THEN

BEGIN

SELECT A.VALUE

INTO v\_pattern

FROM XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES A

WHERE A.NAME =

'path.types.'

|| REGEXP\_REPLACE (p\_Category, '[ ]', '')

|| '.pattern';

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_pattern := NULL;

END;

ELSE

v\_pattern := NULL;

END IF;

IF (v\_pattern IS NOT NULL)

THEN

SELECT REPLACE ('(\w+ )\*' || v\_pattern || '(( )\*\[.\*\])?',

'\\',

'\')

INTO v\_pattern

FROM DUAL;

END IF;

RETURN v\_pattern;

END;

FUNCTION CHECK\_ID (p\_id VARCHAR2, p\_pattern VARCHAR2)

RETURN NUMBER

IS

v\_rtn NUMBER := 0;

BEGIN

IF ( ( (p\_id IS NULL) OR LENGTH (p\_id) = 0))

THEN

v\_rtn := 1;

ELSE

IF ( (p\_pattern IS NULL) OR LENGTH (p\_pattern) = 0)

THEN

v\_rtn := 0;

ELSE

IF (REGEXP\_LIKE (p\_id, '^' || p\_pattern || '$')) -- exactly match, not just contains

THEN

v\_rtn := 0; -- matched

ELSE

v\_rtn := 2; -- bad format

END IF;

END IF;

END IF;

RETURN v\_rtn;

END;

END XNGSTD\_AUDIT\_BATCH\_PKG;

/

--------------------------------------------------------

-- DDL for Package Body XNG\_CSR\_DISCOVERY

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."XNG\_CSR\_DISCOVERY"

IS

--

-- To modify this template, edit file PKGBODY.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package body

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter procedure, function bodies as shown below

PROCEDURE load\_granite\_CSR\_vlan\_paths is

sqlStmt varchar2(32000);

message varchar2(250);

methodName varchar2(30) := 'load\_granite\_CSR\_vlan\_paths';

BEGIN

message := 'Error: '|| methodName ||'(): Can''t truncate table: XNG\_REPORTS.XNG\_CSR\_VLAN\_PATHS\_WK';

sqlStmt := 'truncate table XNG\_REPORTS.XNG\_CSR\_VLAN\_PATHS\_WK' ;

execute immediate sqlStmt;

message := '';

-- this query looks at only the 1st leg of the VLAN

-- and 1st leg of the 1Gig and 10Gig path

sqlStmt:='

insert into xng\_CSR\_vlan\_paths\_wk columns ( EQUIP\_INST\_ID

, card\_inst\_id

, CIRC\_PATH\_INST\_ID, port\_BANDWIDTH, path\_TYPE

, vlan\_inst\_id, vlan\_hum\_id, vlan\_number

, vlan\_status, extract\_date

)

with

csr\_gige\_paths as (

select distinct XNE.EQUIP\_INST\_ID, p.CARD\_INST\_ID --, p.port\_inst\_id, pli.leg\_inst\_id,(pli.rel\_order), PLI.LEG\_NAME

,cpi.circ\_path\_inst\_id gige\_inst\_id, cpi.bandwidth, cpi.type

from xng\_reports.XNG\_CSR\_PARSED\_WK xne,

vzwnet.epa p,

vzwnet.circ\_path\_inst cpi,

vzwnet.path\_leg\_inst pli,

vzwnet.path\_leg\_member plm,

vzwnet.circ\_path\_element cpe

where p.EQUIP\_INST\_ID = xne.EQUIP\_INST\_ID

and cpi.CIRC\_PATH\_INST\_ID in (p.CIRC\_PATH\_INST\_ID, p.NEXT\_PATH\_INST\_ID)

and cpi.BANDWIDTH like ''%Gbps''

and pli.CIRC\_PATH\_INST\_ID = cpi.circ\_path\_INST\_ID

and pli.leg\_inst\_id = plm.leg\_inst\_id

-- and plm.sequence = cpe.sequence

and plm.element\_inst\_id = cpe.element\_inst\_id

and CPE.PORT\_INST\_ID=p.port\_inst\_id

and pli.rel\_order = (select min(rel\_order) rel\_order from vzwnet.path\_leg\_inst li where LI.CIRC\_PATH\_INST\_ID=cpi.circ\_path\_inst\_id)

),

gige\_vlan\_paths as (

select distinct gige.circ\_path\_inst\_id gige\_inst\_id , VLAN.CIRC\_PATH\_INST\_ID vlan\_inst\_id

,vlan.CIRC\_PATH\_HUM\_ID vlan\_hum\_id, regexp\_substr(regexp\_substr(vlan.CIRC\_PATH\_HUM\_ID, ''VLAN(-|[[:space:]]|)\*[[:digit:]]+''), ''[[:digit:]]+'') vlan\_number

, vlan.status

from vzwnet.circ\_path\_inst gige,

vzwnet.circ\_path\_inst vlan,

vzwnet.path\_leg\_inst pli,

vzwnet.path\_leg\_member plm,

vzwnet.circ\_path\_element cpe

where gige.BANDWIDTH like ''%Gbps''

and vlan.BANDWIDTH = ''VLAN'' and vlan.TYPE = ''EBH''

and VLAN.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and GIGE.CIRC\_PATH\_INST\_ID = CPE.PATH\_INST\_ID

and VLAN.CIRC\_PATH\_INST\_ID = PLI.CIRC\_PATH\_INST\_ID

and pli.leg\_inst\_id = plm.leg\_inst\_id

-- and plm.sequence = cpe.sequence

and plm.element\_inst\_id = cpe.element\_inst\_id

and pli.rel\_order=(select min(rel\_order) from vzwnet.path\_leg\_inst pl where PL.CIRC\_PATH\_INST\_ID = vlan.CIRC\_PATH\_INST\_ID)

)select ngp.EQUIP\_INST\_ID, ngp.CARD\_INST\_ID

,ngp.gige\_inst\_id circ\_path\_inst\_id, ngp.bandwidth, ngp.type

, gvp.vlan\_inst\_id, gvp.vlan\_hum\_id, gvp.vlan\_number, gvp.status vlan\_status, sysdate

from csr\_gige\_paths ngp, gige\_vlan\_paths gvp

where ngp.gige\_inst\_id = gvp.gige\_inst\_id

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr(message||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END load\_granite\_csr\_vlan\_paths;

END;

/

--------------------------------------------------------

-- DDL for Package Body XNG\_NGMLS\_DISCOVERY

--------------------------------------------------------

CREATE OR REPLACE PACKAGE BODY "XNG\_REPORTS"."XNG\_NGMLS\_DISCOVERY"

IS

--

-- To modify this template, edit file PKGBODY.TXT in TEMPLATE

-- directory of SQL Navigator

--

-- Purpose: Briefly explain the functionality of the package body

--

-- MODIFICATION HISTORY

-- Person Date Comments

-- --------- ------ ------------------------------------------

-- Enter procedure, function bodies as shown below

PROCEDURE load\_granite\_ngMLS\_equip is

sqlStmt varchar2(32000);

message varchar2(250);

methodName varchar2(30) := 'load\_granite\_ngMLS\_equip';

BEGIN

message := 'Error: '|| methodName ||'(): Can''t truncate table: XNG\_REPORTS.XNG\_NGMLS\_EQUIP\_WK';

sqlStmt := 'truncate table XNG\_REPORTS.XNG\_NGMLS\_EQUIP\_WK' ;

execute immediate sqlStmt;

message := 'Error: '|| methodName ||'(): Can''t insert rows into table: XNG\_REPORTS.XNG\_NGMLS\_EQUIP\_WK';

sqlStmt :=

'INSERT INTO XNG\_REPORTS.XNG\_NGMLS\_EQUIP\_WK (NE\_INST\_ID,EQUIP\_INST\_ID,SITE\_INST\_ID,PARENT\_EQ\_INST\_ID,EQ\_CLASS\_TYPE,DESCR,

MODEL,TYPE,STATUS,VENDOR,EQ\_CLASS,ngmls\_vendor)

SELECT TO\_NUMBER(TRIM ( BOTH ''/'' FROM SYS\_CONNECT\_BY\_PATH (

DECODE (LEVEL,1, ei.equip\_inst\_id,''''),''/'' )

)) ne\_inst\_id,

ei.equip\_inst\_id, ei.site\_inst\_id, ei.parent\_eq\_inst\_id,

--level lev,

DECODE (LEVEL,

1, ''NE'',

DECODE (ei.eq\_class, ''S'', ''SHELF'', ''N/A''))

eq\_class\_type,ei.descr,ei.model,ei.TYPE,ei.status,ei.vendor, ei.eq\_class,

CASE

WHEN vendor = ''CISCO'' THEN ''CI''

WHEN vendor LIKE ''%LUCENT%'' THEN ''AL''

WHEN vendor LIKE ''%ALCATEL%'' THEN ''AL''

ELSE NULL

END ngmls\_vendor

FROM vzwnet.equip\_inst ei

WHERE ei.eq\_class = ''S'' -- LR 10/22(LEVEL = 1 OR ei.eq\_class = ''S'')

START WITH ei.TYPE = ''NGMLS''

CONNECT BY PRIOR ei.equip\_inst\_id = ei.parent\_eq\_inst\_id ';

execute immediate sqlStmt;

message := 'Error: '|| methodName ||'(): Can''t delete rows from table: XNG\_REPORTS.XNG\_NGMLS\_EQUIP\_WK';

sqlStmt :=

' delete from XNG\_REPORTS.XNG\_NGMLS\_EQUIP\_WK where rowid in

( select rowid from ( select rowid, row\_number() over

(partition by equip\_inst\_id, eq\_class\_type order by equip\_inst\_id, eq\_class\_type) dup

from XNG\_REPORTS.XNG\_NGMLS\_EQUIP\_WK ) where dup > 1) ';

execute immediate sqlStmt;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr(message||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END;

PROCEDURE load\_granite\_ngMLS\_vlan\_paths is

sqlStmt varchar2(32000);

message varchar2(250);

methodName varchar2(30) := 'load\_granite\_ngmls\_vlan\_paths';

BEGIN

message := 'Error: '|| methodName ||'(): Can''t truncate table: XNG\_REPORTS.XNG\_NGMLS\_VLAN\_PATHS\_WK';

sqlStmt := 'truncate table XNG\_REPORTS.XNG\_NGMLS\_VLAN\_PATHS\_WK' ;

execute immediate sqlStmt;

message := '';

-- this query looks at only the 1st leg of the VLAN

-- and 1st leg of the 1Gig and 10Gig path

sqlStmt:='

insert into xng\_ngMLS\_vlan\_paths\_wk columns (ne\_inst\_id, EQUIP\_INST\_ID

, card\_inst\_id

, CIRC\_PATH\_INST\_ID, port\_BANDWIDTH, path\_TYPE

, vlan\_inst\_id, vlan\_hum\_id, vlan\_number

, vlan\_status, extract\_date

)

with

ngmls\_gige\_paths as (

select distinct ne\_inst\_id, XNE.EQUIP\_INST\_ID, p.CARD\_INST\_ID --, p.port\_inst\_id, pli.leg\_inst\_id,(pli.rel\_order), PLI.LEG\_NAME

,cpi.circ\_path\_inst\_id gige\_inst\_id, cpi.bandwidth, cpi.type

from xng\_reports.xng\_ngMLS\_equip\_wk xne,

vzwnet.epa p,

vzwnet.circ\_path\_inst cpi,

vzwnet.path\_leg\_inst pli,

vzwnet.path\_leg\_member plm,

vzwnet.circ\_path\_element cpe

where p.EQUIP\_INST\_ID = xne.EQUIP\_INST\_ID

and cpi.CIRC\_PATH\_INST\_ID in (p.CIRC\_PATH\_INST\_ID, p.NEXT\_PATH\_INST\_ID)

and cpi.BANDWIDTH like ''%Gbps''

and pli.CIRC\_PATH\_INST\_ID = cpi.circ\_path\_INST\_ID

and pli.leg\_inst\_id = plm.leg\_inst\_id

-- and plm.sequence = cpe.sequence

and plm.element\_inst\_id = cpe.element\_inst\_id

and CPE.PORT\_INST\_ID=p.port\_inst\_id

and pli.rel\_order = (select min(rel\_order) rel\_order from vzwnet.path\_leg\_inst li where LI.CIRC\_PATH\_INST\_ID=cpi.circ\_path\_inst\_id)

),

gige\_vlan\_paths as (

select distinct gige.circ\_path\_inst\_id gige\_inst\_id , VLAN.CIRC\_PATH\_INST\_ID vlan\_inst\_id

,vlan.CIRC\_PATH\_HUM\_ID vlan\_hum\_id, regexp\_substr(regexp\_substr(vlan.CIRC\_PATH\_HUM\_ID, ''VLAN(-|[[:space:]]|)\*[[:digit:]]+''), ''[[:digit:]]+'') vlan\_number

, vlan.status

from vzwnet.circ\_path\_inst gige,

vzwnet.circ\_path\_inst vlan,

vzwnet.path\_leg\_inst pli,

vzwnet.path\_leg\_member plm,

vzwnet.circ\_path\_element cpe

where gige.BANDWIDTH like ''%Gbps''

and vlan.BANDWIDTH = ''VLAN'' and vlan.TYPE = ''EBH''

and VLAN.CIRC\_PATH\_INST\_ID = CPE.CIRC\_PATH\_INST\_ID

and GIGE.CIRC\_PATH\_INST\_ID = CPE.PATH\_INST\_ID

and VLAN.CIRC\_PATH\_INST\_ID = PLI.CIRC\_PATH\_INST\_ID

and pli.leg\_inst\_id = plm.leg\_inst\_id

--and plm.sequence = cpe.sequence

and plm.element\_inst\_id = cpe.element\_inst\_id

and pli.rel\_order=(select min(rel\_order) from vzwnet.path\_leg\_inst pl where PL.CIRC\_PATH\_INST\_ID = vlan.CIRC\_PATH\_INST\_ID)

)select ngp.ne\_inst\_id, ngp.EQUIP\_INST\_ID, ngp.CARD\_INST\_ID

,ngp.gige\_inst\_id circ\_path\_inst\_id, ngp.bandwidth, ngp.type

, gvp.vlan\_inst\_id, gvp.vlan\_hum\_id, gvp.vlan\_number, gvp.status vlan\_status, sysdate

from ngmls\_gige\_paths ngp, gige\_vlan\_paths gvp

where ngp.gige\_inst\_id = gvp.gige\_inst\_id

';

execute immediate sqlStmt;

commit;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line(SubStr(message||TO\_CHAR(SQLCODE)||': '||SQLERRM, 1, 255));

RAISE;

END load\_granite\_ngMLS\_vlan\_paths;

END;

/

--------------------------------------------------------

-- DDL for Function AR\_GET\_PORT\_CONNECTOR

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."AR\_GET\_PORT\_CONNECTOR" ( MODULE\_MODEL\_IN IN VARCHAR2 )

RETURN VARCHAR2 DETERMINISTIC

IS

BEGIN

IF REGEXP\_SUBSTR(UPPER(MODULE\_MODEL\_IN),'^(SERIAL)') = 'SERIAL'

THEN RETURN 'Logical';

END IF;

CASE UPPER(MODULE\_MODEL\_IN)

WHEN 'CFP-100G-LR4'

THEN RETURN 'CFP';

WHEN 'GLC-LH-SM'

THEN RETURN 'SFP';

WHEN 'GLC-LH-SMD'

THEN RETURN 'SFP';

WHEN 'ONS-XC-10G-S1'

THEN RETURN 'XFP';

WHEN 'SFP-GE-L'

THEN RETURN 'SFP';

WHEN 'SFP-GE-S'

THEN RETURN 'SFP';

WHEN 'SFP-OC12-IR1'

THEN RETURN 'SFP';

WHEN 'SFP-OC12-LR1'

THEN RETURN 'SFP';

WHEN 'SFP-OC3-IR1'

THEN RETURN 'SFP';

WHEN 'SFP-OC3-LR1'

THEN RETURN 'SFP';

WHEN 'SFP-OC3-MM'

THEN RETURN 'SFP';

WHEN 'XFP-10GER-OC192IR'

THEN RETURN 'XFP';

WHEN 'XFP10GLR-192SR-L'

THEN RETURN 'XFP';

WHEN 'XFP-10GLR-OC192SR'

THEN RETURN 'XFP';

WHEN 'XFP-10GZR-OC192LR'

THEN RETURN 'XFP';

WHEN 'SPA-3XOC3-ATM-V2'

THEN RETURN 'SFP';

ELSE

RETURN 'SFP';

END CASE;

END;

/

--------------------------------------------------------

-- DDL for Function CHECKID

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."CHECKID" (p\_id VARCHAR2,

p\_pattern VARCHAR2)

RETURN NUMBER

IS

v\_rtn NUMBER := 0;

BEGIN

IF ( ((p\_id IS NULL) OR LENGTH (p\_id) = 0))

THEN

v\_rtn := 1;

ELSE

IF ((p\_pattern IS NULL) OR LENGTH (p\_pattern) = 0)

THEN

v\_rtn := 0;

ELSE

IF (REGEXP\_LIKE (p\_id, '^'||p\_pattern||'$')) -- exactly match, not just contains

THEN

v\_rtn := 0; -- matched

ELSE

v\_rtn := 2; -- bad format

END IF;

END IF;

END IF;

RETURN v\_rtn;

END;

/

--------------------------------------------------------

-- DDL for Function CLOBAGG

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."CLOBAGG" (

input clob

)

return clob

deterministic

parallel\_enable

aggregate using clobagg\_type;

/

--------------------------------------------------------

-- DDL for Function GETPATHPATTERN

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."GETPATHPATTERN" (

p\_Category VARCHAR2)

RETURN VARCHAR2

IS

v\_pattern VARCHAR2 (100) := NULL;

BEGIN

--refer to PathValidator.init(String bundleName, String category)

IF ( (p\_Category IS NOT NULL))

THEN

BEGIN

SELECT A.VALUE

INTO v\_pattern

FROM XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES A

WHERE A.NAME = 'path.types.' || REGEXP\_REPLACE (p\_Category, '[ ]', '') || '.pattern';

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_pattern := NULL;

END;

ELSE

v\_pattern := NULL;

END IF;

IF (v\_pattern IS NOT NULL)

THEN

SELECT REPLACE ('(\w+ )\*' || v\_pattern || '(( )\*\[.\*\])?', '\\', '\')

INTO v\_pattern

FROM DUAL;

END IF;

RETURN v\_pattern;

END;

/

--------------------------------------------------------

-- DDL for Function GETSUBCATEGORY

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."GETSUBCATEGORY" (

p\_CIRC\_PATH\_HUM\_ID VARCHAR2,

p\_DOM\_IP VARCHAR2,

p\_type\_name VARCHAR2)

RETURN VARCHAR2

IS

v\_switchName VARCHAR2 (100) := NULL;

v\_vendor VARCHAR2 (100) := NULL;

v\_subcategory VARCHAR2 (100) := NULL;

v\_pattern VARCHAR2 (100) := NULL;

v\_pathTypes VARCHAR2 (2000) := NULL;

v\_typeName VARCHAR2 (100) := NULL;

BEGIN

v\_subcategory := NULL;

IF (p\_CIRC\_PATH\_HUM\_ID IS NULL)

THEN

RETURN v\_subcategory;

END IF;

BEGIN

SELECT REGEXP\_REPLACE (P.VALUE, '[ ]', '')

INTO v\_pathTypes

FROM XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES P

WHERE P.NAME = 'path.types';

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_pathTypes := '';

END;

v\_typeName := REGEXP\_REPLACE (p\_type\_name, '[ ]', '');

IF ( v\_pathTypes = ''

OR NOT REGEXP\_LIKE (

',' || v\_pathTypes || ',',

',' || v\_typeName || ',')) -- path.types not contains p\_type\_name

THEN

RETURN v\_subcategory;

END IF;

BEGIN

SELECT P.VALUE

INTO v\_pattern

FROM XNG\_REPORTS.XNGSTD\_AUDIT\_PROPERTIES P

WHERE P.NAME = 'path.types.'||v\_typeName||'.pattern';

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_pattern := '';

END;

IF (v\_pattern = '') -- path.types.typename.pattern not existin in property

THEN

RETURN v\_subcategory;

END IF;

IF ( (p\_DOM\_IP IS NOT NULL AND p\_DOM\_IP NOT LIKE '%not present%')

AND (p\_type\_name = 'EVDO' OR p\_type\_name = '3G'))

THEN -- XngPathAudit.getResults()

RETURN p\_type\_name || 'NORTEL'; -- EVDO/X3G.getSubcategoryForNortel()->SubcategoryValidator.getSubcategory(id,category,dom\_ip\_node)

END IF;

IF ( NOT (p\_type\_name = 'AMPS'

OR p\_type\_name = 'CDMA' OR p\_type\_name = 'CDMALUCENT' OR p\_type\_name = 'CDMAMOTOROLA' OR p\_type\_name = 'CDMANORTEL' OR p\_type\_name = 'CDMAOTHER'

OR p\_type\_name = 'CELL' OR p\_type\_name = 'CELLLUCENT' OR p\_type\_name = 'CELLMOTOROLA' OR p\_type\_name = 'CELLNORTEL' OR p\_type\_name = 'CELLOTHER'

OR p\_type\_name = 'EVDO' OR p\_type\_name = 'EVDOLUCENT' OR p\_type\_name = 'EVDOMOTOROLA' OR p\_type\_name = 'EVDONORTEL' OR p\_type\_name = 'EVDOOTHER'

OR p\_type\_name = 'IPBH' OR p\_type\_name = 'IPBHLUCENT' OR p\_type\_name = 'IPBHMOTOROLA' OR p\_type\_name = 'IPBHNORTEL' OR p\_type\_name = 'IPBHOTHER'

OR p\_type\_name = '3G' OR p\_type\_name = '3GLUCENT' OR p\_type\_name = '3GMOTOROLA' OR p\_type\_name = '3GNORTEL' OR p\_type\_name = '3GOTHER'))

THEN -- only classes name with above type/category extends from SubcategoryValidator.java, others extends from PatternValidator =>PathValidator.getSubcategory(id) return null

RETURN NULL;

END IF;

--SubcategoryValidator.getSubcategory(id,category)

IF (NOT REGEXP\_LIKE (p\_CIRC\_PATH\_HUM\_ID, '^(\w+ )?(\w+)-(.\*)$')) --not exactly match, not just contains

THEN -- p\_CIRC\_PATH\_HUM\_ID not match general pattern

IF (p\_type\_name = 'EVDO')

THEN

v\_subcategory := 'EVDOOTHER';

ELSE

v\_subcategory := NULL;

END IF;

RETURN v\_subcategory;

END IF;

SELECT REGEXP\_REPLACE (p\_CIRC\_PATH\_HUM\_ID, '^(\w+ )?(\w+)-(.\*)$', '\2')

INTO v\_switchName

FROM DUAL;

BEGIN

SELECT DISTINCT B.VENDOR

INTO v\_vendor

FROM XNG\_REPORTS.XNGSTD\_SWITCH\_VENDORS B

WHERE B.SWITCH\_ID = v\_switchName;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_vendor := NULL;

END;

IF (v\_vendor IS NOT NULL)

THEN

v\_subcategory := p\_type\_name || v\_vendor;

ELSE

IF (p\_type\_name = 'EVDO')

THEN

IF (LENGTH (v\_switchName) > 6)

THEN

v\_switchName := SUBSTR (v\_switchName, 1, 6);

BEGIN

SELECT DISTINCT B.VENDOR

INTO v\_vendor

FROM XNG\_REPORTS.XNGSTD\_SWITCH\_VENDORS B

WHERE B.SWITCH\_ID = v\_switchName;

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN

v\_vendor := NULL;

END;

END IF;

IF (v\_vendor IS NOT NULL)

THEN

v\_subcategory := p\_type\_name || v\_vendor;

ELSE

v\_subcategory := 'EVDOOTHER';

END IF;

ELSE --non 'EVDO'

v\_subcategory := 'BADSWITCH';

END IF;

END IF;

RETURN v\_subcategory;

END;

/

--------------------------------------------------------

-- DDL for Function GET\_CIRC\_PATH\_ELEMENTS\_BY\_LEG

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."GET\_CIRC\_PATH\_ELEMENTS\_BY\_LEG" (p\_leg\_inst\_id in vzwnet.circuit\_path\_elements.LEG\_INST\_ID%TYPE,

p\_member\_nbr in vzwnet.circuit\_path\_elements.MEMBER\_NBR%TYPE )

RETURN VARCHAR2

IS

l\_text VARCHAR2(4000) := ' ';

BEGIN

dbms\_output.PUT\_LINE (' PathLegId=' || p\_leg\_inst\_id || '; MemberNbr=' || p\_member\_nbr );

if (p\_member\_nbr > 0 ) then

FOR cur\_rec IN (

SELECT cpe.port as path\_element, cpe.path\_elem\_inst\_id, lm.sequence, cpe.member\_nbr

FROM vzwnet.circuit\_path\_elements cpe join vzwnet.path\_leg\_member lm on cpe.LEG\_INST\_ID = lm.LEG\_INST\_ID and cpe.PATH\_ELEM\_INST\_ID = lm.ELEMENT\_INST\_ID

WHERE cpe.leg\_inst\_id = p\_leg\_inst\_id and cpe.member\_nbr = p\_member\_nbr and cpe.element\_type != 'S' and cpe.element\_type != 'K'

union

SELECT cpe.element\_name as path\_element, cpe.path\_elem\_inst\_id, lm.sequence, cpe.member\_nbr

FROM vzwnet.circuit\_path\_elements cpe join vzwnet.path\_leg\_member lm on cpe.LEG\_INST\_ID = lm.LEG\_INST\_ID and cpe.PATH\_ELEM\_INST\_ID = lm.ELEMENT\_INST\_ID

WHERE cpe.leg\_inst\_id = p\_leg\_inst\_id and cpe.member\_nbr = p\_member\_nbr and (cpe.element\_type = 'S' or cpe.element\_type = 'K')

order by sequence, member\_nbr

) LOOP

l\_text := l\_text || ',' || trim(cur\_rec.path\_element);

END LOOP;

else

FOR cur\_rec IN (

SELECT cpe.port as path\_element, cpe.path\_elem\_inst\_id, lm.sequence

FROM vzwnet.circuit\_path\_elements cpe join vzwnet.path\_leg\_member lm on cpe.LEG\_INST\_ID = lm.LEG\_INST\_ID and cpe.PATH\_ELEM\_INST\_ID = lm.ELEMENT\_INST\_ID

WHERE cpe.leg\_inst\_id = p\_leg\_inst\_id and cpe.element\_type != 'S' and cpe.element\_type != 'K'

union

SELECT cpe.element\_name as path\_element, cpe.path\_elem\_inst\_id, lm.sequence

FROM vzwnet.circuit\_path\_elements cpe join vzwnet.path\_leg\_member lm on cpe.LEG\_INST\_ID = lm.LEG\_INST\_ID and cpe.PATH\_ELEM\_INST\_ID = lm.ELEMENT\_INST\_ID

WHERE cpe.leg\_inst\_id = p\_leg\_inst\_id and (cpe.element\_type = 'S' or cpe.element\_type = 'K')

order by sequence

) LOOP

l\_text := l\_text || ',' || trim(cur\_rec.path\_element);

END LOOP;

end if;

--dbms\_output.put\_line(l\_text);

RETURN ltrim(l\_text, ',');

EXCEPTION WHEN VALUE\_ERROR then

dbms\_output.PUT\_LINE (' PathLegId=' || p\_leg\_inst\_id || '; MemberNbr=' || p\_member\_nbr || ';size=' || length(l\_text) || '; text=' || l\_text );

RETURN ltrim(l\_text, ',');

END;

--create or replace view xng\_reports.MSPP\_VSM\_XNG\_MATCH as

--select nefni.\*, ei.site\_inst\_id, ei.inst\_id, ei.ne\_inst\_id, ei.descr --, nefni.node\_id, nefni.tid

--from

--( select fni.node\_id as vsm\_node\_id, ne.tid, fni.equip\_inst\_id from

-- xng\_reports.INC\_MSPP\_NE ne

-- FULL OUTER JOIN xng\_reports.MSPP\_FM\_NODE\_ID\_V fni ON UPPER (SUBSTR (ne.tid, 1, 8)) = UPPER (SUBSTR (fni.NODE\_ID, 1, 8))

-- AND UPPER (SUBSTR (ne.tid, 9, 2)) = 'MS'

-- AND SUBSTR (ne.tid, 11, 1) = SUBSTR (fni.NODE\_ID, 12, 1) ) nefni

-- FULL OUTER JOIN xng\_reports.MSPP\_DEVICES\_V ei ON ei.NE\_INST\_ID = nefni.EQUIP\_INST\_ID

--;

/\* Formatted on 2010/10/13 14:40 (Formatter Plus v4.8.8) \*/

/

--------------------------------------------------------

-- DDL for Function GET\_MSPP\_PATH\_ELEMENTS\_BY\_LEG

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."GET\_MSPP\_PATH\_ELEMENTS\_BY\_LEG" (p\_leg\_inst\_id in vzwnet.circuit\_path\_elements.LEG\_INST\_ID%TYPE, p\_member\_nbr in vzwnet.circuit\_path\_elements.MEMBER\_NBR%TYPE,

p\_equip\_inst\_id in vzwnet.equip\_inst.EQUIP\_INST\_ID%TYPE default 0)

RETURN VARCHAR2

IS

l\_text VARCHAR2(4000) := ' ';

BEGIN

dbms\_output.PUT\_LINE (' PathLegId=' || p\_leg\_inst\_id || '; MemberNbr=' || p\_member\_nbr );

if (p\_member\_nbr > 0 ) then

FOR cur\_rec IN (

SELECT cpe.port as path\_element, cpe.path\_elem\_inst\_id, lm.sequence, cpe.member\_nbr, cpe.element\_category, cpe.equip\_inst\_id

FROM vzwnet.circuit\_path\_elements cpe join vzwnet.path\_leg\_member lm

on cpe.LEG\_INST\_ID = lm.LEG\_INST\_ID and cpe.PATH\_ELEM\_INST\_ID = lm.ELEMENT\_INST\_ID

WHERE cpe.leg\_inst\_id = p\_leg\_inst\_id and cpe.member\_nbr = p\_member\_nbr

order by lm.sequence, cpe.member\_nbr

) LOOP

if ( (cur\_rec.element\_category = 'MSPP' and not(p\_equip\_inst\_id > 0)) or (cur\_rec.element\_category = 'MSPP' and p\_equip\_inst\_id > 0 and cur\_rec.equip\_inst\_id = p\_equip\_inst\_id ) ) then

l\_text := l\_text || ',' || trim(leading 0 from trim(cur\_rec.path\_element));

else

l\_text := l\_text || ',';

end if;

END LOOP;

else

FOR cur\_rec IN (

SELECT cpe.port as path\_element, cpe.path\_elem\_inst\_id, lm.sequence, cpe.element\_category, cpe.equip\_inst\_id

FROM vzwnet.circuit\_path\_elements cpe join vzwnet.path\_leg\_member lm

on cpe.LEG\_INST\_ID = lm.LEG\_INST\_ID and cpe.PATH\_ELEM\_INST\_ID = lm.ELEMENT\_INST\_ID

WHERE cpe.leg\_inst\_id = p\_leg\_inst\_id

order by lm.sequence

) LOOP

if ( (cur\_rec.element\_category = 'MSPP' and not(p\_equip\_inst\_id > 0)) or (cur\_rec.element\_category = 'MSPP' and p\_equip\_inst\_id > 0 and cur\_rec.equip\_inst\_id = p\_equip\_inst\_id ) ) then

l\_text := l\_text || ',' || trim(leading 0 from trim(cur\_rec.path\_element));

else

l\_text := l\_text || ',';

end if;

END LOOP;

end if;

RETURN ltrim(l\_text, ',');

--dbms\_output.put\_line(l\_text);

EXCEPTION WHEN VALUE\_ERROR then

dbms\_output.PUT\_LINE (' PathLegId=' || p\_leg\_inst\_id || '; MemberNbr=' || p\_member\_nbr || ';size=' || length(l\_text) || '; text=' || l\_text );

RETURN ltrim(l\_text, ',');

END;

/

--------------------------------------------------------

-- DDL for Function ISVALIDPHONENUM

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."ISVALIDPHONENUM" (

p\_phoneNumber VARCHAR2)

RETURN NUMBER

IS

v\_phone VARCHAR2 (100) := '';

v\_phone2 VARCHAR2 (100) := '';

v\_length NUMBER := 0;

v\_length2 NUMBER := 0;

v\_return NUMBER := 0;

BEGIN

IF (p\_phoneNumber IS NULL)

THEN

v\_return := 1;

ELSE

SELECT LENGTH (p\_phoneNumber) INTO v\_length FROM DUAL;

IF (v\_length < 10)

THEN

v\_return := 1;

ELSE

SELECT REGEXP\_REPLACE (p\_phoneNumber, '[ |\(|\)|\-]', '')

INTO v\_phone

FROM DUAL;

SELECT LENGTH (v\_phone) INTO v\_length FROM DUAL;

IF (v\_length < 10)

THEN

v\_return := 1;

ELSE

-- case all digit are repeating

SELECT REGEXP\_REPLACE (v\_phone, '^([0-9])\1\*$', '')

INTO v\_phone2

FROM DUAL;

SELECT LENGTH (v\_phone2) INTO v\_length2 FROM DUAL;

IF (v\_length2 > 0)

THEN

-- exact 9 digit starting non 0 or 1

SELECT REGEXP\_REPLACE (v\_phone, '[2-9]\d{9}', '')

INTO v\_phone2

FROM DUAL;

SELECT LENGTH (v\_phone2) INTO v\_length2 FROM DUAL;

IF (v\_length2 > 0)

THEN

v\_return := 1;

END IF;

ELSE

v\_return := 1; --repeating

END IF;

END IF;

END IF;

END IF;

RETURN v\_return;

END;

/

--------------------------------------------------------

-- DDL for Function VZ\_GETNAME

--------------------------------------------------------

CREATE OR REPLACE FUNCTION "XNG\_REPORTS"."VZ\_GETNAME" (p\_userID varchar2) RETURN varchar2 IS

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: VZ\_GETNAME

PURPOSE: gets the full name based on userid

REVISIONS:

Ver Date Author Description

--------- ---------- --------------- ------------------------------------

1.0 08/28/01 Anon 1. Created this function.

PARAMETERS:

INPUT: p\_userID could be NULL

OUTPUT:

RETURNED VALUE: NUMBER

CALLED BY:

CALLS:

EXAMPLE USE: vz\_getname(nvl(p\_user\_id, '.'')

ASSUMPTIONS:

LIMITATIONS:

ALGORITHM:

NOTES:

move this to a package

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

vs\_fullName varchar2(50);

BEGIN

select nvl(FULL\_NAME, p\_userId)

into vs\_fullName

from xperadmin.user\_inst

where user\_name = rtrim(ltrim(p\_userID));

RETURN vs\_fullName;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

vs\_fullName := p\_userId;

RETURN vs\_fullName;

WHEN OTHERS THEN

vs\_fullName := p\_userId;

RETURN vs\_fullName;

END VZ\_GETNAME;

/

--------------------------------------------------------

-- DDL for Synonymn BTP\_XNG\_AUDIT\_HIST\_EXP

--------------------------------------------------------

CREATE OR REPLACE SYNONYM "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_HIST\_EXP" FOR "XNG\_REPORTS"."BTP\_XNG\_AUDIT\_HIST\_PP\_V";

--------------------------------------------------------

-- DDL for Synonymn NIS\_EXTRACTOR

--------------------------------------------------------

CREATE OR REPLACE SYNONYM "XNG\_REPORTS"."NIS\_EXTRACTOR" FOR "XNG\_APPS"."NIS\_EXTRACTOR";

--------------------------------------------------------

-- DDL for Synonymn NIS\_EXTRACTOR\_1SQ

--------------------------------------------------------

CREATE OR REPLACE SYNONYM "XNG\_REPORTS"."NIS\_EXTRACTOR\_1SQ" FOR "XNG\_APPS"."NIS\_EXTRACTOR\_1SQ";

--------------------------------------------------------

-- DDL for Synonymn NIS\_EXTRACTOR\_CONTROL

--------------------------------------------------------

CREATE OR REPLACE SYNONYM "XNG\_REPORTS"."NIS\_EXTRACTOR\_CONTROL" FOR "XNG\_APPS"."NIS\_EXTRACTOR\_CONTROL";

--------------------------------------------------------

-- DDL for Synonymn NIS\_EXTRACTOR\_CONTROL\_1SQ

--------------------------------------------------------

CREATE OR REPLACE SYNONYM "XNG\_REPORTS"."NIS\_EXTRACTOR\_CONTROL\_1SQ" FOR "XNG\_APPS"."NIS\_EXTRACTOR\_CONTROL\_1SQ";

--------------------------------------------------------

-- DDL for Synonymn NIS\_EXTRACTOR\_GRAPH

--------------------------------------------------------

CREATE OR REPLACE SYNONYM "XNG\_REPORTS"."NIS\_EXTRACTOR\_GRAPH" FOR "XNG\_APPS"."NIS\_EXTRACTOR\_GRAPH";

--------------------------------------------------------

-- DDL for Synonymn NIS\_EXTRACTOR\_PARAM

--------------------------------------------------------

CREATE OR REPLACE SYNONYM "XNG\_REPORTS"."NIS\_EXTRACTOR\_PARAM" FOR "XNG\_APPS"."NIS\_EXTRACTOR\_PARAM";

--------------------------------------------------------

-- Constraints for Table NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("CABLE\_ALL\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("CABLE\_NON\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("CABLE\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("PATH\_ALL\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("PATH\_NON\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("PATH\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("EQUIP\_ALL\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("EQUIP\_NON\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("EQUIP\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("SEG\_ALL\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("SEG\_NON\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM" MODIFY ("SEG\_EBH\_COUNT" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DF\_STATUS\_DETAILS\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_WK" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_WK" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_WK" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DISCONNECT\_REASON\_CODE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DISCONNECT\_AGING

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING" MODIFY ("RUN\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING" MODIFY ("STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table WO\_BY\_QUEUE\_TEMP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_TEMP" MODIFY ("TASK\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_TEMP" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_TEMP" MODIFY ("STATUS\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_TEMP" MODIFY ("WO\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LEASED\_SEG\_DF\_DETAILS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("DF\_SEG\_BW" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("DF\_SEG\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("NDF\_SEG\_BW" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("NDF\_SEG\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("NDF\_SEG\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("DF\_SITE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LEASED\_SEG\_DF\_NDF\_PATH\_UDAS\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_NDF\_PATH\_UDAS\_WK" MODIFY ("TRANSPORT\_ACTION\_UDA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_NDF\_PATH\_UDAS\_WK" MODIFY ("NDF\_PATH\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_NDF\_PATH\_UDAS\_WK" ADD CONSTRAINT "EBH\_DISCONNECT\_NDF\_PATH\_UDASPK" PRIMARY KEY ("NDF\_PATH\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table SWITCH\_PATH\_T1\_TESTABILITY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" ADD CONSTRAINT "PK\_SWITCH\_PATH\_T1\_TESTABILITY" PRIMARY KEY ("SWITCH\_PATH\_T1\_TESTABILITY\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("MATCHING\_LIVE\_XNG\_PATHS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("IS\_ACTIVE\_IN\_XNG" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("IS\_FOUND\_IN\_XNG" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("IS\_ACTIVE\_IN\_NE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("DEVICE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("NETWORK\_ELEMENT\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("SPAN\_TECHNOLOGY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("DOMAIN\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY" MODIFY ("SWITCH\_PATH\_T1\_TESTABILITY\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_SUMMIT\_EXTRACT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_EXTRACT" ADD CONSTRAINT "XNG\_SUMMIT\_EXTRACT\_PK" PRIMARY KEY ("SITE\_INST\_ID", "SITE\_DESCRIPTION")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_EXTRACT" MODIFY ("NAURU" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_EXTRACT" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_EXTRACT" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NE\_VS\_XNG\_SP\_SUMMARY\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY\_WRK" MODIFY ("TOTAL" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY\_WRK" MODIFY ("PER\_MATCHED" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY\_WRK" MODIFY ("NO\_MATCH\_IN\_GI" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY\_WRK" MODIFY ("MATCHED\_GI" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_XCONNECT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_XCONNECT" MODIFY ("SNCNE\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NETSMART\_XCONNECT" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NETSMART\_XCONNECT" ADD CONSTRAINT "NETSMART\_XCONNECT\_PK" PRIMARY KEY ("INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table XNG\_NGMLS\_EQUIP\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK" MODIFY ("EQ\_CLASS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK" MODIFY ("STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_EQUIP\_WK" MODIFY ("NE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SITE\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SITE\_AUDIT" MODIFY ("EXTRACT\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_AUDIT" MODIFY ("POPULATED\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_AUDIT" MODIFY ("ATTR\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_AUDIT" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_AUDIT" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SITE\_INST\_GEOCODE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SITE\_INST\_GEOCODE" ADD PRIMARY KEY ("SITE\_INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table EBH\_DISCONNECT\_DETAILS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."EBH\_DISCONNECT\_DETAILS" MODIFY ("DF\_SEG\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table VZW\_NETWORK\_ORG\_OLD

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_OLD" MODIFY ("MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_OLD" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_OLD" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM" MODIFY ("CLLI\_6" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM" MODIFY ("LEAF\_DOMAIN\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM" MODIFY ("MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table WO\_BY\_QUEUE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE" MODIFY ("TASK\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE" MODIFY ("STATUS\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE" MODIFY ("WO\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table VZW\_INST\_AUDIT\_WINDOW\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW\_WRK" MODIFY ("EDIT\_OPERATION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW\_WRK" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW\_WRK" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW\_WRK" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CIRC\_INST\_AUDIT\_TRAIL\_BAD\_DATA

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CIRC\_INST\_AUDIT\_TRAIL\_BAD\_DATA" MODIFY ("INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table HIGH\_CAPACITY\_PATH\_CHAN\_PARENT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_PARENT" MODIFY ("PATHPARENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_PARENT" MODIFY ("PATHPARENT\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_PARENT" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_PARENT" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_PARENT" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_NEWS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_NEWS" ADD UNIQUE ("ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_NEWS" MODIFY ("TITLE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NEWS" MODIFY ("ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table BTP\_XNG\_UPD\_DETAILS\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_WK" MODIFY ("MATCH\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS\_WK" MODIFY ("XNG\_EXTRACT\_DATE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table WATCHDOG\_HIST

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WATCHDOG\_HIST" MODIFY ("PROCESS\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table RBED\_SEVONE\_DEVICE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DEVICE" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DEVICE" MODIFY ("NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DEVICE" MODIFY ("ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LIVE\_OBJ\_NL\_SITES\_SUMM\_WK\_PATH

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMM\_WK\_PATH" MODIFY ("PATH\_CNT" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table FILTERED\_SEGMENTS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_NE\_VS\_XNG\_AUDIT\_TEMP" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table FILTERED\_SEGMENTS\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS\_WRK" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS\_WRK" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS\_WRK" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS\_WRK" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FILTERED\_SEGMENTS\_WRK" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_NE\_VS\_XNG\_AUDIT" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XCONN\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XCONN\_AUDIT" ADD CONSTRAINT "PK\_XCONN\_AUDIT" PRIMARY KEY ("XNG\_TARGET\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XCONN\_AUDIT" MODIFY ("XNG\_EQUIP\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XCONN\_AUDIT" MODIFY ("XNG\_SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NOCC\_LAST\_TEST\_DETAILS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_DETAILS" MODIFY ("SITE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_DETAILS" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_DETAILS" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_DETAILS" MODIFY ("DOMAIN\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_ADM\_NE\_VS\_XNG\_AUDIT" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TDM\_DISCONNECT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TDM\_DISCONNECT" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_MPLS\_RTR\_PORTS\_PARSED

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_MPLS\_RTR\_PORTS\_PARSED" MODIFY ("PORT\_BW" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MPLS\_RTR\_PORTS\_PARSED" MODIFY ("PORT\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table EBH\_DISCONNECT\_DETAILS\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."EBH\_DISCONNECT\_DETAILS\_WK" MODIFY ("DF\_SEG\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NL\_OBJ\_IN\_LIVE\_PATHS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS" MODIFY ("ELEMENT\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS" MODIFY ("PATH\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS" MODIFY ("SEQUENCE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS" MODIFY ("CIRC\_PATH\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NGMLS\_ISSUE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NGMLS\_ISSUE" MODIFY ("IS\_CRITICAL" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NGMLS\_ISSUE" MODIFY ("DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NGMLS\_ISSUE" MODIFY ("NGMLS\_ISSUE\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NGMLS\_ISSUE" ADD UNIQUE ("DESCRIPTION")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."NGMLS\_ISSUE" ADD PRIMARY KEY ("NGMLS\_ISSUE\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table CIRC\_SEG\_NEW\_DECOM\_DETAIL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("CHG\_TS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("ACTION\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("SITE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("Z\_SITE\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("SEGMENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("SEGMENT\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("DOMAIN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("DOMAIN\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL" MODIFY ("RPT\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table JUNIPER\_NE\_VS\_XNG\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_NE\_VS\_XNG\_AUDIT" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NL\_OBJ\_IN\_LIVE\_PATHS\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS\_WK" MODIFY ("ELEMENT\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS\_WK" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS\_WK" MODIFY ("PATH\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS\_WK" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS\_WK" MODIFY ("SEQUENCE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJ\_IN\_LIVE\_PATHS\_WK" MODIFY ("CIRC\_PATH\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table WO\_BY\_QUEUE\_BACKUP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_BACKUP" MODIFY ("TASK\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_BACKUP" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_BACKUP" MODIFY ("STATUS\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_QUEUE\_BACKUP" MODIFY ("WO\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XOR\_REPORTS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XOR\_REPORTS" ADD CONSTRAINT "XOR\_REPORTS\_U01" UNIQUE ("REPORT\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XOR\_REPORTS" ADD CONSTRAINT "XOR\_REPORTS\_PK" PRIMARY KEY ("REPORT\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table LDAP\_XNG\_MISMATCH

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LDAP\_XNG\_MISMATCH" MODIFY ("USER\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_ENB\_PARSED

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_ENB\_PARSED" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_ENB\_PARSED" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_ENB\_PARSED" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_ENB\_PARSED" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_ENB\_PARSED" MODIFY ("SITE\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NORTEL\_DOM\_INVENTORY\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NORTEL\_DOM\_INVENTORY\_WRK" ADD PRIMARY KEY ("SEQ\_NUM")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table XNG\_VLAN\_PARSED\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_VLAN\_PARSED\_WK" MODIFY ("CIRC\_PATH\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XOR\_REPORTS\_WITH\_VH

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XOR\_REPORTS\_WITH\_VH" ADD CONSTRAINT "XOR\_REPORTS\_WITH\_VH\_U01" UNIQUE ("REPORT\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XOR\_REPORTS\_WITH\_VH" ADD CONSTRAINT "XOR\_REPORTS\_WITH\_VH\_PK" PRIMARY KEY ("REPORT\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table TEST\_HEAD\_QT\_XREF

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF" ADD CONSTRAINT "TEST\_HEAD\_QT\_XREF\_PK" PRIMARY KEY ("TEST\_HEAD\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF" MODIFY ("STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF" MODIFY ("TEST\_HEAD\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEG\_INS\_EBH\_UPGRADED

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEG\_INS\_EBH\_UPGRADED" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEG\_INS\_EBH\_UPGRADED" MODIFY ("SITE\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEGMENTS\_MODIFIED\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MODIFIED\_WK" MODIFY ("CHG\_TS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MODIFIED\_WK" MODIFY ("NEW\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MODIFIED\_WK" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK" MODIFY ("TID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK" MODIFY ("XNG\_EQUIP\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."INCMSPP\_XCONN\_AUDIT\_SUMMARY\_WK" MODIFY ("XNG\_SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("CHG\_TS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("ACTION\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("SITE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("Z\_SITE\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("SEGMENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("SEGMENT\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("DOMAIN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("DOMAIN\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_DETAIL\_SAT" MODIFY ("RPT\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_NE\_VS\_XNG\_AUDIT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_NE\_VS\_XNG\_AUDIT\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NWF\_BAD\_ENBS\_DEL\_SATYA

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA" MODIFY ("EDIT\_OPERATION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_ENBS\_DEL\_SATYA" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table BTP\_XNG\_UPD\_DETAILS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS" MODIFY ("MATCH\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_DETAILS" MODIFY ("XNG\_EXTRACT\_DATE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_VLAN\_PARSED

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_VLAN\_PARSED" ADD CONSTRAINT "XNG\_VLAN\_PARSED\_PK" PRIMARY KEY ("CIRC\_PATH\_INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_VLAN\_PARSED" MODIFY ("CIRC\_PATH\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEMP\_SAT\_S2

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEMP\_SAT\_S2" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table USER\_PATH\_DOMAIN\_PRIMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."USER\_PATH\_DOMAIN\_PRIMARY" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."USER\_PATH\_DOMAIN\_PRIMARY" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."USER\_PATH\_DOMAIN\_PRIMARY" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."USER\_PATH\_DOMAIN\_PRIMARY" MODIFY ("CIRC\_PATH\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LIVE\_OBJ\_NL\_SITES\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMMARY" MODIFY ("PATH\_CNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMMARY" MODIFY ("SEG\_CNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMMARY" MODIFY ("EQUIP\_CNT" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_SS7\_PATHS\_PARSED

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_SS7\_PATHS\_PARSED" ADD CONSTRAINT "XNG\_SS7\_PARSED\_PK" PRIMARY KEY ("CIRC\_PATH\_INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_SS7\_PATHS\_PARSED" MODIFY ("PATH\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SS7\_PATHS\_PARSED" MODIFY ("CIRC\_PATH\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SS7\_PATHS\_PARSED" MODIFY ("CIRC\_PATH\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DISCONNECT\_REASON\_CODE\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE\_WRK" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE\_WRK" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE\_WRK" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_REASON\_CODE\_WRK" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table MSPP\_SUMMARY\_REGION\_MAP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP" MODIFY ("TID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP" MODIFY ("XNG\_EQUIP\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP" MODIFY ("XNG\_SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEGMENTS\_MISSING\_EST\_COST

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST" ADD CONSTRAINT "SEGMENTS\_MISSING\_EST\_COST\_PK" PRIMARY KEY ("CIRC\_INST\_ID", "DOMAIN")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST" MODIFY ("IS\_COST\_MISSING" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MISSING\_EST\_COST" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEMP\_XNG\_CSR

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_CSR" MODIFY ("CSR\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_CSR" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNGSTD\_FAULT\_SUM

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SUM" MODIFY ("CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SUM" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SUM" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEGMENT\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEGMENT\_SUMMARY" MODIFY ("EXTRACT\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENT\_SUMMARY" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENT\_SUMMARY" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TDM\_DISCONNECT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TDM\_DISCONNECT\_WK" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table HIGH\_CAP\_PATH\_CHAN\_05142014

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAP\_PATH\_CHAN\_05142014" MODIFY ("PATHPARENT\_BANDWIDTH" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAP\_PATH\_CHAN\_05142014" MODIFY ("PATHPARENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAP\_PATH\_CHAN\_05142014" MODIFY ("PATHPARENT\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DF\_STATUS\_DETAILS\_NEW

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_NEW" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_NEW" MODIFY ("SUB\_MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_NEW" MODIFY ("MARKET" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DEL\_ENODEBS\_TYPES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DEL\_ENODEBS\_TYPES" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DEL\_ENODEBS\_TYPES" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DEL\_ENODEBS\_TYPES" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DEL\_ENODEBS\_TYPES" MODIFY ("TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table READ\_ONLY\_OLD\_CK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."READ\_ONLY\_OLD\_CK" MODIFY ("ATTR\_VALUE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."READ\_ONLY\_OLD\_CK" MODIFY ("ATTR\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."READ\_ONLY\_OLD\_CK" MODIFY ("DESCR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_NGMLS\_VLAN\_PATHS\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_VLAN\_PATHS\_WK" MODIFY ("VLAN\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_VLAN\_PATHS\_WK" MODIFY ("PATH\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_VLAN\_PATHS\_WK" MODIFY ("PORT\_BANDWIDTH" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_VLAN\_PATHS\_WK" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_NGMLS\_VLAN\_PATHS\_WK" MODIFY ("NE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NOCC\_LAST\_TEST\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_SUMMARY" MODIFY ("RUN\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_SUMMARY" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_SUMMARY" MODIFY ("DOMAIN\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_SUMMARY" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NOCC\_LAST\_TEST\_SUMMARY" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table REPORT\_DCOM\_PATH\_DETAIL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" MODIFY ("ROW\_NUM" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" MODIFY ("REPORT\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" ADD CONSTRAINT "REPORT\_DCOM\_PATH\_DETAIL\_PK" PRIMARY KEY ("REPORT\_ID", "ROW\_NUM", "AREA", "REGION")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table SAM\_CSR\_VLAN\_AUDIT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT\_WK" MODIFY ("CSR\_VENDOR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEMP\_SAT\_A1

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEMP\_SAT\_A1" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEMP\_S2

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEMP\_S2" MODIFY ("NAURU\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEMP\_S2" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEMP\_S2" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEGS\_UPGRADED\_EBH

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEGS\_UPGRADED\_EBH" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGS\_UPGRADED\_EBH" MODIFY ("SITE\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_SUMMIT\_ANT\_EXTRACT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT" ADD CONSTRAINT "XNG\_SUMMIT\_ANT\_EXTRACT\_PK" PRIMARY KEY ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "EQUIP\_INST\_ID", "NAURU\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT" MODIFY ("NAURU\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT" MODIFY ("ANTENNA\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DF\_STATUS\_DETAILS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table MOTO\_CDMA\_SPANS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."MOTO\_CDMA\_SPANS" ADD CONSTRAINT "MOTO\_CDMA\_SPANS\_PK" PRIMARY KEY ("ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."MOTO\_CDMA\_SPANS" MODIFY ("VSM\_DEVICE\_NAME\_OMCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MOTO\_CDMA\_SPANS" MODIFY ("ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table T1\_EXEMPTIONS\_ID\_DEL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."T1\_EXEMPTIONS\_ID\_DEL" MODIFY ("T1\_EXEMPTION\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."T1\_EXEMPTIONS\_ID\_DEL" ADD CONSTRAINT "PK\_T1\_EXEMPTIONS\_ID" PRIMARY KEY ("T1\_EXEMPTION\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table MOTO\_CDMA\_T1\_INV

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."MOTO\_CDMA\_T1\_INV" MODIFY ("VSM\_DEVICE\_NAME\_OMCR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table REPORT\_DCOM\_PATH\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_SUMMARY" MODIFY ("ROW\_NUM" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_SUMMARY" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_SUMMARY" MODIFY ("REPORT\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_SUMMARY" ADD CONSTRAINT "REPORT\_DCOM\_PATH\_SUMMARY\_PK" PRIMARY KEY ("REPORT\_ID", "ROW\_NUM")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table SEGMENTS\_TERM\_EXPIRATION

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION" ADD CONSTRAINT "SEGMENTS\_TERM\_EXPIRATION\_PK" PRIMARY KEY ("CIRC\_INST\_ID", "DOMAIN")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION" MODIFY ("IS\_TERM\_EXPIRED" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table ALL\_PROCESSES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."ALL\_PROCESSES" ADD CONSTRAINT "ALL\_PROCESSES\_U01" UNIQUE ("PROCESS\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."ALL\_PROCESSES" ADD CONSTRAINT "ALL\_PROCESSES\_PK" PRIMARY KEY ("PROCESS\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."ALL\_PROCESSES" MODIFY ("EXEC\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."ALL\_PROCESSES" MODIFY ("PROCESS\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NCM\_NGMLS\_VLAN\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_AUDIT" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_AUDIT" MODIFY ("NGMLS\_VENDOR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table USER\_CIRC\_DOMAIN\_PRIMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."USER\_CIRC\_DOMAIN\_PRIMARY" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."USER\_CIRC\_DOMAIN\_PRIMARY" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."USER\_CIRC\_DOMAIN\_PRIMARY" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."USER\_CIRC\_DOMAIN\_PRIMARY" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NCM\_CSR\_VLAN\_AUDITS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDITS" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDITS" MODIFY ("CSR\_VENDOR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table EQUIPMENT\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."EQUIPMENT\_SUMMARY" MODIFY ("EXTRACT\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."EQUIPMENT\_SUMMARY" MODIFY ("VSM\_SWITCH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."EQUIPMENT\_SUMMARY" MODIFY ("VSM\_DACS\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."EQUIPMENT\_SUMMARY" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."EQUIPMENT\_SUMMARY" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table WO\_BY\_Q\_TEMP\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_Q\_TEMP\_AUDIT" MODIFY ("FIELD\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_Q\_TEMP\_AUDIT" MODIFY ("EDIT\_OPERATION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_Q\_TEMP\_AUDIT" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_Q\_TEMP\_AUDIT" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_BY\_Q\_TEMP\_AUDIT" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table WD\_ERROR\_DETAILS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WD\_ERROR\_DETAILS" MODIFY ("LOG\_DATE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table ORDER\_REQD\_FIELDS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."ORDER\_REQD\_FIELDS" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table VPI\_GI\_AUDIT\_FIXES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VPI\_GI\_AUDIT\_FIXES" MODIFY ("GI\_SITE\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table RBED\_SEVONE\_OBJECT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_OBJECT" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_OBJECT" MODIFY ("DEVICEID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table COPY\_WK\_TABLES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."COPY\_WK\_TABLES" ADD CONSTRAINT "COPY\_WK\_TABLES\_PK" PRIMARY KEY ("PROCESS\_ID", "SEQ\_NO")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table INC\_MSPP\_XCONN\_AUDIT\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_SUMMARY" ADD CONSTRAINT "INC\_MSPP\_XCONN\_AUDIT\_SUMMA\_U01" UNIQUE ("TID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_SUMMARY" MODIFY ("TID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_SUMMARY" MODIFY ("XNG\_EQUIP\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."INC\_MSPP\_XCONN\_AUDIT\_SUMMARY" MODIFY ("XNG\_SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CIRC\_PATH\_ATTR\_SETT\_WL\_0513

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CIRC\_PATH\_ATTR\_SETT\_WL\_0513" MODIFY ("ATTR\_VALUE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_PATH\_ATTR\_SETT\_WL\_0513" MODIFY ("VAL\_ATTR\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_PATH\_ATTR\_SETT\_WL\_0513" MODIFY ("CIRC\_PATH\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_SUMMIT\_SITE\_EXTRACT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT" ADD CONSTRAINT "XNG\_SUMMIT\_SITE\_EXTRACT\_PK" PRIMARY KEY ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "NAURU\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT" MODIFY ("NAURU\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XOR\_MENUS\_REPORTS\_LINK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XOR\_MENUS\_REPORTS\_LINK" ADD CONSTRAINT "XOR\_MENUS\_REPORTS\_LINK\_PK" PRIMARY KEY ("REPORT\_ID", "MENU\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table INC\_DEVICES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."INC\_DEVICES" MODIFY ("TID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNGSTD\_FAULT\_SUM\_BK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SUM\_BK" MODIFY ("CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SUM\_BK" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_SUM\_BK" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEVONE\_PERFORMANCE\_DATA

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEVONE\_PERFORMANCE\_DATA" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEVONE\_PERFORMANCE\_DATA" MODIFY ("OBJECT\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEVONE\_PERFORMANCE\_DATA" MODIFY ("DEVICE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEVONE\_PERFORMANCE\_DATA" MODIFY ("DEVICE\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_JOB\_STATUS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_JOB\_STATUS" ADD CONSTRAINT "XNG\_JOB\_STATUS\_PK" PRIMARY KEY ("JOB\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_JOB\_STATUS" MODIFY ("RUN\_TIME\_IN\_MINUTES" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_JOB\_STATUS" MODIFY ("JOB\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XOR\_MENUS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XOR\_MENUS" ADD CONSTRAINT "XOR\_MENUS\_U01" UNIQUE ("MENU\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XOR\_MENUS" ADD CONSTRAINT "XOR\_MENUS\_PK" PRIMARY KEY ("MENU\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table NSP\_LIGHT\_REQD

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NSP\_LIGHT\_REQD" MODIFY ("FIELD\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NSP\_LIGHT\_REQD" MODIFY ("EDIT\_OPERATION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NSP\_LIGHT\_REQD" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NSP\_LIGHT\_REQD" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NSP\_LIGHT\_REQD" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_SUMMIT\_SEG\_EXTRACT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT" ADD CONSTRAINT "XNG\_SUMMIT\_SEG\_EXTRACT\_PK" PRIMARY KEY ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "CIRC\_INST\_ID", "NAURU\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT" MODIFY ("CIRCUIT\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT" MODIFY ("NAURU\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEMP\_XNG\_CSR\_PORT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_CSR\_PORT" MODIFY ("MSC\_CSR\_PORT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_CSR\_PORT" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DARK\_FIBER\_STATUS\_WK\_DEL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_WK\_DEL" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_WK\_DEL" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_WK\_DEL" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEMP\_XNG\_ENODE\_B\_SATYA

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B\_SATYA" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B\_SATYA" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B\_SATYA" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_ADM\_NE\_VS\_XNG\_ADT\_TMP" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_NE\_VS\_XNG\_AUDIT\_TEMP" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table FIELDS\_INFO\_UDA

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA" MODIFY ("ATTR\_DATA\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA" MODIFY ("ATTR\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."FIELDS\_INFO\_UDA" MODIFY ("GROUP\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LDAP\_XNG\_MISMATCH\_DUPS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LDAP\_XNG\_MISMATCH\_DUPS" MODIFY ("USER\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table PROCESS\_DEPENDENCY\_MATRIX

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."PROCESS\_DEPENDENCY\_MATRIX" ADD CONSTRAINT "PROCESS\_DEPENDENCY\_MATRIX\_PK" PRIMARY KEY ("PROCESS\_ID", "DS\_PROCESS\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table SAM\_NGMLS\_VLAN\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT" MODIFY ("NGMLS\_VENDOR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEVONE\_CSR

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEVONE\_CSR" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEVONE\_CSR" MODIFY ("DEVICE\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XOR\_TABS\_MENUS\_LINK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XOR\_TABS\_MENUS\_LINK" ADD CONSTRAINT "XOR\_TABS\_MENUS\_LINK\_PK" PRIMARY KEY ("MENU\_ID", "TAB\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table BTP\_DATA\_FOR\_UPD\_OLD

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."BTP\_DATA\_FOR\_UPD\_OLD" MODIFY ("EXTRACT\_DATE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NWF\_BAD\_END

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_END" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_END" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_END" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_END" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_END" MODIFY ("EDIT\_OPERATION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_END" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_END" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NWF\_BAD\_END" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK" MODIFY ("CLLI\_6" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK" MODIFY ("LEAF\_DOMAIN\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK" MODIFY ("MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_HPOV\_EH\_GI\_CSR\_CLLI\_SUM\_WK" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table WD\_ERROR\_CODE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WD\_ERROR\_CODE" ADD CONSTRAINT "WD\_ERROR\_CODE\_PK" PRIMARY KEY ("ERROR\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table XNG\_CONTACTS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_CONTACTS" MODIFY ("SITE\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEMP\_SAT\_S1

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEMP\_SAT\_S1" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEST\_HEAD\_XREF

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEST\_HEAD\_XREF" ADD CONSTRAINT "TEST\_HEAD\_XREF\_PK" PRIMARY KEY ("TEST\_HEAD\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."TEST\_HEAD\_XREF" MODIFY ("STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEST\_HEAD\_XREF" MODIFY ("TEST\_HEAD\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table JUNIPER\_DEVICES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_DEVICES" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_DEVICES" MODIFY ("DEVICE\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_DEVICES" MODIFY ("DEVICE\_IP" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_DEVICES" MODIFY ("HOST\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_DEVICES" ADD CONSTRAINT "JUNIPER\_DEVICES\_PK" PRIMARY KEY ("HOST\_NAME", "DEVICE\_IP", "DEVICE\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table VPI\_GI\_ALT\_VALS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VPI\_GI\_ALT\_VALS" MODIFY ("SITE\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SWITCH\_PATH\_T1\_TESTABILITY\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" ADD CONSTRAINT "PK\_SWITCH\_PATH\_TESTABILITY\_WK" PRIMARY KEY ("SWITCH\_PATH\_T1\_TESTABILITY\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("MATCHING\_LIVE\_XNG\_PATHS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("IS\_ACTIVE\_IN\_XNG" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("IS\_FOUND\_IN\_XNG" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("IS\_ACTIVE\_IN\_NE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("DEVICE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("NETWORK\_ELEMENT\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("SPAN\_TECHNOLOGY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("DOMAIN\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_T1\_TESTABILITY\_WK" MODIFY ("SWITCH\_PATH\_T1\_TESTABILITY\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table HIGH\_CAPACITY\_PATH\_CHAN

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN" MODIFY ("PATHPARENT\_BANDWIDTH" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN" MODIFY ("PATHPARENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN" MODIFY ("PATHPARENT\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CAS\_5557\_5564\_5555

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CAS\_5557\_5564\_5555" MODIFY ("ATTR\_VALUE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CAS\_5557\_5564\_5555" MODIFY ("VAL\_ATTR\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CAS\_5557\_5564\_5555" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_CSR\_PARSED

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_CSR\_PARSED" ADD CONSTRAINT "XNG\_CSR\_PARSED\_PK" PRIMARY KEY ("EQUIP\_INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_CSR\_PARSED" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table MICROWAVE\_UTIL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL" MODIFY ("SITE\_A" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL" MODIFY ("EQUIP\_VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL" MODIFY ("EQUIP\_CATEGORY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL" MODIFY ("PATH\_CATEGORY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL" MODIFY ("PATH\_BANDWIDTH" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL" MODIFY ("PATH\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_ADM\_FW\_EQUIP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_ADM\_FW\_EQUIP" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_ADM\_FW\_EQUIP" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_ADM\_FW\_EQUIP" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_ADM\_FW\_EQUIP" MODIFY ("EQ\_CLASS\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_ADM\_FW\_EQUIP" MODIFY ("NE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DARK\_FIBER\_STATUS\_DEL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_DEL" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_DEL" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_DEL" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_MWR\_PARSED

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_MWR\_PARSED" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MWR\_PARSED" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table MSPP\_SUMMARY\_REGION\_MAP\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP\_WK" MODIFY ("TID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP\_WK" MODIFY ("XNG\_EQUIP\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MSPP\_SUMMARY\_REGION\_MAP\_WK" MODIFY ("XNG\_SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_CSR\_PARSED\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_CSR\_PARSED\_WK" ADD CONSTRAINT "XNG\_CSR\_PARSED\_WK\_PK" PRIMARY KEY ("EQUIP\_INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XNG\_CSR\_PARSED\_WK" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table RBED\_SEVONE\_DATA

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DATA" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DATA" MODIFY ("INDICATOR\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DATA" MODIFY ("DEVICE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."RBED\_SEVONE\_DATA" MODIFY ("DEVICE\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table ORDER\_REQD\_FIELDS\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."ORDER\_REQD\_FIELDS\_WRK" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table INC\_MSPP\_NE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."INC\_MSPP\_NE" ADD CONSTRAINT "INC\_MSPP\_NE\_PK" PRIMARY KEY ("NEID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."INC\_MSPP\_NE" ADD CONSTRAINT "INC\_MSPP\_NE\_C01" CHECK (TID is not null) ENABLE;

ALTER TABLE "XNG\_REPORTS"."INC\_MSPP\_NE" ADD CONSTRAINT "INC\_MSPP\_NE\_U01" UNIQUE ("TID", "GNE\_TID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table DECOMM\_REQD\_FIELDS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DECOMM\_REQD\_FIELDS" MODIFY ("RUN\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DECOMM\_REQD\_FIELDS" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DECOMM\_REQD\_FIELDS" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DECOMM\_REQD\_FIELDS" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DECOMM\_REQD\_FIELDS" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table INCOMPLETE\_PATH

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."INCOMPLETE\_PATH" MODIFY ("CIRC\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."INCOMPLETE\_PATH" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."INCOMPLETE\_PATH" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."INCOMPLETE\_PATH" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."INCOMPLETE\_PATH" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SITE\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SITE\_SUMMARY" MODIFY ("EXTRACT\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_SUMMARY" MODIFY ("SITE\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_SUMMARY" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_SUMMARY" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_VLAN\_MSPP\_TAM\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_VLAN\_MSPP\_TAM\_WK" MODIFY ("PORT\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LDAP\_XNG\_MATCH\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LDAP\_XNG\_MATCH\_WRK" MODIFY ("USER\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SAM\_NGMLS\_VLAN\_AUDIT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SAM\_NGMLS\_VLAN\_AUDIT\_WK" MODIFY ("NGMLS\_VENDOR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_MSPP\_DEVICES\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES\_WK" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES\_WK" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES\_WK" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES\_WK" MODIFY ("TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table UNQ\_XNG\_CONTACTS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."UNQ\_XNG\_CONTACTS" MODIFY ("SITE\_HUM\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table HIGH\_CAPACITY\_PATH\_CHAN\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_WK" MODIFY ("PATHPARENT\_BANDWIDTH" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_WK" MODIFY ("PATHPARENT\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_WK" MODIFY ("PATHPARENT\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NCM\_NGMLS\_VLAN\_AUDIT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_AUDIT\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_NGMLS\_VLAN\_AUDIT\_WK" MODIFY ("NGMLS\_VENDOR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_CSR\_VLAN\_PATHS\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_CSR\_VLAN\_PATHS\_WK" MODIFY ("VLAN\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_CSR\_VLAN\_PATHS\_WK" MODIFY ("PATH\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_CSR\_VLAN\_PATHS\_WK" MODIFY ("PORT\_BANDWIDTH" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_CSR\_VLAN\_PATHS\_WK" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SWITCH\_PATH\_ISSUE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_ISSUE" ADD CONSTRAINT "PK\_SWITCH\_PATH\_ISSUE" PRIMARY KEY ("SWITCH\_PATH\_ISSUE\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_ISSUE" MODIFY ("IS\_CRITICAL" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_ISSUE" MODIFY ("DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SWITCH\_PATH\_ISSUE" MODIFY ("SWITCH\_PATH\_ISSUE\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table TEMP\_XNG\_ENODE\_B

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B" MODIFY ("SITE\_DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEMP\_XNG\_ENODE\_B" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CSR\_ISSUE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CSR\_ISSUE" ADD UNIQUE ("DESCRIPTION")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."CSR\_ISSUE" ADD PRIMARY KEY ("CSR\_ISSUE\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."CSR\_ISSUE" MODIFY ("IS\_CRITICAL" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CSR\_ISSUE" MODIFY ("DESCRIPTION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CSR\_ISSUE" MODIFY ("CSR\_ISSUE\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table JUNIPER\_XNG\_EQUIP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_XNG\_EQUIP" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_XNG\_EQUIP" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_XNG\_EQUIP" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_XNG\_EQUIP" MODIFY ("EQ\_CLASS\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_XNG\_EQUIP" MODIFY ("NE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table VSM\_DEVICES\_DOMAIN\_MAP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VSM\_DEVICES\_DOMAIN\_MAP" ADD CONSTRAINT "VSM\_DEVICES\_DOMAIN\_MAP\_PK" PRIMARY KEY ("VSM\_DEVICE\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."VSM\_DEVICES\_DOMAIN\_MAP" MODIFY ("LEAF\_DOMAIN\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LIVE\_OBJ\_NL\_SITES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES" MODIFY ("PATH\_CNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES" MODIFY ("SEG\_CNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES" MODIFY ("EQUIP\_CNT" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table GRANITE\_IP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."GRANITE\_IP" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."GRANITE\_IP" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."GRANITE\_IP" MODIFY ("IP" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."GRANITE\_IP" MODIFY ("ATTR\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."GRANITE\_IP" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."GRANITE\_IP" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XOR\_TABS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XOR\_TABS" ADD CONSTRAINT "XOR\_TABS\_U01" UNIQUE ("TAB\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."XOR\_TABS" ADD CONSTRAINT "XOR\_TABS\_PK" PRIMARY KEY ("TAB\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table NETSMART\_SNCNE\_AUDIT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT\_WK" ADD CONSTRAINT "NETSMART\_SNCNE\_AUDIT\_WK\_PK" PRIMARY KEY ("INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT\_WK" MODIFY ("TID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table JUNIPER\_VLANS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_VLANS" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_VLANS" MODIFY ("VLAN\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_VLANS" MODIFY ("DEVICE\_IP" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_VLANS" MODIFY ("HOST\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table VZW\_NETWORK\_ORG\_NEW

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_NEW" MODIFY ("TERRITORY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_NEW" MODIFY ("MARKET\_TERRITORY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_NEW" MODIFY ("MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_NEW" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_NEW" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG\_NEW" MODIFY ("INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LIVE\_OBJ\_NL\_SITES\_SUMM\_WK\_SEG

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMM\_WK\_SEG" MODIFY ("SEG\_CNT" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table PATH\_TYPE\_TO\_SERVICE\_MAP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."PATH\_TYPE\_TO\_SERVICE\_MAP" ADD PRIMARY KEY ("PATH\_TYPE")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 1048576 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."PATH\_TYPE\_TO\_SERVICE\_MAP" MODIFY ("SERVICE\_DESC" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."PATH\_TYPE\_TO\_SERVICE\_MAP" MODIFY ("PATH\_TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table MICROWAVE\_UTIL\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK" MODIFY ("SITE\_A" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK" MODIFY ("EQUIP\_VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK" MODIFY ("EQUIP\_CATEGORY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK" MODIFY ("PATH\_CATEGORY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK" MODIFY ("PATH\_BANDWIDTH" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK" MODIFY ("PATH\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."MICROWAVE\_UTIL\_WRK" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table PATH\_STANDARD\_REGEX

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."PATH\_STANDARD\_REGEX" MODIFY ("PATTERN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."PATH\_STANDARD\_REGEX" MODIFY ("REGEX\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table AUDIT\_STATUS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."AUDIT\_STATUS" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."AUDIT\_STATUS" MODIFY ("AUDIT\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."AUDIT\_STATUS" MODIFY ("INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table VZW\_NETWORK\_ORG

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG" MODIFY ("TERRITORY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG" MODIFY ("MARKET\_TERRITORY" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG" MODIFY ("MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_NETWORK\_ORG" MODIFY ("INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_FW\_EQUIP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_FW\_EQUIP" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_FW\_EQUIP" MODIFY ("TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_FW\_EQUIP" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_FW\_EQUIP" MODIFY ("EQ\_CLASS\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_FW\_EQUIP" MODIFY ("NE\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table EST\_COST\_AUDIT\_DETAIL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."EST\_COST\_AUDIT\_DETAIL" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."EST\_COST\_AUDIT\_DETAIL" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."EST\_COST\_AUDIT\_DETAIL" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_MSPP\_DEVICES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES" MODIFY ("SITE\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES" MODIFY ("VENDOR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MSPP\_DEVICES" MODIFY ("TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table BTP\_XNG\_UPD\_AUDIT\_LOG\_FIX

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_AUDIT\_LOG\_FIX" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."BTP\_XNG\_UPD\_AUDIT\_LOG\_FIX" MODIFY ("INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNGSTD\_AUDIT\_PROPERTIES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_AUDIT\_PROPERTIES" MODIFY ("VALUE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_AUDIT\_PROPERTIES" MODIFY ("NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_SNCNE\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_SNCNE\_AUDIT" MODIFY ("TID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NCM\_CSR\_VLAN\_AUDIT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDIT\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NCM\_CSR\_VLAN\_AUDIT\_WK" MODIFY ("CSR\_VENDOR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_EQPT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_EQPT" MODIFY ("SNCNE\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NETSMART\_EQPT" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NETSMART\_EQPT" ADD CONSTRAINT "NETSMART\_EQPT\_PK" PRIMARY KEY ("INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table TEST\_HEAD\_QT\_XREF\_BAK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF\_BAK" MODIFY ("STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."TEST\_HEAD\_QT\_XREF\_BAK" MODIFY ("TEST\_HEAD\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CLLI\_DOMAIN\_MAP\_BAK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP\_BAK" MODIFY ("LEAF\_DOMAIN\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("CABLE\_ALL\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("CABLE\_NON\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("CABLE\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("PATH\_ALL\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("PATH\_NON\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("PATH\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("EQUIP\_ALL\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("EQUIP\_NON\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("EQUIP\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("SEG\_ALL\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("SEG\_NON\_EBH\_COUNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NL\_OBJS\_IN\_LIVE\_PATHS\_SUMM\_WK" MODIFY ("SEG\_EBH\_COUNT" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table BTP\_SEG\_ID\_CLEANUP\_LOG

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."BTP\_SEG\_ID\_CLEANUP\_LOG" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."BTP\_SEG\_ID\_CLEANUP\_LOG" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEGMENTS\_MODIFIED

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MODIFIED" MODIFY ("CHG\_TS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MODIFIED" MODIFY ("NEW\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_MODIFIED" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DF\_STATUS\_DETAILS\_NEW\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_NEW\_WK" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_NEW\_WK" MODIFY ("SUB\_MARKET" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DF\_STATUS\_DETAILS\_NEW\_WK" MODIFY ("MARKET" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."JUNIPER\_NE\_VS\_XNG\_AUDIT\_WK" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NE\_VS\_XNG\_SP\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY" MODIFY ("TOTAL" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY" MODIFY ("PER\_MATCHED" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY" MODIFY ("NO\_MATCH\_IN\_GI" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NE\_VS\_XNG\_SP\_SUMMARY" MODIFY ("MATCHED\_GI" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table VZW\_INST\_AUDIT\_WINDOW

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW" MODIFY ("EDIT\_OPERATION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."VZW\_INST\_AUDIT\_WINDOW" MODIFY ("ELEMENT\_TYPE" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LIVE\_OBJ\_NL\_SITES\_SUMM\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMM\_WK" MODIFY ("PATH\_CNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMM\_WK" MODIFY ("SEG\_CNT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LIVE\_OBJ\_NL\_SITES\_SUMM\_WK" MODIFY ("EQUIP\_CNT" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CIRC\_SEG\_NEW\_DECOM\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("PERCENTAGE\_NEW\_BY\_TESS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("COUNT\_NEW\_NOT\_TESS\_NO\_CONTRACT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("COUNT\_NEW\_NOT\_TESS\_CONTRACT" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("COUNT\_NEW\_BY\_TESS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("COUNT\_NEW\_TOTAL" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("COUNT\_DECOM" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("COUNT\_NEW" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("AREA" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("PERIOD\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_SUMMARY" MODIFY ("PERIOD" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SAM\_CSR\_VLAN\_AUDIT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT" MODIFY ("MATCH\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SAM\_CSR\_VLAN\_AUDIT" MODIFY ("CSR\_VENDOR" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SEGMENTS\_TERM\_EXPIRATION\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION\_WK" MODIFY ("IS\_TERM\_EXPIRED" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION\_WK" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SEGMENTS\_TERM\_EXPIRATION\_WK" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNGSTD\_AUDIT\_SUMMARY\_REPORT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_AUDIT\_SUMMARY\_REPORT" MODIFY ("CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_AUDIT\_SUMMARY\_REPORT" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_AUDIT\_SUMMARY\_REPORT" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table OSS\_RC\_ENB

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."OSS\_RC\_ENB" ADD CONSTRAINT "OSS\_RC\_ENB\_PK" PRIMARY KEY ("NODE", "SERVER\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table REPORT\_CONTROL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."REPORT\_CONTROL" MODIFY ("RUN\_STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_CONTROL" MODIFY ("SYS\_CREATION\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_CONTROL" MODIFY ("RUN\_DATE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_CONTROL" MODIFY ("REPORT\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_CONTROL" MODIFY ("REPORT\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_CONTROL" ADD CONSTRAINT "REPORT\_CONTROL\_PK" PRIMARY KEY ("REPORT\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table DISCONNECT\_AGING\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING\_WRK" MODIFY ("STATUS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING\_WRK" MODIFY ("CIRC\_HUM\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING\_WRK" MODIFY ("CIRC\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING\_WRK" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DISCONNECT\_AGING\_WRK" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CIRC\_PATH\_ATTR\_SETTINGS\_0608

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CIRC\_PATH\_ATTR\_SETTINGS\_0608" MODIFY ("ATTR\_VALUE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_PATH\_ATTR\_SETTINGS\_0608" MODIFY ("VAL\_ATTR\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_PATH\_ATTR\_SETTINGS\_0608" MODIFY ("CIRC\_PATH\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LEASED\_SEG\_DF\_DETAILS\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("DF\_SEG\_BW" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("DF\_SEG\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("NDF\_SEG\_BW" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("NDF\_SEG\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("NDF\_SEG\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("DF\_SITE\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LEASED\_SEG\_DF\_DETAILS\_WK" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN" MODIFY ("PROC\_START\_TS" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."CIRC\_SEG\_NEW\_DECOM\_RPT\_GEN" MODIFY ("RPT\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table VZW\_VSM\_XNG\_XREF\_BK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."VZW\_VSM\_XNG\_XREF\_BK" ADD PRIMARY KEY ("VSM\_DEVICE\_NAME")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table CLLI\_DOMAIN\_MAP

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP" ADD CONSTRAINT "CLLI\_DOMAIN\_MAP\_PK" PRIMARY KEY ("CLLI", "LEAF\_DOMAIN\_INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."CLLI\_DOMAIN\_MAP" MODIFY ("LEAF\_DOMAIN\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table DARK\_FIBER\_STATUS\_NEW\_DEL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_NEW\_DEL" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_NEW\_DEL" MODIFY ("REGION" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."DARK\_FIBER\_STATUS\_NEW\_DEL" MODIFY ("AREA" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NETSMART\_SNCNE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NETSMART\_SNCNE" MODIFY ("TID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NETSMART\_SNCNE" MODIFY ("INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."NETSMART\_SNCNE" ADD CONSTRAINT "NETSMART\_SNCNE\_PK" PRIMARY KEY ("INST\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table LUCENT\_AUDIT\_SWITCH\_SUMM\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LUCENT\_AUDIT\_SWITCH\_SUMM\_WRK" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LUCENT\_AUDIT\_SWITCH\_SUMM\_WRK" MODIFY ("REGION" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table T1\_ISSUES\_ID\_DEL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."T1\_ISSUES\_ID\_DEL" ADD CONSTRAINT "PK\_T1\_ISSUES\_ID" PRIMARY KEY ("T1\_ISSUE\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table REPORT\_DCOM\_PATH\_NOTIFICATION

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" ADD CONSTRAINT "REPORT\_DCOM\_PATH\_NOTIF\_PK" PRIMARY KEY ("USER\_NAME", "CIRC\_PATH\_INST\_ID", "AREA", "REGION")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("STATUS" CONSTRAINT "DPN\_STATUS\_NN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("NOTIFICATION\_DATE" CONSTRAINT "DPN\_NOTIF\_DATE\_NN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("NOTIFICATION\_ID" CONSTRAINT "DPN\_NOTIFICATION\_ID\_NN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("REPORT\_ID" CONSTRAINT "DPN\_REPORT\_ID\_NN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("SYS\_CREATION\_DATE" CONSTRAINT "DPN\_SYS\_CRE\_DATE\_NN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("CIRC\_PATH\_INST\_ID" CONSTRAINT "DPN\_CPI\_NN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("REGION" CONSTRAINT "DPN\_REGION\_NN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("AREA" CONSTRAINT "DPN\_AREA\_NN" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_NOTIFICATION" MODIFY ("USER\_NAME" CONSTRAINT "DPN\_USER\_NAME\_NN" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table SITE\_PORTAL\_DOMAIN\_MAPPING

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."SITE\_PORTAL\_DOMAIN\_MAPPING" MODIFY ("SITE\_PORTAL\_DOMAIN\_MAPPING\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_PORTAL\_DOMAIN\_MAPPING" MODIFY ("DOMAIN\_INST\_ID" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_PORTAL\_DOMAIN\_MAPPING" MODIFY ("SITE\_PORTAL\_DOMAIN\_CODE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."SITE\_PORTAL\_DOMAIN\_MAPPING" ADD CONSTRAINT "SITE\_PORTAL\_DOMAIN\_MAPPING\_PK" PRIMARY KEY ("SITE\_PORTAL\_DOMAIN\_MAPPING\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table HIGH\_CAPACITY\_PATH\_CHAN\_CHILD

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_CHILD" MODIFY ("PATH\_TYPE" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."HIGH\_CAPACITY\_PATH\_CHAN\_CHILD" MODIFY ("PATH\_NAME" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table XNG\_MWR\_PARSED\_WK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_MWR\_PARSED\_WK" MODIFY ("DESCR" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."XNG\_MWR\_PARSED\_WK" MODIFY ("EQUIP\_INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table LUCENT\_AUDIT\_SWITCH\_SUMM

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."LUCENT\_AUDIT\_SWITCH\_SUMM" MODIFY ("LEAF\_DOMAIN\_NAME" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."LUCENT\_AUDIT\_SWITCH\_SUMM" MODIFY ("REGION" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table CSR\_DEVICES

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CSR\_DEVICES" ADD CONSTRAINT "CSR\_DEVICES\_U01" UNIQUE ("CSR\_DEVICE\_NAME") DISABLE;

ALTER TABLE "XNG\_REPORTS"."CSR\_DEVICES" ADD CONSTRAINT "CSR\_DEVICES\_PK" PRIMARY KEY ("CSR\_ID")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table WO\_TASK\_AGING\_DATES\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WO\_TASK\_AGING\_DATES\_WRK" MODIFY ("VERSION\_CHGD" NOT NULL ENABLE);

ALTER TABLE "XNG\_REPORTS"."WO\_TASK\_AGING\_DATES\_WRK" MODIFY ("INST\_ID" NOT NULL ENABLE);

--------------------------------------------------------

-- Constraints for Table NORTEL\_DOM\_INVENTORY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NORTEL\_DOM\_INVENTORY" ADD PRIMARY KEY ("SEQ\_NUM")

USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS

STORAGE(INITIAL 8388608 NEXT 8388608 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "SS\_DATA00" ENABLE;

--------------------------------------------------------

-- Constraints for Table XNGSTD\_FAULT\_EQUIP\_CNT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNGSTD\_FAULT\_EQUIP\_CNT" MODIFY ("CREATION\_DATE" NOT NULL ENABLE);

--------------------------------------------------------

-- Ref Constraints for Table CSR\_AUDIT\_ISSUE

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."CSR\_AUDIT\_ISSUE" ADD CONSTRAINT "CSR\_AUDIT\_ISSUE\_R01" FOREIGN KEY ("CSR\_ISSUE\_ID")

REFERENCES "XNG\_REPORTS"."CSR\_ISSUE" ("CSR\_ISSUE\_ID") ENABLE;

--------------------------------------------------------

-- Ref Constraints for Table NORTEL\_PORT\_INVENTORY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NORTEL\_PORT\_INVENTORY" ADD CONSTRAINT "DOM\_REF" FOREIGN KEY ("DOM\_SEQ\_NUM")

REFERENCES "XNG\_REPORTS"."NORTEL\_DOM\_INVENTORY" ("SEQ\_NUM") ENABLE;

--------------------------------------------------------

-- Ref Constraints for Table NORTEL\_PORT\_INVENTORY\_WRK

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."NORTEL\_PORT\_INVENTORY\_WRK" ADD CONSTRAINT "DOM\_SEQ\_REF" FOREIGN KEY ("DOM\_SEQ\_NUM")

REFERENCES "XNG\_REPORTS"."NORTEL\_DOM\_INVENTORY\_WRK" ("SEQ\_NUM") ENABLE;

--------------------------------------------------------

-- Ref Constraints for Table PROCESS\_DEPENDENCY\_MATRIX

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."PROCESS\_DEPENDENCY\_MATRIX" ADD CONSTRAINT "PROCESS\_DEPENDENCY\_MATRIX\_R01" FOREIGN KEY ("PROCESS\_ID")

REFERENCES "XNG\_REPORTS"."ALL\_PROCESSES" ("PROCESS\_ID") ON DELETE CASCADE ENABLE;

--------------------------------------------------------

-- Ref Constraints for Table REPORT\_DCOM\_PATH\_DETAIL

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_DETAIL" ADD CONSTRAINT "REPORT\_DCOM\_PATH\_DETAIL\_FK" FOREIGN KEY ("REPORT\_ID")

REFERENCES "XNG\_REPORTS"."REPORT\_CONTROL" ("REPORT\_ID") ON DELETE CASCADE ENABLE;

--------------------------------------------------------

-- Ref Constraints for Table REPORT\_DCOM\_PATH\_SUMMARY

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."REPORT\_DCOM\_PATH\_SUMMARY" ADD CONSTRAINT "REPORT\_DCOM\_PATH\_SUMMARY\_FK" FOREIGN KEY ("REPORT\_ID")

REFERENCES "XNG\_REPORTS"."REPORT\_CONTROL" ("REPORT\_ID") ON DELETE CASCADE ENABLE;

--------------------------------------------------------

-- Ref Constraints for Table WD\_ERROR\_DETAILS

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."WD\_ERROR\_DETAILS" ADD CONSTRAINT "WD\_ERROR\_DETAILS\_R02" FOREIGN KEY ("ERROR\_ID")

REFERENCES "XNG\_REPORTS"."WD\_ERROR\_CODE" ("ERROR\_ID") ENABLE;

--------------------------------------------------------

-- Ref Constraints for Table XNG\_SUMMIT\_ANT\_EXTRACT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_ANT\_EXTRACT" ADD CONSTRAINT "XNG\_SUMMIT\_ANT\_EXTRACT\_FK" FOREIGN KEY ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "NAURU\_ID")

REFERENCES "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT" ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "NAURU\_ID") ON DELETE CASCADE ENABLE;

--------------------------------------------------------

-- Ref Constraints for Table XNG\_SUMMIT\_SEG\_EXTRACT

--------------------------------------------------------

ALTER TABLE "XNG\_REPORTS"."XNG\_SUMMIT\_SEG\_EXTRACT" ADD CONSTRAINT "XNG\_SUMMIT\_SEG\_EXTRACT\_FK" FOREIGN KEY ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "NAURU\_ID")

REFERENCES "XNG\_REPORTS"."XNG\_SUMMIT\_SITE\_EXTRACT" ("SITE\_INST\_ID", "SITE\_DESCRIPTION", "NAURU\_ID") ON DELETE CASCADE ENABLE;