Database Sample Code

Contents

[1. External Tab Delimited File Example 2](#_Toc373244263)

[2. External CSV File Example No. 1 2](#_Toc373244264)

[3. External CSV File Example No. 2 (using regexp) 3](#_Toc373244265)

[4. External Restful Webservices Pipe Delimited File 4](#_Toc373244266)

[a) Embedded java 4](#_Toc373244267)

[b) Create DB Stored Procedure 9](#_Toc373244268)

[c) Call Java DB Stored Procedure 9](#_Toc373244269)

[5. Triggers for SCD Type 2 Data Model 9](#_Toc373244270)

[6. Package Example 17](#_Toc373244271)

[d) Simple Cursor 18](#_Toc373244272)

[e) Simple Logging Example 24](#_Toc373244273)

[f) Simple History of Status Example 24](#_Toc373244274)

[g) Bulk Collect User Defined Type Example 26](#_Toc373244275)

[h) Instantiate Bulk Collect User Defined Type Example 26](#_Toc373244276)

[i) Simple Set Operations 28](#_Toc373244277)

[j) Simple Case Statement 32](#_Toc373244278)

[k) Simple Decode Statement 32](#_Toc373244279)

[l) Access DBLink 33](#_Toc373244280)

[m) Bulk Collect 37](#_Toc373244281)

[n) Bulk Collect Cursor 37](#_Toc373244282)

[o) Simple DB Link Example 51](#_Toc373244283)

[p) Nested Cursor 52](#_Toc373244284)

[q) Exception Sample 56](#_Toc373244285)

# External Tab Delimited File Example

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

-- 1. First physically create the directory c:\temp

-- 2. Then login as system and do:

-- A. CREATE OR REPLACE DIRECTORY TMP AS 'C:\temp';

CREATE OR REPLACE DIRECTORY tmp AS 'C:\temp';

-- B. GRANT READ, WRITE ON DIRECTORY TMP TO SAMPLECODE

GRANT READ, WRITE ON DIRECTORY tmp TO SAMPLECODE;

-- 3. Create the external table

DROP TABLE invoices\_external;

SELECT COUNT(\*) FROM invoices\_external;

CREATE TABLE invoices\_external

(

invoice\_id CHAR(6),

invoice\_date CHAR(13),

invoice\_amt CHAR(9),

account\_number CHAR(11)

)

ORGANIZATION EXTERNAL

(

TYPE oracle\_loader DEFAULT

DIRECTORY tmp ACCESS

PARAMETERS (records delimited BY newline SKIP 2 fields

( invoice\_id CHAR(6), invoice\_date CHAR(13),

invoice\_amt CHAR(9), account\_number CHAR(11)))

-- make sure there are no extra lines that might be interpreted as null later when loading into PK

LOCATION ('load\_invoices.txt')

);

SELECT \* FROM invoices\_external;

-- create table with correct data types

CREATE TABLE invoices\_revised

(

invoice\_id INTEGER,

invoice\_date DATE,

invoice\_amt NUMBER,

account\_number VARCHAR2(13)

);

INSERT INTO invoices\_revised

(invoice\_id, invoice\_date,invoice\_amt,account\_number)

SELECT invoice\_id, to\_date(invoice\_date,'mm/dd/yyyy'),

to\_number(invoice\_amt), account\_number

FROM invoices\_external;

SELECT \* FROM invoices\_revised;

# External CSV File Example No. 1

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

-- 1. First physically create the directory c:\temp

-- 2. Then login as system and do:

-- A. CREATE OR REPLACE DIRECTORY TMP AS 'C:\temp';

CREATE OR REPLACE DIRECTORY tmp AS 'C:\temp';

-- B. GRANT READ, WRITE ON DIRECTORY TMP TO CRUISES;

GRANT READ, WRITE ON DIRECTORY tmp TO samplecode;

-- 3. Create the external table

DROP TABLE song\_test;

CREATE TABLE song\_test

(sid NUMBER,

aid number,

title VARCHAR2(175),

wrt VARCHAR2(175),

secs NUMBER)

ORGANIZATION EXTERNAL

(

TYPE ORACLE\_LOADER

DEFAULT DIRECTORY TMP

ACCESS PARAMETERS

(RECORDS DELIMITED BY NEWLINE

FIELDS TERMINATED BY ","

OPTIONALLY ENCLOSED BY '"'

MISSING FIELD VALUES ARE NULL

REJECT ROWS WITH ALL NULL FIELDS

( sid,

aid,

title,

wrt,

secs

)

)

LOCATION ('songs.csv')

)

REJECT LIMIT UNLIMITED;

SELECT \* FROM song\_test;

# External CSV File Example No. 2 (using regexp)

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

-- 1. First physically create the directory c:\temp

-- 2. Then login as system and do:

-- A. CREATE OR REPLACE DIRECTORY TMP AS 'C:\temp';

CREATE OR REPLACE DIRECTORY tmp AS 'C:\temp';

-- B. GRANT READ, WRITE ON DIRECTORY TMP TO CRUISES;

GRANT READ, WRITE ON DIRECTORY tmp TO samplecode;

DROP TABLE song\_ext;

CREATE TABLE song\_ext

(

song\_id NUMBER,

artist\_id NUMBER,

title VARCHAR2(100),

writer VARCHAR2(100),

seconds NUMBER

);

DECLARE

F UTL\_FILE.FILE\_TYPE;

V\_SID VARCHAR2(100);

V\_AID VARCHAR2(100);

V\_SONG VARCHAR2(100);

V\_WRT VARCHAR2(75);

V\_SECS VARCHAR2(100);

V\_LINE VARCHAR2(1000);

BEGIN

F := UTL\_FILE.FOPEN ('TMP', 'songs.csv', 'R');

IF UTL\_FILE.IS\_OPEN(F) THEN

LOOP

BEGIN

UTL\_FILE.GET\_LINE(F, V\_LINE, 1000);

IF V\_LINE IS NULL THEN

EXIT;

END IF;

V\_SID := REGEXP\_SUBSTR(V\_LINE, '[^,]+', 1, 1);

V\_AID := REGEXP\_SUBSTR(V\_LINE, '[^,]+', 1, 2);

V\_SONG := REGEXP\_SUBSTR(V\_LINE, '[^,]+', 1, 3);

V\_WRT := REGEXP\_SUBSTR(V\_LINE, '[^,]+', 1, 4);

V\_SECS := REGEXP\_SUBSTR(V\_LINE, '[^,]+', 1, 5);

INSERT INTO song\_ext VALUES(to\_number(V\_SID), to\_number(V\_AID), V\_SONG, V\_WRT,to\_number(V\_SECS));

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

EXIT;

END;

END LOOP;

END IF;

UTL\_FILE.FCLOSE(F);

END;

/

SELECT \* FROM song\_ext;

# External Restful Webservices Pipe Delimited File

## Embedded java

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

-- 1. First grant permission to user DASHBOARD

EXECUTE DBMS\_JAVA.GRANT\_PERMISSION( 'DASHBOARD', 'SYS:java.net.SocketPermission', '171.70.67.174:80', 'connect,resolve' );

-- 2. Next create JAVA SOURCE

DROP JAVA SOURCE DASHBOARD.GET\_EDCS\_REAL\_TIME;

CREATE OR REPLACE AND RESOLVE JAVA SOURCE NAMED DASHBOARD.GET\_EDCS\_REAL\_TIME as import java.io.IOException;

import java.io.IOException;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.MalformedURLException;

import java.net.URL;

import java.sql.\*;

public class GET\_EDCS\_REAL\_TIME {

public static void main(String[] args) {

int[] edcs\_1\_RecordNums = new int[0];

String[] edcs\_2\_EDCSNums = new String[0];

int[] edcs\_3\_Revs = new int[0];

String[] edcs\_4\_FileNames = new String[0];

String[] edcs\_5\_Titles = new String[0];

String[] edcs\_6\_Types = new String[0];

String[] edcs\_7\_Status = new String[0];

String[] edcs\_8\_StatusDate = new String[0];

String[] edcs\_9\_AppDate = new String[0];

String[] edcs\_10\_AppBy = new String[0];

String thisConnectionStr =

"jdbc:oracle:thin:@66.187.221.77:1526:gatepwhs";

String thisDBusr = "dashboard";

String thisDBpw = "dwetl1sg00d";

int myCount = 0;

int newSize = 0;

Connection conn = null;

String thisEDCS = null;

String thisEDCSs = null;

thisEDCS = args[0];

//thisEDCS = "811064";

thisEDCSs = thisEDCS;

thisEDCS = "EDCS-" + thisEDCS + ":\*";

URL url = null;

// System.out.println("setting the URL");

BufferedReader in = null;

String myurl = "http://wwwin-eng.cisco.com/cgi-

bin/edcs/getedcs.cgi?SEARCHTYPE=DOC&KEY=DOCNO&VALUE="+thisEDCS;

try {

url = new URL(myurl);

} catch (MalformedURLException e1) {

e1.printStackTrace();

}

try {

in = new BufferedReader(

new InputStreamReader(

url.openStream()));

} catch (IOException e2) {

e2.printStackTrace();

}

String replyString;

// System.out.println("got the URL");

try {

while((replyString = in.readLine()) != null){

// this\_sList\_array = replyString.split("\\|");

if (myCount == edcs\_1\_RecordNums.length){

// Arrays are full. Make new, bigger arrays,

newSize++; // Size of new array.

// 1 irecNum ---------------------

// get and set from Loop control variable

int[] temp1 = new int[newSize]; // The new array.

System.arraycopy(edcs\_1\_RecordNums, 0, temp1, 0,

edcs\_1\_RecordNums.length);

edcs\_1\_RecordNums = temp1; // Set old array to refer to new array.

edcs\_1\_RecordNums[myCount] = myCount;

// 2 ilistEDCSWatchNums ---------------------

// get and set from parameter

String[] temp2 = new String[newSize]; // The new array.

System.arraycopy(edcs\_2\_EDCSNums, 0, temp2, 0,

edcs\_2\_EDCSNums.length);

edcs\_2\_EDCSNums = temp2; // Set old array to refer to new array.

edcs\_2\_EDCSNums[myCount] = thisEDCSs;

// 3 rev number ---------------------

// get

int iRevIdxStart = replyString.indexOf("REV=");

int iRevIdxStop = replyString.indexOf("|FILESIZE=");

String sthisRev = replyString.substring(iRevIdxStart+4, iRevIdxStop);

int thisRev = Integer.parseInt(sthisRev);

// set

int[] temp3 = new int[newSize]; // The new array.

System.arraycopy(edcs\_3\_Revs, 0, temp3, 0, edcs\_3\_Revs.length);

edcs\_3\_Revs = temp3; // Set old array to refer to new array.

edcs\_3\_Revs[myCount] = thisRev;

// --------------------------------------------

// add from here down

// 4 File Name ---------------------

// get

int iFNIdxStart = replyString.indexOf("FILENAME=");

int iFNIdxStop = replyString.indexOf("|DOCNO=");

String thisFileName = replyString.substring(iFNIdxStart+9,

iFNIdxStop);

// set

String[] temp4 = new String[newSize]; // The new array.

System.arraycopy(edcs\_4\_FileNames, 0, temp4, 0,

edcs\_4\_FileNames.length);

edcs\_4\_FileNames = temp4; // Set old array to refer to new array.

edcs\_4\_FileNames[myCount] = thisFileName;

// 5 Title ---------------------

// get

int iTITLEIdxStart = replyString.indexOf("TITLE=");

int iTITLEIdxStop = replyString.indexOf("|STATUS=");

String thisFileTitle = replyString.substring(iTITLEIdxStart+6,

iTITLEIdxStop);

//set

String[] temp5 = new String[newSize]; // The new array.

System.arraycopy(edcs\_5\_Titles, 0, temp5, 0, edcs\_5\_Titles.length);

edcs\_5\_Titles = temp5; // Set old array to refer to new array.

edcs\_5\_Titles[myCount] = thisFileTitle;

// 6 Type ---------------------

// get

// public String[] edcs\_5\_Types = new String[0];

int iTYPEIdxStart = replyString.indexOf("FORMAT=");

int iTYPEIdxStop = replyString.indexOf("|AUTHOR=");

String thisFileType = replyString.substring(iTYPEIdxStart+7,

iTYPEIdxStop);

//set

String[] temp6 = new String[newSize]; // The new array.

System.arraycopy(edcs\_6\_Types, 0, temp6, 0, edcs\_6\_Types.length);

edcs\_6\_Types = temp6; // Set old array to refer to new array.

edcs\_6\_Types[myCount] = thisFileType;

// 7 Status ---------------------

// get

int iSTATIdxStart = replyString.indexOf("STATUS=");

int iSTATIdxStop = replyString.indexOf("|REV=");

String thisStatus = replyString.substring(iSTATIdxStart+7,

iSTATIdxStop);

// set

String[] temp7 = new String[newSize]; // The new array.

System.arraycopy(edcs\_7\_Status, 0, temp7, 0, edcs\_7\_Status.length);

edcs\_7\_Status = temp7; // Set old array to refer to new array.

edcs\_7\_Status[myCount] = thisStatus;

// 8 Status Date ---------------------

// get

int iSDATEIdxStart = replyString.indexOf("DATE=");

int iSDATEIdxStop = replyString.indexOf("|FORMAT");

String thisStatusDate = replyString.substring(iSDATEIdxStart+5,

iSDATEIdxStop);

// set

String[] temp8 = new String[newSize]; // The new array.

System.arraycopy(edcs\_8\_StatusDate, 0, temp8, 0,

edcs\_8\_StatusDate.length);

edcs\_8\_StatusDate = temp8; // Set old array to refer to new array.

edcs\_8\_StatusDate[myCount] = thisStatusDate;

// 9 Approved Date ---------------------

int iADATEIdxStart = replyString.indexOf("APPROVE\_DATE=");

int iADATEIdxStop = replyString.indexOf("|APPROVER");

String thisAppDate = replyString.substring(iADATEIdxStart+13,

iADATEIdxStop);

// set

String[] temp9 = new String[newSize]; // The new array.

System.arraycopy(edcs\_9\_AppDate, 0, temp9, 0, edcs\_9\_AppDate.length);

edcs\_9\_AppDate = temp9; // Set old array to refer to new array.

edcs\_9\_AppDate[myCount] = thisAppDate;

// 10 Approved by ---------------------

// get

// public String[] edcs\_9\_AppBy = new String[0];

// 10

int iABYIdxStart = replyString.indexOf("APPROVER=");

int iABYIdxStop = replyString.length();

String thisAppBy = replyString.substring(iABYIdxStart+9, iABYIdxStop);

// set

String[] temp10 = new String[newSize]; // The new array.

System.arraycopy(edcs\_10\_AppBy, 0, temp10, 0, edcs\_10\_AppBy.length);

edcs\_10\_AppBy = temp10; // Set old array to refer to new array.

edcs\_10\_AppBy[myCount] = thisAppBy;

myCount++;

} // if

} // while

} catch (IOException e3) {

e3.printStackTrace();

}

try {

in.close();

} catch (IOException e4) {

e4.printStackTrace();

}

// update database

// load the Oracle JDBC driver

try {

DriverManager.registerDriver(new

oracle.jdbc.driver.OracleDriver());

} catch (SQLException e5) {

e5.printStackTrace();

}

try { conn = DriverManager.getConnection(thisConnectionStr,

thisDBusr, thisDBpw);

} catch (SQLException e6) {

e6.printStackTrace();

}

java.sql.Date sqlDate1 = null;

java.sql.Date sqlDate2 = null;

PreparedStatement pstmt = null;

for (int i = 0 ; i < edcs\_1\_RecordNums.length; i++)

{

String sSQL = "insert into edcs\_test3 ( " +

" record\_number " +

" , edcs\_number " +

" , rev " +

" , file\_name " +

" , file\_title " +

" , file\_type " +

" , status " +

" , status\_date " +

" , approved\_date " +

" , app\_by " +

" ) " +

" values (?,?,?, ?,?,?, ? " +

", to\_date( ? , 'MM/DD/YYYY') " +

", to\_date( ? , 'MM/DD/YYYY') " +

", ? " +

" ) ";

// set up statDate

String date2 = null;

if (edcs\_9\_AppDate[i] != null && !edcs\_9\_AppDate[i].equals(""))

{

date2 = edcs\_9\_AppDate[i];

}

else {

date2 = "";

}

try {

pstmt = conn.prepareStatement(sSQL);

} catch (Exception e9) {

e9.printStackTrace();

}

try {

pstmt.setInt(1, edcs\_1\_RecordNums[i]);

pstmt.setString(2, edcs\_2\_EDCSNums[i]);

pstmt.setInt(3, edcs\_3\_Revs[i]);

pstmt.setString(4,edcs\_4\_FileNames[i]);

pstmt.setString(5,edcs\_5\_Titles[i]);

pstmt.setString(6,edcs\_6\_Types[i]);

pstmt.setString(7,edcs\_7\_Status[i]);

pstmt.setString(8,edcs\_8\_StatusDate[i]);

pstmt.setString(9,date2);

// set appBy

if (edcs\_10\_AppBy[i] != null && !edcs\_10\_AppBy[i].equals(""))

pstmt.setString(10,edcs\_10\_AppBy[i]);

else

pstmt.setString(10,"");

} catch (SQLException e10) {

System.out.println(" e10 exception");

e10.printStackTrace();

}

try {

pstmt.executeUpdate();

} catch (SQLException e11) {

System.out.println(" e11 exception");

System.out.println(e11.getMessage());

e11.printStackTrace();

}

try {

pstmt.close();

} catch (Exception e12) {

System.out.println(" e11 exception");

System.out.println(e12.getMessage());

e12.printStackTrace();

}

}

System.out.println("main is done");

}//main

}

/

## Create DB Stored Procedure

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

-- 3. Next create stored procedure

CREATE OR REPLACE PROCEDURE DASHBOARD.EDCSSP\_REALTIME(

id IN VARCHAR2)

AS

LANGUAGE JAVA NAME 'GET\_EDCS\_REAL\_TIME.main(java.lang.String[])';

/

## Call Java DB Stored Procedure

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

CALL DASHBOARD.EDCSSP\_REALTIME ('824098');

# Triggers for SCD Type 2 Data Model

-- <http://en.wikipedia.org/wiki/Slowly_changing_dimension>

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

DROP TRIGGER KM\_TRACKER.IC\_ARTIFACT\_HISTORY\_TRIGR;

CREATE OR REPLACE TRIGGER KM\_TRACKER."IC\_ARTIFACT\_HISTORY\_TRIGR" AFTER

INSERT OR

UPDATE OR

DELETE ON ic\_artifact FOR EACH row DECLARE

history\_action VARCHAR2(15 byte);

BEGIN

IF inserting THEN

history\_action := 'INSERTING';

elsif updating THEN

history\_action := 'UPDATING';

ELSE

history\_action := 'DELETING';

END IF;

IF inserting OR updating THEN

INSERT

INTO ic\_artifact\_history

(

entered\_by ,

entered\_date ,

history\_action ,

ic\_key ,

watch\_flag ,

last\_modified\_by ,

last\_modified\_date

)

VALUES

(

NVL(v('APP\_USER'),USER) ,

sysdate ,

history\_action ,

:new.ic\_key ,

:new.watch\_flag ,

:new.last\_modified\_by ,

:new.last\_modified\_date

);

ELSE

-- deleting

INSERT

INTO ic\_artifact\_history

(

entered\_by ,

entered\_date ,

history\_action ,

ic\_key ,

watch\_flag ,

last\_modified\_by ,

last\_modified\_date

)

VALUES

(

NVL(v('APP\_USER'),USER) ,

sysdate ,

history\_action ,

:old.ic\_key ,

:old.watch\_flag ,

:old.last\_modified\_by ,

:old.last\_modified\_date

);

END IF;

END;

/

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

DROP TRIGGER KM\_TRACKER.IC\_ARTIFACT\_KEY\_HISTORY\_TRIGR;

CREATE OR REPLACE TRIGGER KM\_TRACKER."IC\_ARTIFACT\_KEY\_HISTORY\_TRIGR" before

INSERT ON ic\_artifact\_history REFERENCING NEW AS NEW FOR EACH row BEGIN

SELECT ic\_artifact\_hist\_key\_seq.nextval INTO :new.pk\_key FROM dual;

END;

/

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

DROP TRIGGER KM\_TRACKER.IC\_ARTIFACT\_MOD\_TRIGR;

CREATE OR REPLACE TRIGGER KM\_TRACKER."IC\_ARTIFACT\_MOD\_TRIGR" before

INSERT OR

UPDATE ON ic\_artifact FOR EACH row BEGIN IF inserting THEN

SELECT 'IC-' || TO\_CHAR(ic\_artifact\_seq.nextval) INTO :new.ic\_key FROM dual;

:new.last\_modified\_by := NVL(v('APP\_USER'),USER);

:new.last\_modified\_date := sysdate;

elsif updating THEN

:new.last\_modified\_by := NVL(v('APP\_USER'),USER);

:new.last\_modified\_date := sysdate;

END IF;

END;

/

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

DROP TRIGGER KM\_TRACKER.IC\_HISTORY\_KEY\_TRIGR;

CREATE OR REPLACE TRIGGER KM\_TRACKER."IC\_HISTORY\_KEY\_TRIGR" before

INSERT ON ic\_history REFERENCING NEW AS NEW FOR EACH row BEGIN

SELECT ic\_history\_key\_seq.nextval INTO :new.pk\_key FROM dual;

END;

/

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

DROP TRIGGER KM\_TRACKER.IC\_HISTORY\_TRIGR;

CREATE OR REPLACE TRIGGER KM\_TRACKER."IC\_HISTORY\_TRIGR" AFTER

INSERT OR

UPDATE OR

DELETE ON ic FOR EACH row DECLARE history\_action VARCHAR2(25 byte);

BEGIN

IF inserting THEN

history\_action := 'INSERTING';

elsif updating THEN

history\_action := 'UPDATING';

ELSE

history\_action := 'DELETING';

END IF;

IF inserting OR updating THEN

INSERT

INTO ic\_history

(

entered\_by ,

entered\_date ,

history\_action ,

ic\_key ,

src\_version ,

src\_origin ,

src\_title ,

team ,

sme ,

src\_technology\_type ,

src\_pass\_through\_flag ,

src\_status ,

src\_status\_date ,

src\_project\_name ,

src\_underlying\_technology ,

src\_components ,

src\_edcs\_doc\_num ,

src\_edcs\_approved\_date ,

src\_edcs\_error ,

src\_remedy\_ticket\_num ,

src\_url ,

src\_space\_name ,

src\_path ,

src\_file\_name ,

src\_file\_type ,

src\_location ,

src\_date\_captured ,

src\_3rd\_party\_origin\_flag ,

src\_3rd\_party\_company ,

src\_referenced\_doc ,

src\_derived\_from ,

src\_related\_doc ,

src\_3rdparty\_content\_flag ,

src\_3rdparty\_content\_details ,

src\_remediation\_unique\_id ,

src\_st\_project\_name ,

src\_st\_views ,

src\_st\_label ,

src\_exp\_date ,

wf\_status ,

wf\_status\_date ,

wf\_owner ,

wf\_issue\_flag ,

wf\_comment ,

fg\_title ,

fg\_file\_name ,

fg\_file\_type ,

fg\_edcs\_doc\_num ,

fg\_edcs\_rev ,

fg\_edcs\_status ,

fg\_edcs\_status\_date ,

fg\_edcs\_approved\_date ,

fg\_edcs\_error ,

fg\_delivered\_technology ,

fg\_3rd\_party\_content\_flag ,

fg\_3rd\_party\_content\_details ,

fg\_referenced\_docs ,

fg\_derived\_from ,

fg\_related\_docs ,

fg\_release ,

fg\_watch\_flag ,

rts\_location ,

rts\_file\_name ,

comments ,

misc\_1 ,

misc\_2 ,

misc\_3 ,

last\_modified\_by ,

last\_modified\_date ,

src\_tag1 ,

src\_tag2 ,

wf\_target\_release ,

sme\_email ,

src\_wf\_owner ,

src\_wf\_id ,

dsgn\_src\_form

)

VALUES

(

NVL(v('APP\_USER'),USER) ,

sysdate ,

history\_action ,

:new.ic\_key ,

:new.src\_version ,

:new.src\_origin ,

:new.src\_title ,

:new.team ,

:new.sme ,

:new.src\_technology\_type ,

:new.src\_pass\_through\_flag ,

:new.src\_status ,

:new.src\_status\_date ,

:new.src\_project\_name ,

:new.src\_underlying\_technology ,

:new.src\_components ,

:new.src\_edcs\_doc\_num ,

:new.src\_edcs\_approved\_date ,

:new.src\_edcs\_error ,

:new.src\_remedy\_ticket\_num ,

:new.src\_url ,

:new.src\_space\_name ,

:new.src\_path ,

:new.src\_file\_name ,

:new.src\_file\_type ,

:new.src\_location ,

:new.src\_date\_captured ,

:new.src\_3rd\_party\_origin\_flag ,

:new.src\_3rd\_party\_company ,

:new.src\_referenced\_doc ,

:new.src\_derived\_from ,

:new.src\_related\_doc ,

:new.src\_3rdparty\_content\_flag ,

:new.src\_3rdparty\_content\_details ,

:new.src\_remediation\_unique\_id ,

:new.src\_st\_project\_name ,

:new.src\_st\_views ,

:new.src\_st\_label ,

:new.src\_exp\_date ,

:new.wf\_status ,

:new.wf\_status\_date ,

:new.wf\_owner ,

:new.wf\_issue\_flag ,

:new.wf\_comment ,

:new.fg\_title ,

:new.fg\_file\_name ,

:new.fg\_file\_type ,

:new.fg\_edcs\_doc\_num ,

:new.fg\_edcs\_rev ,

:new.fg\_edcs\_status ,

:new.fg\_edcs\_status\_date ,

:new.fg\_edcs\_approved\_date ,

:new.fg\_edcs\_error ,

:new.fg\_delivered\_technology ,

:new.fg\_3rd\_party\_content\_flag ,

:new.fg\_3rd\_party\_content\_details ,

:new.fg\_referenced\_docs ,

:new.fg\_derived\_from ,

:new.fg\_related\_docs ,

:new.fg\_release ,

:new.fg\_watch\_flag ,

:new.rts\_location ,

:new.rts\_file\_name ,

:new.comments ,

:new.misc\_1 ,

:new.misc\_2 ,

:new.misc\_3 ,

:new.last\_modified\_by ,

:new.last\_modified\_date ,

:new.src\_tag1 ,

:new.src\_tag2 ,

:new.wf\_target\_release ,

:new.sme\_email ,

:new.src\_wf\_owner ,

:new.src\_wf\_id ,

:new.dsgn\_src\_form

);

ELSE

-- deleting

INSERT

INTO ic\_history

(

entered\_by ,

entered\_date ,

history\_action ,

ic\_key ,

src\_version ,

src\_origin ,

src\_title ,

team ,

sme ,

src\_technology\_type ,

src\_pass\_through\_flag ,

src\_status ,

src\_status\_date ,

src\_project\_name ,

src\_underlying\_technology ,

src\_components ,

src\_edcs\_doc\_num ,

src\_edcs\_approved\_date ,

src\_edcs\_error ,

src\_remedy\_ticket\_num ,

src\_url ,

src\_space\_name ,

src\_path ,

src\_file\_name ,

src\_file\_type ,

src\_location ,

src\_date\_captured ,

src\_3rd\_party\_origin\_flag ,

src\_3rd\_party\_company ,

src\_referenced\_doc ,

src\_derived\_from ,

src\_related\_doc ,

src\_3rdparty\_content\_flag ,

src\_3rdparty\_content\_details ,

src\_remediation\_unique\_id ,

src\_st\_project\_name ,

src\_st\_views ,

src\_st\_label ,

src\_exp\_date ,

wf\_status ,

wf\_status\_date ,

wf\_owner ,

wf\_issue\_flag ,

wf\_comment ,

fg\_title ,

fg\_file\_name ,

fg\_file\_type ,

fg\_edcs\_doc\_num ,

fg\_edcs\_rev ,

fg\_edcs\_status ,

fg\_edcs\_status\_date ,

fg\_edcs\_approved\_date ,

fg\_edcs\_error ,

fg\_delivered\_technology ,

fg\_3rd\_party\_content\_flag ,

fg\_3rd\_party\_content\_details ,

fg\_referenced\_docs ,

fg\_derived\_from ,

fg\_related\_docs ,

fg\_release ,

fg\_watch\_flag ,

rts\_location ,

rts\_file\_name ,

comments ,

misc\_1 ,

misc\_2 ,

misc\_3 ,

last\_modified\_by ,

last\_modified\_date ,

src\_tag1 ,

src\_tag2 ,

wf\_target\_release ,

sme\_email ,

src\_wf\_owner ,

src\_wf\_id ,

dsgn\_src\_form

)

VALUES

(

NVL(v('APP\_USER'),USER) ,

sysdate ,

history\_action ,

:old.ic\_key ,

:old.src\_version ,

:old.src\_origin ,

:old.src\_title ,

:old.team ,

:old.sme ,

:old.src\_technology\_type ,

:old.src\_pass\_through\_flag ,

:old.src\_status ,

:old.src\_status\_date ,

:old.src\_project\_name ,

:old.src\_underlying\_technology ,

:old.src\_components ,

:old.src\_edcs\_doc\_num ,

:old.src\_edcs\_approved\_date ,

:old.src\_edcs\_error ,

:old.src\_remedy\_ticket\_num ,

:old.src\_url ,

:old.src\_space\_name ,

:old.src\_path ,

:old.src\_file\_name ,

:old.src\_file\_type ,

:old.src\_location ,

:old.src\_date\_captured ,

:old.src\_3rd\_party\_origin\_flag ,

:old.src\_3rd\_party\_company ,

:old.src\_referenced\_doc ,

:old.src\_derived\_from ,

:old.src\_related\_doc ,

:old.src\_3rdparty\_content\_flag ,

:old.src\_3rdparty\_content\_details ,

:old.src\_remediation\_unique\_id ,

:old.src\_st\_project\_name ,

:old.src\_st\_views ,

:old.src\_st\_label ,

:old.src\_exp\_date ,

:old.wf\_status ,

:old.wf\_status\_date ,

:old.wf\_owner ,

:old.wf\_issue\_flag ,

:old.wf\_comment ,

:old.fg\_title ,

:old.fg\_file\_name ,

:old.fg\_file\_type ,

:old.fg\_edcs\_doc\_num ,

:old.fg\_edcs\_rev ,

:old.fg\_edcs\_status ,

:old.fg\_edcs\_status\_date ,

:old.fg\_edcs\_approved\_date ,

:old.fg\_edcs\_error ,

:old.fg\_delivered\_technology ,

:old.fg\_3rd\_party\_content\_flag ,

:old.fg\_3rd\_party\_content\_details ,

:old.fg\_referenced\_docs ,

:old.fg\_derived\_from ,

:old.fg\_related\_docs ,

:old.fg\_release ,

:old.fg\_watch\_flag ,

:old.rts\_location ,

:old.rts\_file\_name ,

:old.comments ,

:old.misc\_1 ,

:old.misc\_2 ,

:old.misc\_3 ,

:old.last\_modified\_by ,

:old.last\_modified\_date ,

:old.src\_tag1 ,

:old.src\_tag2 ,

:old.wf\_target\_release ,

:old.sme\_email ,

:old.src\_wf\_owner ,

:old.src\_wf\_id ,

:old.dsgn\_src\_form

);

END IF;

END;

/

DROP TRIGGER KM\_TRACKER.IC\_MOD\_TRIGR;

CREATE OR REPLACE TRIGGER KM\_TRACKER."IC\_MOD\_TRIGR" before

INSERT OR

UPDATE ON ic FOR EACH row BEGIN

IF inserting THEN

:new.last\_modified\_by := NVL(v('APP\_USER'),USER);

:new.last\_modified\_date := sysdate;

ELSIF updating THEN

:new.last\_modified\_by := NVL(v('APP\_USER'),USER);

:new.last\_modified\_date := sysdate;

END IF;

END;

/

# Package Example

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

CREATE OR REPLACE PROCECURE

PACKAGE BODY DASHBOARD.metric\_rollups\_dtv

AS

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

-- Calls and Records Other Procedures

PROCEDURE driver\_dtv(

i\_start\_date IN DATE DEFAULT SYSDATE )

IS

v\_dtv\_dual\_date DATE;

v\_dtv\_connect\_status VARCHAR2 (15);

v\_msg VARCHAR2 (1000);

v\_im\_stage\_dtv\_last\_ran\_date DATE;

v\_im\_stage\_dtv\_last\_ran\_status VARCHAR2 (15);

v\_im\_hist\_dtv\_last\_ran\_date DATE;

v\_im\_hist\_dtv\_last\_ran\_status VARCHAR2 (15);

v\_im\_stage\_status\_dtv VARCHAR2 (15);

v\_im\_hist\_status\_dtv VARCHAR2 (15);

v\_im\_rollup\_status\_dtv VARCHAR2 (15);

v\_im\_ru\_dtv\_last\_ran\_dateDATE;

v\_im\_ru\_dtv\_last\_ran\_status VARCHAR2 (15);

v\_pm\_pull\_last\_ran DATE;

v\_cm\_pull\_last\_ran DATE;

v\_pm\_ru\_last\_ran DATE;

v\_cm\_ru\_last\_ran DATE;

v\_im\_status VARCHAR2 (15);

v\_pm\_status VARCHAR2 (15);

v\_cm\_status VARCHAR2 (15);

-- Get all period types in range of dates for rollups

## Simple Cursor

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

CURSOR period\_cur ( i\_last\_ran\_date IN DATE )

IS

SELECT period\_id,

period\_name,

period\_type,

start\_date,

end\_date

FROM rdr\_period

WHERE end\_date >= TO\_DATE ( TO\_CHAR (TRUNC (i\_last\_ran\_date - 1), 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' )

AND end\_date <= TO\_DATE ( TO\_CHAR (TRUNC (SYSDATE - 1), 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' );

BEGIN

metric\_utilities.log\_writer ('ALL', 'DTV METRIC\_ROLLUP DRIVER START', SYSDATE, NULL);

-- Get last run date get\_im\_stage\_dtv

metric\_utilities.determine\_last\_ran ('IM - DTV Load Stage', 'M', v\_im\_stage\_dtv\_last\_ran\_date, v\_im\_stage\_dtv\_last\_ran\_status);

v\_im\_stage\_dtv\_last\_ran\_date := v\_im\_stage\_dtv\_last\_ran\_date + 1 / 86400;

metric\_utilities.determine\_last\_ran ('IM - DTV Load History', 'M', v\_im\_hist\_dtv\_last\_ran\_date, v\_im\_hist\_dtv\_last\_ran\_status);

v\_im\_hist\_dtv\_last\_ran\_date := v\_im\_hist\_dtv\_last\_ran\_date + 1 / 86400;

metric\_utilities.determine\_last\_ran ('IM - DTV All Period Rollups', 'M', v\_im\_ru\_dtv\_last\_ran\_date, v\_im\_ru\_dtv\_last\_ran\_status);

v\_im\_ru\_dtv\_last\_ran\_date := v\_im\_ru\_dtv\_last\_ran\_date + 1 / 86400;

- metric\_utilities.determine\_last\_ran ('PM - DTV Pull', 'M', v\_pm\_pull\_last\_ran, v\_pm\_status);

metric\_utilities.determine\_last\_ran ('PM - DTV Rollup', 'M', v\_pm\_ru\_last\_ran, v\_pm\_status);

metric\_utilities.determine\_last\_ran ('CM - DTV Pull', 'M', v\_cm\_pull\_last\_ran, v\_cm\_status);

v\_cm\_pull\_last\_ran := v\_cm\_pull\_last\_ran + 1 / 86400;

metric\_utilities.determine\_last\_ran ('CM - DTV Rollup', 'M', v\_cm\_ru\_last\_ran, v\_cm\_status);

v\_cm\_ru\_last\_ran := v\_cm\_ru\_last\_ran + 1 / 86400;

-- Check DTV link

BEGIN

SELECT SYSDATE INTO v\_dtv\_dual\_date FROM DUAL@DTVREMP1.CISCO.COM hd;

-- handles remaining exceptions not handled by Named System Exceptions or Ueser Exceptions

EXCEPTION

WHEN OTHERS THEN

v\_dtv\_dual\_date := NULL;

v\_msg := SQLERRM;

metric\_utilities.log\_writer ( 'ALL', 'DTV METRIC\_ROLLUP DRIVER ERROR - dtv link validation', SYSDATE, v\_msg );

END;

IF v\_dtv\_dual\_date IS NOT NULL THEN

v\_dtv\_connect\_status := 'SUCCESS';

ELSE

v\_dtv\_connect\_status := 'FAILED';

-- Log the connection status

metric\_utilities.log\_writer ( 'ALL', 'METRIC\_ROLLUP DTV Connection status '

|| v\_dtv\_connect\_status, SYSDATE, NULL );

-- LOAD IM\_STAGE\_DATA

IF v\_dtv\_connect\_status = 'SUCCESS' THEN

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_from\_date,

calculation\_to\_date,

status

)

VALUES

(

'IM - DTV Load Stage',

SYSDATE,

NULL,

v\_im\_stage\_dtv\_last\_ran\_date,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

'STARTED'

);

ELSE

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_from\_date,

calculation\_to\_date,

status

)

VALUES

(

'IM - DTV Load Stage',

SYSDATE,

NULL,

v\_im\_stage\_dtv\_last\_ran\_date,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

'No Connection Available'

);

END IF;

-- If connection is good then load im\_stage\_data\_dtv

IF v\_dtv\_connect\_status = 'SUCCESS' THEN

im\_get\_data\_dtv ( v\_im\_stage\_dtv\_last\_ran\_date, TO\_DATE (TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY') || '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM'), v\_im\_stage\_status\_dtv );

metric\_utilities.log\_writer ('ALL', 'IM - DTV Load Stage Complete', SYSDATE, NULL);

ELSE

metric\_utilities.set\_current\_run\_stop ('IM - DTV Load Stage', v\_dtv\_connect\_status);

END IF;

metric\_utilities.set\_current\_run\_stop ('IM - DTV Load Stage', v\_im\_stage\_status\_dtv);

-- LOAD IM\_HISTORY\_DATA

If stage table loaded then update im\_history\_data

IF v\_im\_stage\_status\_dtv = 'SUCCESS' THEN

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_from\_date,

calculation\_to\_date,

status

)

VALUES

(

'IM - DTV Load History',

SYSDATE,

NULL,

v\_im\_hist\_dtv\_last\_ran\_date,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

'STARTED'

);

ELSE

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_from\_date,

calculation\_to\_date,

status

)

VALUES

(

'IM - DTV Load History',

SYSDATE,

NULL,

v\_im\_hist\_dtv\_last\_ran\_date,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

'Load Stage Failed'

);

END IF;

IF v\_im\_stage\_status\_dtv = 'SUCCESS' THEN

im\_update\_history\_dtv (v\_im\_hist\_status\_dtv);

metric\_utilities.log\_writer ('ALL', 'IM - DTV Load History Complete', SYSDATE, NULL);

ELSE

metric\_utilities.set\_current\_run\_stop ('IM - DTV Load History', v\_im\_hist\_status\_dtv);

END IF;

metric\_utilities.set\_current\_run\_stop ('IM - DTV Load History', v\_im\_hist\_status\_dtv);

-- DTV rollup: loop through and rollup for each period id since last time we ran

-- this assumes that the rollup has run at least once successfully

-- therefore the metric\_control table has been seeded with one record

IF v\_im\_hist\_status\_dtv = 'SUCCESS' THEN

-- Log the start of All Period Rollups Startting

metric\_utilities.log\_writer ('ALL', 'IM - DTV All Period Rollups Starting', SYSDATE, NULL);

-- Enter into metric\_control the starting of all Period Rollups

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_from\_date,

calculation\_to\_date,

status

)

VALUES

(

'IM - DTV All Period Rollups',

SYSDATE,

NULL,

v\_im\_ru\_dtv\_last\_ran\_date,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

'STARTED'

);

-- Then proceed to calculate each individual rollup

FOR period\_rec IN period\_cur

(

v\_im\_ru\_dtv\_last\_ran\_date

)

LOOP

-- Enter the start of the indivual DTV rollup

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_from\_date,

calculation\_to\_date,

status,

period\_id

)

VALUES

(

'IM - DTV Rollup',

SYSDATE,

NULL,

period\_rec.start\_date,

period\_rec.end\_date,

'STARTED',

period\_rec.period\_id

);

COMMIT;

-- run the actual indivual DTV rollup

im\_rollup\_data\_dtv (period\_rec.start\_date, period\_rec.end\_date, period\_rec.period\_id, period\_rec.period\_type, period\_rec.period\_name, v\_im\_rollup\_status\_dtv);

-- set the stop with the status returned

metric\_utilities.set\_current\_run\_stop ('IM - DTV Rollup', v\_im\_rollup\_status\_dtv);

metric\_utilities.log\_writer ( 'ALL', 'IM\_DTV:' || period\_rec.period\_type || ' is Complete - Period: ' || period\_rec.period\_id, SYSDATE, NULL );

END LOOP;

-- Log the end of all DTV Rollups

metric\_utilities.log\_writer ('ALL', 'IM - DTV All Period Rollups Complete', SYSDATE, NULL);

-- Set the metric\_control entry for all with the status of the last period status

metric\_utilities.set\_current\_run\_stop ( 'IM - DTV All Period Rollups', v\_im\_rollup\_status\_dtv );

ELSE

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_from\_date,

calculation\_to\_date,

status

)

VALUES

(

'IM - DTV All Period Rollups',

SYSDATE,

NULL,

v\_im\_ru\_dtv\_last\_ran\_date,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

'FAILED: No Hostory Update Success'

);

-- Log the failure all DTV Rollups

metric\_utilities.log\_writer ('ALL', 'METRIC\_ROLLUP\_DTV IM Failed', SYSDATE, NULL);

END IF;

metric\_utilities.log\_writer ('ALL', 'PM - DTV Pull Starting ' || v\_pm\_status, SYSDATE, NULL);

IF v\_pm\_status = 'SUCCESS' THEN

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_to\_date,

calculation\_from\_date,

status

)

VALUES

(

'PM - DTV Pull',

SYSDATE,

NULL,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

v\_pm\_pull\_last\_ran,

'STARTED'

);

COMMIT;

-- metric\_utilities.set\_current\_run\_start('PM', period\_rec.start\_date, period\_rec.period\_type, period\_rec.end\_date, v\_pm\_status);

ELSE

metric\_utilities.set\_current\_run\_stop ('PM - DTV Pull', v\_pm\_status);

END IF;

IF v\_pm\_status = 'SUCCESS' THEN

pm\_get\_data\_dtv ( v\_pm\_pull\_last\_ran, TO\_DATE (TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY') || '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM'), v\_pm\_status );

ELSE

metric\_utilities.set\_current\_run\_stop ('PM - DTV Pull', v\_pm\_status);

END IF;

metric\_utilities.log\_writer ( 'ALL', 'PM - DTV Update Starting ' || v\_pm\_status, SYSDATE, NULL );

IF v\_pm\_status = 'SUCCESS' THEN

pm\_update\_history\_dtv (v\_pm\_status);

ELSE

metric\_utilities.set\_current\_run\_stop ('PM - DTV Pull', v\_pm\_status);

END IF;

metric\_utilities.set\_current\_run\_stop ('PM - DTV Pull', v\_pm\_status);

--Check for successful determination of last run date AND THEN proceed

metric\_utilities.log\_writer ( 'ALL', 'PM - DTV Rollup Starting ' || v\_pm\_status, SYSDATE, NULL );

IF v\_pm\_status = 'SUCCESS' THEN

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_to\_date,

calculation\_from\_date,

status

)

VALUES

(

'PM - DTV Rollup',

SYSDATE,

NULL,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

v\_pm\_pull\_last\_ran,

'STARTED'

);

COMMIT;

pm\_rollup\_data\_dtv (v\_pm\_status);

ELSE

metric\_utilities.set\_current\_run\_stop ('PM - DTV Rollup', v\_pm\_status);

END IF;

metric\_utilities.set\_current\_run\_stop ('PM - DTV Rollup', v\_pm\_status);

--Do CM

--Check for successful determination of last run date AND THEN proceed

## Simple Logging Example

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

**metric\_utilities.log\_writer ( 'ALL', 'CM - DTV Get Data Starting ' || v\_cm\_status, SYSDATE, NULL );**

## Simple History of Status Example

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

**IF v\_cm\_status = 'SUCCESS' THEN**

**INSERT**

**INTO metric\_control**

**(**

**metric\_name,**

**run\_start\_date,**

**run\_end\_date,**

**calculation\_to\_date,**

**calculation\_from\_date,**

**status**

**)**

**VALUES**

**(**

**'CM - DTV Pull',**

**SYSDATE,**

**NULL,**

**TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')**

**|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),**

**v\_cm\_pull\_last\_ran,**

**'STARTED'**

**);**

**ELSE**

**metric\_utilities.set\_current\_run\_stop ('CM - DTV Pull', v\_cm\_status);**

**END IF;**

IF v\_cm\_status = 'SUCCESS' THEN

cm\_get\_data\_dtv ( v\_cm\_pull\_last\_ran, TO\_DATE (TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY') || '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM'), v\_cm\_status );

ELSE

metric\_utilities.set\_current\_run\_stop ('CM - DTV Pull', v\_cm\_status);

END IF;

metric\_utilities.log\_writer ( 'ALL', 'CM - DTV Update Starting ' || v\_cm\_status, SYSDATE, NULL );

IF v\_cm\_status = 'SUCCESS' THEN

cm\_update\_history\_dtv (v\_cm\_status);

ELSE

metric\_utilities.set\_current\_run\_stop ('CM - DTV Pull', v\_cm\_status);

END IF;

metric\_utilities.set\_current\_run\_stop ('CM - DTV Pull', v\_cm\_status);

metric\_utilities.log\_writer ( 'ALL', 'CM - DTV Rollup Starting ' || v\_cm\_status, SYSDATE, NULL );

IF v\_cm\_status = 'SUCCESS' THEN

INSERT

INTO metric\_control

(

metric\_name,

run\_start\_date,

run\_end\_date,

calculation\_to\_date,

calculation\_from\_date,

status

)

VALUES

(

'CM - DTV Rollup',

SYSDATE,

NULL,

TO\_DATE ( TO\_CHAR (SYSDATE - 1, 'MM/DD/YYYY')

|| '11:59:59 PM', 'MM/DD/YYYY HH:MI:SS AM' ),

v\_cm\_pull\_last\_ran,

'STARTED'

);

cm\_rollup\_data\_dtv (v\_cm\_status);

ELSE

metric\_utilities.set\_current\_run\_stop ('CM - DTV Rollup', v\_cm\_status);

END IF;

metric\_utilities.set\_current\_run\_stop ('CM - DTV Rollup', v\_cm\_status);

metric\_utilities.log\_writer ('ALL', 'METRIC\_ROLLUP DRIVER COMPLETE', SYSDATE, NULL);

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

metric\_utilities.log\_writer ('ALL', 'METRIC\_ROLLUP DRIVER ERROR', SYSDATE, v\_msg);

END driver\_dtv;

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

-- INCIDENT MANAGEMENT

-- Gets stage data FROM Incident Managment bASe tables

PROCEDURE im\_get\_data\_DTV

(

i\_start\_date IN DATE ,

i\_stop\_date IN DATE ,

o\_im\_status OUT VARCHAR

)

IS

-- DTV REMEDY (Source: 'DTVREMP1.CISCO.COM

v\_msg VARCHAR2(1000);

v\_lASt\_updated DATE;

-- autonotify flag

## Bulk Collect User Defined Type Example

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

**TYPE ticket\_list\_an**

**IS**

**TABLE OF VARCHAR2(15) INDEX BY BINARY\_INTEGER;**

**TYPE task\_count\_list**

**IS**

**TABLE OF NUMBER INDEX BY BINARY\_INTEGER;**

## Instantiate Bulk Collect User Defined Type Example

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

**my\_ticket\_list\_an ticket\_list\_an;**

**my\_task\_count\_list task\_count\_list;**

-- all others

-- 0. ticket\_list

TYPE ticket\_list

IS

TABLE OF VARCHAR2(15)INDEX BY BINARY\_INTEGER;

-- 1. company\_id

TYPE company\_list

IS

TABLE OF VARCHAR2(254)

INDEX BY BINARY\_INTEGER;

-- 2. auto\_notify\_flag

TYPE auto\_notify\_list

IS

TABLE OF VARCHAR2(1)

INDEX BY BINARY\_INTEGER;

-- 3. alarm\_to\_analyze

TYPE in\_progress\_date\_list

IS

TABLE OF DATE INDEX BY BINARY\_INTEGER;

TYPE reported\_date\_list

IS

TABLE OF DATE INDEX BY BINARY\_INTEGER;

-- 4. inc\_mttc

-- 6. met\_mttc sla

TYPE alarm\_to\_crt\_mins\_list

IS

TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

-- 5. inc\_mttn

-- 7. mttn\_sla\_met

TYPE tkt\_create\_to\_notify\_mins\_list

IS

TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

TYPE submitter\_list

IS

TABLE OF VARCHAR2 (254 BYTE)

INDEX BY BINARY\_INTEGER;

-- 8-11. met\_Px\_SLA

TYPE priority\_list

IS

TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

TYPE reported\_network\_condition\_tbl

IS

TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

TYPE tkt\_create\_to\_resolved\_hrs

IS

TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

-- create variables of that type

-- 0. Ticket\_list

my\_ticket\_list ticket\_list;

-- 1. company\_id

my\_company\_list company\_list;

-- 2. auto\_notify\_flag

my\_auto\_notify\_list auto\_notify\_list;

-- 3. alarm\_to\_analyze

my\_in\_progress\_date\_list in\_progress\_date\_list;

my\_reported\_date\_list reported\_date\_list;

-- 4. inc\_mttc

-- 6. met MTTC SLA

my\_alarm\_to\_crt\_mins\_list alarm\_to\_crt\_mins\_list;

-- 5. inc\_mttn

-- 7. mttn\_sla\_met

my\_tkt\_create\_to\_not\_mins\_list tkt\_create\_to\_notify\_mins\_list;

my\_submitter\_list submitter\_list;

-- 8-11. met\_Px\_SLA

my\_priority\_list priority\_list;

my\_reptd\_network\_cond\_list reported\_network\_condition\_tbl;

my\_tkt\_create\_to\_res\_hrs\_list tkt\_create\_to\_resolved\_hrs;

-- Create local variables

-- 1. company\_id

v\_company\_id VARCHAR2(15);

-- 2. auto\_notify\_flag

v\_auto\_notify\_flag VARCHAR2(1);

-- 3. alarm\_to\_analyze

v\_alarm\_to\_analyze\_mins NUMBER;

-- 4. inc\_mttc

v\_inc\_mttcreate VARCHAR2(1);

-- 6. met\_mttc\_sla

v\_mttcreate\_sla\_met VARCHAR2(1);

-- 5. met\_mttn

v\_inc\_mttn VARCHAR2(1);

-- 7. met\_mttn\_sla

v\_mttn\_sla\_met VARCHAR2(1);

-- 8-11. met\_Px\_SLA

v\_mttr\_p1\_sla\_met VARCHAR2(1);

v\_mttr\_p2\_sla\_met VARCHAR2(1);

v\_mttr\_p3\_sla\_met VARCHAR2(1);

v\_mttr\_p4\_sla\_met VARCHAR2(1);

reported\_net\_cond\_is\_blank BOOLEAN;

BEGIN

reported\_net\_cond\_is\_blank := FALSE;

v\_mttr\_p1\_sla\_met := 'N';

v\_mttr\_p2\_sla\_met := 'N';

v\_mttr\_p3\_sla\_met := 'N';

v\_lASt\_updated := SYSDATE;

-- Truncate our work tables

-- May have more than one entry for a ticket number in the work table

METRIC\_UTILITIES.TRUNC\_TABLE('dAShboard.im\_stage\_ids\_to\_pull\_work\_dtv');

-- Get distinct ticket numbers

METRIC\_UTILITIES.TRUNC\_TABLE('dAShboard.im\_stage\_data\_ids\_to\_pull\_dtv');

METRIC\_UTILITIES.TRUNC\_TABLE('dAShboard.im\_stage\_data\_dtv');

COMMIT;

-- Determine which incident numbers to pull

INSERT

INTO im\_stage\_ids\_to\_pull\_work\_dtv

(incident\_number)

## Simple Set Operations

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

**SELECT hd.incident\_number**

**FROM aradmin.hpd\_help\_desk@dtvremp1.cisco.com hd**

**WHERE metric\_utilities.convert\_to\_date(hd.lASt\_modified\_date) >**

**i\_start\_date**

**UNION**

**SELECT request\_id02**

**FROM aradmin.hpd\_ASsociations@dtvremp1.cisco.com ha**

**WHERE metric\_utilities.convert\_to\_date(ha.lASt\_modified\_date) >**

**i\_start\_date**

**UNION**

**SELECT incident\_number**

**FROM aradmin.hpd\_worklog@dtvremp1.cisco.com hw**

**WHERE metric\_utilities.convert\_to\_date(hw.lASt\_modified\_date) >**

**i\_start\_date;**

**COMMIT;**

INSERT INTO im\_stage\_data\_ids\_to\_pull\_dtv

(incident\_number)

SELECT DISTINCT incident\_number FROM im\_stage\_ids\_to\_pull\_work\_dtv;

COMMIT;

--Populate stage data

INSERT

INTO DASHBOARD.IM\_STAGE\_DATA\_DTV

(

incident\_number ,

alarm\_identifier ,

alarm\_type ,

ASsigned\_group ,

ASsigned\_support\_company ,

ASsigned\_support\_organization ,

ASsignee ,

ASsignee\_groups ,

city ,

closure\_product\_category\_tier1 ,

closure\_product\_category\_tier2 ,

closure\_product\_category\_tier3 ,

closure\_product\_name ,

company

-- set in the cursor below

--, companyid

,

correlated\_ci ,

correlated\_id ,

country ,

country\_code ,

created\_by ,

effort\_time\_spent\_minutes ,

first\_name ,

generic\_categorization\_tier\_1 ,

generic\_categorization\_tier\_2 ,

generic\_categorization\_tier\_3 ,

impact ,

impact\_rating ,

lASt\_modified\_by ,

lASt\_name ,

lookupkeyword ,

managed ,

manufacturer ,

mapped\_severity ,

mom\_identifier ,

mom\_type ,

organization ,

owner ,

owner\_group ,

owner\_group\_id ,

owner\_support\_company ,

owner\_support\_organization ,

partner ,

priority ,

product\_categorization\_tier\_1 ,

product\_categorization\_tier\_2 ,

product\_categorization\_tier\_3 ,

product\_model\_version ,

product\_name ,

recurrence ,

region ,

related\_alarm\_code ,

reported\_source ,

reportedci ,

resolution\_category ,

resolution\_category\_tier\_2 ,

resolution\_category\_tier\_3 ,

service\_type ,

severity ,

short\_description ,

site ,

site\_group ,

site\_id ,

state\_province ,

status ,

status\_reASon ,

submitter ,

total\_escalation\_level ,

total\_ola\_acknowledgeesc\_level ,

total\_ola\_resolution\_esc\_level ,

total\_time\_spent ,

total\_transfers ,

urgency

-- Atomic Dates

,

closed\_date ,

estimated\_resolution\_date ,

initial\_notification\_date ,

isolation\_date ,

lASt\_\_ASsigned\_date ,

lASt\_ackowledged\_date ,

lASt\_modified\_date ,

lASt\_occurence ,

lASt\_resolved\_date ,

reported\_date ,

responded\_date ,

submit\_date

-- Calculated Dates

,

in\_progress\_date ,

day\_closed ,

day\_opened ,

month\_closed ,

month\_opened ,

week\_closed ,

week\_opened

-- Calculated Flags

-- Set in the cursor below

-- , auto\_notify\_flag

,

auto\_tkt\_flag ,

closed\_AS\_duplicate\_flag ,

closed\_AS\_ros\_infrAStr\_flag ,

closed\_AS\_merged\_flag ,

sched\_maint\_flag

-- Calculated Counts

,

num\_cust\_escalations ,

num\_customer\_updates ,

num\_entitlement\_wil ,

num\_external\_escalations ,

num\_internal\_escalations ,

num\_touches

-- no off\_clock\_time\_mins possible with DTV Remedy

-- , off\_clock\_time\_mins

,

alarm\_to\_notify\_mins ,

alarm\_to\_isolate\_mins ,

alarm\_to\_resolve\_mins ,

alarm\_to\_close\_mins

-- Set in the cursor below

-- , alarm\_to\_analyze\_mins

,

alarm\_to\_create\_mins ,

Tkt\_Create\_To\_Closed\_Hrs ,

Tkt\_Create\_To\_Isolate\_Mins ,

Tkt\_Create\_To\_Notify\_Mins ,

Tkt\_Create\_To\_Resolved\_Hrs ,

Tkt\_Isolate\_To\_Resolved\_Mins ,

Tkt\_Resolved\_To\_Closed\_Mins ,

data\_source

-- Set in the cursor below

--, inc\_mttcreate

--, inc\_mttn

,

description ,

system\_created ,

lASt\_updated ,

reported\_network\_condition ,

notification\_required

)

SELECT

-- atomic data 1 x 1

hd.incident\_number ,

hd.alarm\_identifier ,

hd.alarm\_type ,

hd.ASsigned\_group ,

hd.ASsigned\_support\_company ,

hd.ASsigned\_support\_organization ,

hd.ASsignee ,

hd.ASsignee\_groups ,

hd.city ,

hd.closure\_product\_category\_tier1 ,

hd.closure\_product\_category\_tier2 ,

hd.closure\_product\_category\_tier3 ,

hd.closure\_product\_name ,

hd.company

--, hd.corporate\_id

,

hd.correlated\_ci ,

hd.correlated\_id ,

hd.country ,

hd.country\_code ,

hd.created\_by ,

hd.effort\_time\_spent\_minutes ,

hd.first\_name ,

hd.generic\_categorization\_tier\_1 ,

hd.generic\_categorization\_tier\_2 ,

hd.generic\_categorization\_tier\_3 ,

hd.impact ,

hd.impact\_rating ,

hd.lASt\_modified\_by ,

hd.lASt\_name ,

hd.lookupkeyword ,

hd.managed ,

hd.manufacturer ,

hd.mapped\_severity ,

hd.mom\_identifier ,

hd.mom\_type ,

hd.organization ,

hd.owner ,

hd.owner\_group ,

hd.owner\_group\_id ,

hd.owner\_support\_company ,

hd.owner\_support\_organization ,

hd.partner

-- Priority

-- Reported Network Condition Priority

-- 0 Network Event 3

-- 1 Degraded Service 2

-- 2 Outage1

,

## Simple Case Statement

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

CASE

WHEN hd.reported\_network\_condition = 0

THEN 3

WHEN hd.reported\_network\_condition = 1

THEN 2

WHEN hd.reported\_network\_condition = 2

THEN 1

WHEN hd.reported\_network\_condition IS NULL

THEN hd.priority

END AS priority ,

hd.product\_categorization\_tier\_1 ,

hd.product\_categorization\_tier\_2 ,

hd.product\_categorization\_tier\_3 ,

hd.product\_model\_version ,

hd.product\_name ,

hd.recurrence ,

hd.region ,

hd.related\_alarm\_code ,

hd.reported\_source ,

hd.reportedci ,

hd.resolution\_category ,

hd.resolution\_category\_tier\_2 ,

hd.resolution\_category\_tier\_3 ,

hd.service\_type ,

hd.severity ,

hd.short\_description ,

hd.site ,

hd.site\_group ,

hd.site\_id ,

hd.state\_province ,

## Simple Decode Statement

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

**DECODE (hd.status,**

**0, 'New',**

**1, 'ASsigned',**

**2, 'In Progress',**

**3, 'Pending',**

**4, 'Resolved',**

**5, 'Closed',**

**6, 'Cancelled') AS status ,**

hd.status\_reASon ,

hd.submitter ,

hd.total\_escalation\_level ,

hd.total\_ola\_acknowledgeesc\_level ,

hd.total\_ola\_resolution\_esc\_level ,

hd.total\_time\_spent ,

hd.total\_transfers ,

hd.urgency

-- Atomic Dates

,

metric\_utilities.convert\_to\_date(hd.closed\_date) ,

metric\_utilities.convert\_to\_date(hd.estimated\_resolution\_date) ,

metric\_utilities.convert\_to\_date(hd.initial\_notification\_date) ,

metric\_utilities.convert\_to\_date(hd.isolation\_date) ,

metric\_utilities.convert\_to\_date(hd.lASt\_\_ASsigned\_date) ,

metric\_utilities.convert\_to\_date(hd.lASt\_acknowledged\_date) ,

metric\_utilities.convert\_to\_date(hd.lASt\_modified\_date) ,

metric\_utilities.convert\_to\_date(hd.lASt\_occurrence) ,

metric\_utilities.convert\_to\_date(hd.lASt\_resolved\_date) ,

metric\_utilities.convert\_to\_date(hd.reported\_date) ,

metric\_utilities.convert\_to\_date(hd.responded\_date) ,

metric\_utilities.convert\_to\_date(hd.submit\_date)

-- Calculated Dates

-- NOTE: the cisco remedy system uses A\_CSC\_HPD\_HELPDESKSTATUSLOG

-- which creates a new record every time a ticket changes ststus

-- this table can used to determine earliest date a ticket went into a

-- particular status and how long it wAS in that status

-- DTV REMEDY does not have this table, instead we use the H1074 table

-- which holds a single record for each incident and just the most recent

-- time went into specific status T2 is time it enbters "In Progress"

,

(SELECT metric\_utilities.convert\_to\_date(h.t2)

## Access DBLink

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

**FROM aradmin.h1074@dtvremp1.cisco.com h**

WHERE h.entryid = hd.incident\_number

) in\_progress\_date

-- this is the original Cisco Remedy code left in for readability

--,( SELECT MIN(in\_progress\_date)+(v\_offset/24)

-- FROM rdr\_ticket rdrt

-- WHERE hd.incident\_number = rdrt.incident\_number

-- GROUP BY incident\_number) AS

--in\_progress\_date

,

TRUNC(metric\_utilities.convert\_to\_date(hd.closed\_date),'DD')

AS day\_closed ,

TRUNC(metric\_utilities.convert\_to\_date(hd.submit\_date),'DD')

AS day\_opened ,

TRUNC(metric\_utilities.convert\_to\_date(hd.closed\_date),'MM')

AS month\_closed ,

TRUNC(metric\_utilities.convert\_to\_date(hd.submit\_date),'MM')

AS month\_opened ,

TRUNC(metric\_utilities.convert\_to\_date(hd.closed\_date),'WW')

AS week\_closed ,

TRUNC(metric\_utilities.convert\_to\_date(hd.submit\_date),'WW')

AS week\_opened

-- Calculated Flags

--, tmp.auto\_notify auto\_notify\_flag

,

CASE

WHEN hd.submitter = 'WEBSERVICES'

THEN 'Y'

ELSE 'N'

END AS auto\_tkt\_flag ,

CASE

WHEN hd.status = 5

AND hd.resolution\_category\_tier\_2 = 'Duplicate Ticket'

THEN 'Y'

ELSE 'N'

END AS closed\_AS\_duplicate\_flag ,

CASE

WHEN hd.status = 5

AND hd.resolution\_category = 'ROS InfrAStructure'

THEN 'Y'

ELSE 'N'

END AS closed\_AS\_ros\_infrAStr\_flag ,

CASE

WHEN hd.status = 5

AND hd.status\_reASon = 1500

THEN 'Y'

ELSE 'N'

END AS closed\_AS\_merged\_flag ,

CASE

WHEN hd.resolution\_category\_tier\_2 = 'Scheduled Maint'

THEN 'Y'

ELSE 'N'

END AS sched\_maint\_flag

-- Calculated Counts

,

NVL(

(SELECT COUNT(hw.work\_log\_type) num\_cust\_escalations

FROM aradmin.hpd\_worklog@dtvremp1.cisco.com hw

WHERE hw.work\_log\_type IN (4201)

AND hd.incident\_number = hw.incident\_number

GROUP BY hw.incident\_number

),0) AS num\_cust\_escalations ,

NVL (

(SELECT COUNT(hw.work\_log\_type) num\_customer\_updates

FROM aradmin.hpd\_worklog@dtvremp1.cisco.com hw

WHERE hw.work\_log\_type IN (5105, 6000, 7000, 8000, 9000, 10000, 11000, 11001)

AND hd.incident\_number = hw.incident\_number

GROUP BY hw.incident\_number

),0) AS num\_customer\_updates ,

NVL(

(SELECT COUNT(hw.work\_log\_type) num\_entitlement\_wil

FROM aradmin.hpd\_worklog@dtvremp1.cisco.com hw

WHERE hw.work\_log\_type IN (18008)

AND hd.incident\_number = hw.incident\_number

GROUP BY hw.incident\_number

),0) AS num\_entitlement\_wil ,

NVL(

(SELECT COUNT(hw.work\_log\_type)

FROM aradmin.hpd\_worklog@dtvremp1.cisco.com hw

WHERE hw.work\_log\_type IN (18007)

AND hd.incident\_number = hw.incident\_number

GROUP BY hw.incident\_number

),0) AS num\_external\_escalations ,

NVL(

(SELECT COUNT(hw.work\_log\_type) num\_internal\_escalations

FROM aradmin.hpd\_worklog@dtvremp1.cisco.com hw

WHERE hw.work\_log\_type IN (18006)

AND hd.incident\_number = hw.incident\_number

GROUP BY hw.incident\_number

),0) AS num\_internal\_escalations ,

(SELECT COUNT(hw.work\_log\_type) total\_touches

FROM aradmin.hpd\_worklog@dtvremp1.cisco.com hw

WHERE hw.incident\_number = hd.incident\_number

GROUP BY hw.incident\_number

) AS num\_touches

,

(metric\_utilities.convert\_to\_date(hd.initial\_notification\_date) - metric\_utilities.convert\_to\_date(hd.reported\_date))\*(24\*60) alarm\_to\_notify\_mins

,

(metric\_utilities.convert\_to\_date(hd.isolation\_date) - metric\_utilities.convert\_to\_date(hd.reported\_date))\*(24\*60) alarm\_to\_isolate\_mins

,

CASE

WHEN hd.lASt\_resolved\_date IS NOT NULL

AND hd.reported\_date IS NOT NULL

AND hd.lASt\_resolved\_date > hd.reported\_date

AND hd.statusIN (4,5)

THEN (metric\_utilities.convert\_to\_date(hd.lASt\_resolved\_date) - metric\_utilities.convert\_to\_date(hd.reported\_date))\*(24\*60)

WHEN hd.lASt\_resolved\_date IS NULL

AND hd.closed\_date IS NOT NULL

AND hd.reported\_date IS NOT NULL

AND hd.closed\_date > hd.reported\_date

AND hd.status IN (4,5)

THEN (metric\_utilities.convert\_to\_date(hd.closed\_date) - metric\_utilities.convert\_to\_date(hd.reported\_date))\*(24\*60)

END AS alarm\_to\_resolve\_mins

,

(metric\_utilities.convert\_to\_date(hd.closed\_date) - metric\_utilities.convert\_to\_date(hd.reported\_date))\*(24\*60) alarm\_to\_close\_mins

,

(metric\_utilities.convert\_to\_date(hd.submit\_date) - metric\_utilities.convert\_to\_date(hd.reported\_date))\*(24\*60) alarm\_to\_create\_mins

,

(hd.Closed\_Date - hd.Submit\_Date) / (60 \* 60) AS Tkt\_Create\_To\_Closed\_Hrs ,

(hd.Isolation\_Date- hd.Submit\_Date) / (60)AS Tkt\_Create\_To\_Isolate\_Mins ,

(hd.Initial\_Notification\_Date - hd.Submit\_Date) / (60)AS

Tkt\_Create\_To\_Notify\_Mins

-- tkt\_create\_to\_resolved

,

CASE

WHEN hd.lASt\_resolved\_date IS NOT NULL

AND hd.submit\_date IS NOT NULL

AND hd.lASt\_resolved\_date > hd.submit\_date

AND hd.statusIN (4,5)

THEN (metric\_utilities.convert\_to\_date(hd.lASt\_resolved\_date) - metric\_utilities.convert\_to\_date(hd.submit\_date))\*(24)

WHEN hd.lASt\_resolved\_date IS NULL

AND hd.closed\_date IS NOT NULL

AND hd.submit\_date IS NOT NULL

AND hd.closed\_date > hd.submit\_date

AND hd.status IN (4,5)

THEN (metric\_utilities.convert\_to\_date(hd.closed\_date) - metric\_utilities.convert\_to\_date(hd.submit\_date))\*(24)

END AS tkt\_create\_to\_resolved\_hrs

-- tkt\_isolate\_to\_resolved

,

CASE

WHEN hd.lASt\_resolved\_date IS NOT NULL

AND hd.isolation\_dateIS NOT NULL

AND hd.lASt\_resolved\_date > hd.isolation\_date

AND hd.statusIN (4,5)

THEN (metric\_utilities.convert\_to\_date(hd.lASt\_resolved\_date) - metric\_utilities.convert\_to\_date(hd.isolation\_date))\*(24\*60)

WHEN hd.lASt\_resolved\_date IS NULL

AND hd.closed\_date IS NOT NULL

AND hd.isolation\_dateIS NOT NULL

AND hd.closed\_date > hd.isolation\_date

AND hd.status IN (4,5)

THEN (metric\_utilities.convert\_to\_date(hd.closed\_date) - metric\_utilities.convert\_to\_date(hd.isolation\_date))\*(24\*60)

END AS tkt\_isolate\_to\_resolved\_mins

-- tkt\_closed\_to\_resolved

,

CASE

WHEN hd.lASt\_resolved\_date IS NOT NULL

AND hd.closed\_date IS NOT NULL

AND hd.closed\_date > hd.lASt\_resolved\_date

AND hd.status IN (4,5)

THEN (metric\_utilities.convert\_to\_date(hd.closed\_date) - metric\_utilities.convert\_to\_date(hd.lASt\_resolved\_date))\*(24\*60)

END AS tkt\_resolved\_to\_closed\_mins ,

'DTVREMP1.CISCO.COM'

,

REPLACE(REPLACE(hd.description,chr(10),chr(32)),chr(9),chr(32)) ,

DECODE(hd.submitter, 'WEBSERVICES', 'Y', 'N') AS system\_created ,

v\_lASt\_updated

-- REPORTED\_NETWORK\_CONDITION

-- ENUMID REMEDY VALUE PRIORITY

-- 0 Network Event 3

-- 1 Degraded Service 2

-- 2 Outage 1

,

reported\_network\_condition ,

DECODE( notification\_required, 0, 'N', 1, 'Y') AS notification\_required

FROM aradmin.hpd\_help\_desk@dtvremp1.cisco.com hd

WHERE company NOTIN ('ROS-LAB', 'Cisco ROS')

AND hd.incident\_number IN

(SELECT incident\_number FROM im\_stage\_data\_ids\_to\_pull\_dtv

)

ORDER BY hd.incident\_number;

COMMIT;

## Bulk Collect

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

**SELECT i.incident\_number**

**, NVL(COUNT(CASE**

**WHEN t.tASkname = 'Initial Notification - Manual'**

**THEN 1**

**END),0) AS cnt BULK COLLECT**

**INTO my\_ticket\_list\_an, my\_tASk\_count\_list**

**FROM im\_stage\_data\_dtv i ,**

**aradmin.tms\_tASk@dtvremp1.cisco.com t**

**WHERE i.incident\_number = t.rootrequestid(+)**

**AND submit\_date > to\_date('11/12/2008', 'dd/mm/yyyy')**

**GROUP BY i.incident\_number;**

## Bulk Collect Cursor

**FOR i IN my\_ticket\_list\_an.FIRST .. my\_ticket\_list\_an.LAST**

**LOOP**

**IF my\_tASk\_count\_list(i) > 0 THEN**

**UPDATE im\_stage\_data\_dtv**

**SET auto\_notify\_flag = 'N'**

**WHERE my\_ticket\_list\_an(i) = incident\_number;**

**ELSE**

**UPDATE im\_stage\_data\_dtv**

**SET auto\_notify\_flag = 'Y'**

**WHERE my\_ticket\_list\_an(i) = incident\_number;**

**END IF;**

**END LOOP;**

SELECT

-- 0. ticket\_list

im.incident\_number

-- 1. company\_id

,

im.company ,

auto\_notify\_flag

-- 2. auto\_notify\_flag

-- 3. alarm\_to\_analyze

,

im.in\_progress\_date ,

im.reported\_date

-- 4. inc\_mttc

-- 6. met\_mttc sla

,

im.alarm\_to\_create\_mins

-- 5. inc\_mttn

-- 7. mttn\_sla\_met

,

im.tkt\_create\_to\_notify\_mins ,

im.submitter

-- 8-11. met\_Px\_SLA

,

im.priority ,

im.reported\_network\_condition ,

im.tkt\_create\_to\_resolved\_hrs BULK COLLECT

INTO

-- 0. ticket\_list

my\_ticket\_list

-- 1. company\_id

,

my\_company\_list

-- 2. auto\_notify\_flag

,

my\_auto\_notify\_list

-- 3. alarm\_to\_analyze

,

my\_in\_progress\_date\_list ,

my\_reported\_date\_list

-- 4. inc\_mttc

-- 6. met\_mttc sla

,

my\_alarm\_to\_crt\_mins\_list

-- 5. inc\_mttn

-- 7. mttn\_sla\_met

,

my\_tkt\_create\_to\_not\_mins\_list ,

my\_submitter\_list

-- 8-11. met\_Px\_SLA

,

my\_priority\_list ,

my\_reptd\_network\_cond\_list ,

my\_tkt\_create\_to\_res\_hrs\_list

FROM im\_stage\_data\_dtv im

WHERE 1 =1

AND submit\_date > to\_date('11/12/2008', 'dd/mm/yyyy');

-- and submit\_date > to\_date('01/06/2008', 'dd/mm/yyyy')

-- and submit\_date <= to\_date('12/12/2008', 'dd/mm/yyyy');

-- bulk collect cursor

FOR i IN my\_ticket\_list.FIRST .. my\_ticket\_list.LAST

LOOP

-- 1. company\_id

v\_company\_id := NULL;

-- 2. auto\_notify\_flag

v\_auto\_notify\_flag := NULL;

-- 3. alarm\_to\_analyze

v\_alarm\_to\_analyze\_mins := NULL;

-- 4. inc\_mttc

v\_inc\_mttcreate := NULL;

-- 6. met\_mttc\_sla

v\_mttcreate\_sla\_met:= NULL;

-- 5. met\_mttn

v\_inc\_mttn := NULL;

-- 7. met\_mttn\_sla

v\_mttn\_sla\_met:= NULL;

-- 8-11. met\_Px\_SLA

v\_mttr\_p1\_sla\_met := NULL;

v\_mttr\_p2\_sla\_met := NULL;

v\_mttr\_p3\_sla\_met := NULL;

v\_mttr\_p4\_sla\_met := NULL;

-- 1. company id

CASE

WHEN my\_company\_list(i) = 'DIRECTV-NGBN' THEN

v\_company\_id := 'CPY000000001337';

WHEN my\_company\_list(i) = 'DIRECTV-BROADBAND' THEN

v\_company\_id := 'CPY000000001338';

WHEN my\_company\_list(i) = 'DIRECTV-CONUS' THEN

v\_company\_id := 'CPY000000001339';

WHEN my\_company\_list(i) = 'DIRECTV' THEN

v\_company\_id := '';

END CASE;

-- 2. autonotify

v\_auto\_notify\_flag := my\_auto\_notify\_list(i);

-- 3. alarm\_to\_analyse\_mins in\_progress\_date - reported\_date

v\_alarm\_to\_analyze\_mins := (my\_in\_progress\_date\_list(i) - my\_reported\_date\_list(i))\*24\*60;

IF v\_alarm\_to\_analyze\_mins < 0 THEN

v\_alarm\_to\_analyze\_mins := NULL;

END IF;

-- 4. inc\_mttcreate

IF my\_alarm\_to\_crt\_mins\_list(i) >= 0 AND my\_alarm\_to\_crt\_mins\_list(i) < 100 AND my\_submitter\_list(i) = 'WEBSERVICES' THEN

v\_inc\_mttcreate := 'Y';

ELSE

v\_inc\_mttcreate:= 'N';

END IF;

-- 5. inc\_mttn

IF my\_alarm\_to\_crt\_mins\_list(i) >= 0 AND my\_alarm\_to\_crt\_mins\_list(i) < 100 AND v\_auto\_notify\_flag = 'N' THEN

v\_inc\_mttn := 'Y';

ELSE

v\_inc\_mttn := 'N';

END IF;

-- 6. MTTC determine if this record met MTTC SLA

IF my\_alarm\_to\_crt\_mins\_list(i) <= 7 AND v\_inc\_mttcreate = 'Y' THEN

v\_mttcreate\_sla\_met := 'Y';

ELSE

v\_mttcreate\_sla\_met := 'N';

END IF;

-- 7. met\_mttn\_sla

IF my\_tkt\_create\_to\_not\_mins\_list(i) <= 15 AND v\_inc\_mttn = 'Y' THEN

v\_mttn\_sla\_met := 'Y';

ELSE

v\_mttn\_sla\_met := 'N';

END IF;

-- 8 Determine P1 MTTR SLA

-- use the reported\_network\_condition if available otherwise use priority

-- determine if null

IF my\_reptd\_network\_cond\_list(i) IS NULL THEN

reported\_net\_cond\_is\_blank := TRUE;

END IF;

-- P1 met MTTR SLA

IF NOT (reported\_net\_cond\_is\_blank) AND my\_reptd\_network\_cond\_list(i) = 2 AND my\_tkt\_create\_to\_res\_hrs\_list(i) <= 4 THEN

v\_mttr\_p1\_sla\_met := 'Y';

ELSE

IF NOT (reported\_net\_cond\_is\_blank) AND my\_reptd\_network\_cond\_list(i)= 2 THEN

v\_mttr\_p1\_sla\_met := 'N';

ELSE

v\_mttr\_p1\_sla\_met := NULL;

END IF;

END IF;

IF reported\_net\_cond\_is\_blank AND my\_priority\_list(i) = 1 AND my\_tkt\_create\_to\_res\_hrs\_list(i) <= 4 THEN

v\_mttr\_p1\_sla\_met := 'Y';

ELSE

IF reported\_net\_cond\_is\_blank AND my\_priority\_list(i) = 1 THEN

v\_mttr\_p1\_sla\_met := 'N';

ELSE

v\_mttr\_p1\_sla\_met := NULL;

END IF;

END IF;

IF NOT (reported\_net\_cond\_is\_blank) AND my\_reptd\_network\_cond\_list(i) = 1 AND my\_tkt\_create\_to\_res\_hrs\_list(i) <= 4 THEN

v\_mttr\_p2\_sla\_met := 'Y';

ELSE

IF NOT (reported\_net\_cond\_is\_blank) AND my\_reptd\_network\_cond\_list(i)= 1 THEN

v\_mttr\_p2\_sla\_met := 'N';

ELSE

v\_mttr\_p2\_sla\_met := NULL;

END IF;

END IF;

IF reported\_net\_cond\_is\_blank AND my\_priority\_list(i) = 2 AND my\_tkt\_create\_to\_res\_hrs\_list(i) <= 4 THEN

v\_mttr\_p2\_sla\_met := 'Y';

ELSE

IF reported\_net\_cond\_is\_blank AND my\_priority\_list(i) = 2 THEN

v\_mttr\_p2\_sla\_met := 'N';

ELSE

v\_mttr\_p2\_sla\_met := NULL;

END IF;

END IF;

-- 10 Determine P3 MTTR SLA

IF NOT (reported\_net\_cond\_is\_blank) AND my\_reptd\_network\_cond\_list(i) = 0 AND my\_tkt\_create\_to\_res\_hrs\_list(i) <= 24 THEN

v\_mttr\_p3\_sla\_met := 'Y';

ELSE

IF NOT (reported\_net\_cond\_is\_blank) AND my\_reptd\_network\_cond\_list(i)= 0 THEN

v\_mttr\_p3\_sla\_met := 'N';

ELSE

v\_mttr\_p3\_sla\_met := NULL;

END IF;

END IF;

IF reported\_net\_cond\_is\_blank AND my\_priority\_list(i) = 3 AND my\_tkt\_create\_to\_res\_hrs\_list(i) <= 24 THEN

v\_mttr\_p3\_sla\_met := 'Y';

ELSE

IF reported\_net\_cond\_is\_blank AND my\_priority\_list(i) = 3 THEN

v\_mttr\_p3\_sla\_met := 'N';

ELSE

v\_mttr\_p3\_sla\_met := NULL;

END IF;

END IF;

BEGIN

UPDATE im\_stage\_data\_dtv

SET

-- 1. company\_id

companyid = v\_company\_id

-- 2. auto\_notify\_flag

,

auto\_notify\_flag = v\_auto\_notify\_flag

-- 3. alarm\_to\_analyze

,

alarm\_to\_analyze\_mins = v\_alarm\_to\_analyze\_mins

-- 4. inc\_mttc

,

inc\_mttcreate = v\_inc\_mttcreate

-- 6. met\_mttc\_sla

,

mttcreate\_sla\_met = v\_mttcreate\_sla\_met

-- 5. inc\_mttc

,

inc\_mttn = v\_inc\_mttn

-- 7. met\_mttn\_sla

,

mttn\_sla\_met = v\_mttn\_sla\_met

-- 8-11. met\_Px\_sla

,

mttr\_p1\_sla\_met = v\_mttr\_p1\_sla\_met ,

mttr\_p2\_sla\_met = v\_mttr\_p2\_sla\_met ,

mttr\_p3\_sla\_met = v\_mttr\_p3\_sla\_met ,

mttr\_p4\_sla\_met = v\_mttr\_p4\_sla\_met

WHERE incident\_number = my\_ticket\_list(i);

COMMIT;

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_im\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL' ,'IM\_GET\_DATA ERROR - update stage\_dtv', SYSDATE, v\_msg);

END;

END LOOP;

o\_im\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_im\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL' ,'IM\_GET\_DATA\_DTV ERROR', SYSDATE, v\_msg);

END im\_get\_data\_DTV;

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

PROCEDURE im\_update\_history\_dtv(

o\_im\_status OUT VARCHAR)

IS

v\_msg VARCHAR2(1000);

BEGIN

--To prevent duplicate rows we delete any dups that might occur

DELETE

FROM im\_history\_data\_dtv

WHERE incident\_number IN

(SELECT incident\_number FROM im\_stage\_data\_dtv

);

COMMIT;

--Update the im\_history table

INSERT

INTO im\_history\_data\_dtv

(

incident\_number ,

alarm\_identifier ,

alarm\_type ,

ASsigned\_group ,

ASsigned\_support\_company ,

ASsigned\_support\_organization ,

ASsignee ,

ASsignee\_groups ,

city ,

closure\_product\_category\_tier1 ,

closure\_product\_category\_tier2 ,

closure\_product\_category\_tier3 ,

closure\_product\_name ,

company ,

companyid ,

correlated\_ci ,

correlated\_id ,

country ,

country\_code ,

created\_by ,

effort\_time\_spent\_minutes ,

first\_name ,

generic\_categorization\_tier\_1 ,

generic\_categorization\_tier\_2 ,

generic\_categorization\_tier\_3 ,

impact ,

impact\_rating ,

lASt\_modified\_by ,

lASt\_name ,

lookupkeyword ,

managed ,

manufacturer ,

mapped\_severity ,

mom\_identifier ,

mom\_type ,

organization ,

owner ,

owner\_group ,

owner\_group\_id ,

owner\_support\_company ,

owner\_support\_organization ,

partner ,

priority ,

product\_categorization\_tier\_1 ,

product\_categorization\_tier\_2 ,

product\_categorization\_tier\_3 ,

product\_model\_version ,

product\_name ,

recurrence ,

region ,

related\_alarm\_code ,

reported\_source ,

reportedci ,

resolution\_category ,

resolution\_category\_tier\_2 ,

resolution\_category\_tier\_3 ,

service\_type ,

severity ,

short\_description ,

site ,

site\_group ,

site\_id ,

state\_province ,

status ,

status\_reASon ,

submitter ,

total\_escalation\_level ,

total\_ola\_acknowledgeesc\_level ,

total\_ola\_resolution\_esc\_level ,

total\_time\_spent ,

total\_transfers ,

urgency

-- Atomic Dates

,

closed\_date ,

estimated\_resolution\_date ,

initial\_notification\_date ,

isolation\_date ,

lASt\_\_ASsigned\_date ,

lASt\_ackowledged\_date ,

lASt\_modified\_date ,

lASt\_occurence ,

lASt\_resolved\_date ,

reported\_date ,

responded\_date ,

submit\_date

-- Calculated Dates

,

in\_progress\_date ,

day\_closed ,

day\_opened ,

month\_closed ,

month\_opened ,

week\_closed ,

week\_opened

-- Calculated Flags

,

auto\_notify\_flag ,

auto\_tkt\_flag ,

closed\_AS\_duplicate\_flag ,

closed\_AS\_ros\_infrAStr\_flag ,

closed\_AS\_merged\_flag ,

sched\_maint\_flag

-- Calculated Counts

,

num\_cust\_escalations ,

num\_customer\_updates ,

num\_entitlement\_wil ,

num\_external\_escalations ,

num\_internal\_escalations ,

num\_touches ,

off\_clock\_time\_mins ,

alarm\_to\_notify\_mins ,

alarm\_to\_isolate\_mins ,

alarm\_to\_resolve\_mins ,

alarm\_to\_close\_mins ,

alarm\_to\_analyze\_mins ,

alarm\_to\_create\_mins ,

Tkt\_Create\_To\_Closed\_Hrs ,

Tkt\_Create\_To\_Isolate\_Mins ,

Tkt\_Create\_To\_Notify\_Mins ,

Tkt\_Create\_To\_Resolved\_Hrs ,

Tkt\_Isolate\_To\_Resolved\_Mins ,

Tkt\_Resolved\_To\_Closed\_Mins ,

data\_source ,

inc\_mttcreate ,

inc\_mttn ,

description ,

system\_created ,

mttcreate\_sla\_met ,

mttn\_sla\_met ,

mttr\_p1\_sla\_met ,

mttr\_p2\_sla\_met ,

mttr\_p3\_sla\_met ,

mttr\_p4\_sla\_met ,

lASt\_updated ,

reported\_network\_condition ,

notification\_required

)

SELECT incident\_number ,

alarm\_identifier ,

alarm\_type ,

ASsigned\_group ,

ASsigned\_support\_company ,

ASsigned\_support\_organization ,

ASsignee ,

ASsignee\_groups ,

city ,

closure\_product\_category\_tier1 ,

closure\_product\_category\_tier2 ,

closure\_product\_category\_tier3 ,

closure\_product\_name ,

company ,

companyid ,

correlated\_ci ,

correlated\_id ,

country ,

country\_code ,

created\_by ,

effort\_time\_spent\_minutes ,

first\_name ,

generic\_categorization\_tier\_1 ,

generic\_categorization\_tier\_2 ,

generic\_categorization\_tier\_3 ,

impact ,

impact\_rating ,

lASt\_modified\_by ,

lASt\_name ,

lookupkeyword ,

managed ,

manufacturer ,

mapped\_severity ,

mom\_identifier ,

mom\_type ,

organization ,

owner ,

owner\_group ,

owner\_group\_id ,

owner\_support\_company ,

owner\_support\_organization ,

partner ,

priority ,

product\_categorization\_tier\_1 ,

product\_categorization\_tier\_2 ,

product\_categorization\_tier\_3 ,

product\_model\_version ,

product\_name ,

recurrence ,

region ,

related\_alarm\_code ,

reported\_source ,

reportedci ,

resolution\_category ,

resolution\_category\_tier\_2 ,

resolution\_category\_tier\_3 ,

service\_type ,

severity ,

short\_description ,

site ,

site\_group ,

site\_id ,

state\_province ,

status ,

status\_reASon ,

submitter ,

total\_escalation\_level ,

total\_ola\_acknowledgeesc\_level ,

total\_ola\_resolution\_esc\_level ,

total\_time\_spent ,

total\_transfers ,

urgency

-- Atomic Dates

,

closed\_date ,

estimated\_resolution\_date ,

initial\_notification\_date ,

isolation\_date ,

lASt\_\_ASsigned\_date ,

lASt\_ackowledged\_date ,

lASt\_modified\_date ,

lASt\_occurence ,

lASt\_resolved\_date ,

reported\_date ,

responded\_date ,

submit\_date

-- Calculated Dates

,

in\_progress\_date ,

day\_closed ,

day\_opened ,

month\_closed ,

month\_opened ,

week\_closed ,

week\_opened

-- Calculated Flags

,

auto\_notify\_flag ,

auto\_tkt\_flag ,

closed\_AS\_duplicate\_flag ,

closed\_AS\_ros\_infrAStr\_flag ,

closed\_AS\_merged\_flag ,

sched\_maint\_flag

-- Calculated Counts

,

num\_cust\_escalations ,

num\_customer\_updates ,

num\_entitlement\_wil ,

num\_external\_escalations ,

num\_internal\_escalations ,

num\_touches ,

off\_clock\_time\_mins ,

alarm\_to\_notify\_mins ,

alarm\_to\_isolate\_mins ,

alarm\_to\_resolve\_mins ,

alarm\_to\_close\_mins ,

alarm\_to\_analyze\_mins ,

alarm\_to\_create\_mins ,

Tkt\_Create\_To\_Closed\_Hrs ,

Tkt\_Create\_To\_Isolate\_Mins ,

Tkt\_Create\_To\_Notify\_Mins ,

Tkt\_Create\_To\_Resolved\_Hrs ,

Tkt\_Isolate\_To\_Resolved\_Mins ,

Tkt\_Resolved\_To\_Closed\_Mins ,

data\_source ,

inc\_mttcreate ,

inc\_mttn ,

description ,

system\_created ,

mttcreate\_sla\_met ,

mttn\_sla\_met ,

mttr\_p1\_sla\_met ,

mttr\_p2\_sla\_met ,

mttr\_p3\_sla\_met ,

mttr\_p4\_sla\_met ,

lASt\_updated ,

reported\_network\_condition ,

notification\_required

FROM im\_stage\_data\_dtv;

COMMIT;

o\_im\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_im\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL' ,'IM\_DTV\_UPDATE\_HISTORY ERROR', SYSDATE, v\_msg);

END im\_update\_history\_dtv;

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

--Rolls up Incident Managment data for reporting

PROCEDURE im\_rollup\_data\_dtv(

i\_start\_date IN DATE,

i\_stop\_date IN DATE,

i\_period\_id IN NUMBER,

i\_period\_type IN VARCHAR,

i\_period\_name IN VARCHAR,

o\_im\_status OUT VARCHAR)

IS

v\_msg VARCHAR2(1000);

v\_status VARCHAR2(20);

BEGIN

-- this is an external procedure

im\_kpi\_ru\_proc\_dtv (i\_start\_date ,i\_stop\_date,i\_period\_id ,i\_period\_type,i\_period\_name ,o\_im\_status);

IF i\_period\_type = 'DAY' THEN

-- this is an external procedure

im\_backlog\_buckets\_proc\_dtv (i\_start\_date,i\_stop\_date,i\_period\_id ,i\_period\_type,i\_period\_name,o\_im\_status);

-- this is an external procedure

im\_mttr\_buckets\_proc\_dtv (i\_start\_date,i\_stop\_date,i\_period\_id ,i\_period\_type,i\_period\_name,o\_im\_status);

END IF;

o\_im\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_im\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL', 'IM\_ROLLUP\_DATA ERROR', SYSDATE, v\_msg);

END im\_rollup\_data\_dtv;

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

-- PROBLEM MANAGEMENT

-- Gets stage data FROM Problem Managment bASe tables

PROCEDURE pm\_get\_data\_dtv(

i\_start\_date IN DATE ,

i\_stop\_date IN DATE ,

o\_pm\_status OUT VARCHAR)

v\_msg VARCHAR2(1000);

BEGIN

-- Problem tickets

-- 4. delete pbi record ids and records FROM lASt pull

METRIC\_UTILITIES.TRUNC\_TABLE('pm\_ids\_to\_pull');

METRIC\_UTILITIES.TRUNC\_TABLE('pm\_stage\_data');

COMMIT;

-- 5. get pbi ids to pull this time

INSERT

INTO pm\_ids\_to\_pull (pbi\_id)

SELECT ppi.problem\_investigation\_id

FROM aradmin.pbm\_problem\_investigation@DTVREMP1.CISCO.COM ppi

WHERE metric\_utilities.convert\_to\_date(lASt\_modified\_date) > i\_start\_date

UNION

SELECT piw.problem\_investigation\_id

FROM aradmin.pbm\_investigation\_worklog@DTVREMP1.CISCO.COM piw

WHERE metric\_utilities.convert\_to\_date(lASt\_modified\_date) > i\_start\_date

UNION

SELECT psda.request\_id01

FROM aradmin.pbm\_solution\_databASe@DTVREMP1.CISCO.COM psd,

aradmin.pbm\_solution\_db\_ASsociations@DTVREMP1.CISCO.COM psda

WHERE psd.solution\_databASe\_id = psda.request\_id02

AND SUBSTR(psda.request\_id01,1,3) = 'PBI'

AND metric\_utilities.convert\_to\_date(psd.lASt\_modified\_date) >

i\_start\_date;

COMMIT;

-- 6. insert into pm\_stage\_data

INSERT

INTO pm\_stage\_data

(

-- atomic fields (1-1 mapping)

pbi\_id,

submit\_date,

lASt\_modified\_date,

status,

priority,

status\_reASon,

company,

site,

ASsigned\_group,

ASsignee,

categorization\_tier1,

categorization\_tier2,

categorization\_tier3,

product\_categorization\_tier1,

product\_categorization\_tier2,

product\_categorization\_tier3,

-- calculated dates

lASt\_touch\_date,

closed\_date,

resolved\_date,

-- calculated times

ttr\_seconds,

ttc\_seconds,

mtttouch\_seconds,

-- calculated flags

closed\_AS\_duplicate\_flag,

pbi\_created\_by\_inc\_flag,

pbi\_created\_by\_crq\_flag,

pbi\_created\_crq\_flag,

pbi\_created\_sdb\_flag,

-- calculated counts

num\_touches,

num\_escalations,

num\_crq\_ASsociations,

num\_inc\_ASsociations,

num\_sdb\_ASsociations,

num\_other\_ASsociations,

data\_source

)

SELECT

-- atomic fields (1-1 mapping)

DISTINCT(ppi.problem\_investigation\_id),

metric\_utilities.convert\_to\_date(ppi.submit\_date) AS submit\_date,

metric\_utilities.convert\_to\_date(ppi.lASt\_modified\_date)AS lASt\_modified\_date,

DECODE (ppi.investigation\_status, 0, 'Draft', 1, 'Under Review', 2, 'Request For Authorization', 3, 'ASsigned', 4, 'Under Investigation', 5, 'Pending', 6, 'Completed', 7, 'Rejected', 8, 'Closed', 9, 'Cancelled') AS status,

ppi.priority,

DECODE (ppi.invesitgation\_status\_reASon, 1000, 'Known Error', 2000, 'Unresolveable', 3000, 'Solution DatabASe', 4000, 'Enhancement Request', 5000, 'Duplicate Investigation', 'None' ) AS status\_reASon,

ppi.company,

ppi.site,

ppi.ASsigned\_group,

NVL(ppi.ASsignee, 'Not ASsigned'),

NVL(ppi.categorization\_tier\_1,'No User Value'),

NVL(ppi.categorization\_tier\_2,'No User Value'),

NVL(ppi.categorization\_tier\_3,'No User Value'),

NVL(ppi.product\_categorization\_tier\_1,'No User Value'),

NVL(ppi.product\_categorization\_tier\_2,'No User Value'),

NVL(ppi.product\_categorization\_tier\_3,'No User Value'),

-- calculated dates

calc\_mmttouch.lASt\_touch\_date,

calc\_ttc.closed\_date,

calc\_ttr.resolved\_date,

-- calculated times

NVL(calc\_ttr.ttr\_seconds,0),

NVL(calc\_ttc.ttc\_seconds,0),

NVL(calc\_mmttouch.mtttouch,0),

-- calculated flags

CASE

WHEN ppi.investigation\_status = 8

AND ppi.invesitgation\_status\_reASon = 5000

THEN 'Y'

ELSE 'N'

END AS closed\_AS\_duplicate\_flag,

'N' pbi\_created\_by\_inc\_flag,

'N' pbi\_created\_by\_crq\_flag,

'N' pbi\_created\_crq\_flag,

'N' pbi\_created\_sdb\_flag,

-- calculated counts

NVL(count\_touches.num\_touches,0),

NVL(count\_escalations.num\_escalations,0),

NVL(count\_ASsociations.num\_CRQ\_ASsociations,0),

NVL(count\_ASsociations.num\_INC\_ASsociations,0),

NVL(count\_ASsociations.num\_SDB\_ASsociations,0),

NVL(count\_ASsociations.num\_other\_ASsociations,0),

'DTV'

FROM aradmin.pbm\_problem\_investigation@DTVREMP1.CISCO.COM ppi,

aradmin.pbm\_investigation\_ASsociations@DTVREMP1.CISCO.COM pia,

-- calc\_mmttouch

(

SELECT piw.problem\_investigation\_id pbi\_id,

metric\_utilities.convert\_to\_date(MAX(piw.submit\_date)) lASt\_touch\_date,

metric\_utilities.time\_diff(metric\_utilities.convert\_to\_date(MIN(piw.submit\_date)), metric\_utilities.convert\_to\_date(MAX(piw.submit\_date))) / (COUNT(work\_log\_id)-1) mtttouch

FROM aradmin.pbm\_investigation\_worklog@DTVREMP1.CISCO.COM piw

HAVING (COUNT(work\_log\_id)-1) > 0

GROUP BY piw.problem\_investigation\_id

) calc\_mmttouch,

-- calc\_ttc

(

SELECT ppi.problem\_investigation\_id pbi\_id,

metric\_utilities.convert\_to\_date(h.t8) closed\_date,

NVL(metric\_utilities.time\_diff( metric\_utilities.convert\_to\_date(ppi.submit\_date), metric\_utilities.convert\_to\_date(h.t8)),0) ttc\_seconds

FROM aradmin.pbm\_problem\_investigation@DTVREMP1.CISCO.COM ppi,

aradmin.h1121@DTVREMP1.CISCO.COM h

WHERE ppi.sys\_problem\_investigation\_id = h.entryid

) calc\_ttc,

-- calc\_ttr

(

SELECT ppi.problem\_investigation\_id AS pbi\_id,

metric\_utilities.convert\_to\_date(h.t6) resolved\_date,

NVL(metric\_utilities.time\_diff( metric\_utilities.convert\_to\_date(ppi.submit\_date), metric\_utilities.convert\_to\_date(h.t6)),0) ttr\_seconds

FROM aradmin.pbm\_problem\_investigation@DTVREMP1.CISCO.COM ppi,

aradmin.h1121@DTVREMP1.CISCO.COM h

WHERE ppi.sys\_problem\_investigation\_id = h.entryid

) calc\_ttr,

-- count\_touches

(

SELECT ppi.problem\_investigation\_id AS pbi\_id,

COUNT(piw.problem\_investigation\_id) AS num\_touches

FROM aradmin.pbm\_problem\_investigation@DTVREMP1.CISCO.COM ppi,

aradmin.pbm\_investigation\_worklog@DTVREMP1.CISCO.COM piw

WHERE ppi.problem\_investigation\_id = piw.problem\_investigation\_id

GROUP BY ppi.problem\_investigation\_id

) count\_touches,

-- count\_escalations

(

SELECT ppi.problem\_investigation\_id pbi\_id,

COUNT(piw.problem\_investigation\_id) AS num\_escalations

FROM aradmin.pbm\_problem\_investigation@DTVREMP1.CISCO.COM ppi,

aradmin.pbm\_investigation\_worklog@DTVREMP1.CISCO.COM piw

WHERE ppi.problem\_investigation\_id = piw.problem\_investigation\_id

AND SUBSTR(piw.description,1,17) = 'Formal escalation'

GROUP BY ppi.problem\_investigation\_id

) count\_escalations,

-- count\_ASsociations

(

SELECT pia.request\_id02,

NVL(COUNT(

CASE

WHEN SUBSTR(pia.request\_id01,1,3) = 'CRQ'

THEN 1

ELSE NULL

END),0) AS num\_CRQ\_ASsociations,

NVL(COUNT(

CASE

WHEN SUBSTR(pia.request\_id01,1,3) = 'INC'

THEN 1

ELSE NULL

END),0) AS num\_INC\_ASsociations,

NVL(COUNT(

CASE

WHEN SUBSTR(pia.request\_id01,1,3) = 'SDB'

THEN 1

ELSE NULL

END),0) AS num\_SDB\_ASsociations,

NVL(COUNT(

CASE

WHEN SUBSTR(pia.request\_id01,1,3) NOT IN ('CRQ','INC','SDB')

THEN 1

ELSE NULL

END),0) AS num\_other\_ASsociations

FROM aradmin.pbm\_investigation\_ASsociations@DTVREMP1.CISCO.COM pia

GROUP BY pia.request\_id02

) count\_ASsociations

WHERE 1 =1

AND ppi.problem\_investigation\_id = calc\_mmttouch.pbi\_id(+)

AND ppi.problem\_investigation\_id = calc\_ttc.pbi\_id(+)

AND ppi.problem\_investigation\_id = calc\_ttr.pbi\_id(+)

AND ppi.problem\_investigation\_id = pia.request\_id02(+)

AND ppi.problem\_investigation\_id = count\_touches.pbi\_id(+)

AND ppi.problem\_investigation\_id = count\_escalations.pbi\_id(+)

AND ppi.problem\_investigation\_id = count\_ASsociations.request\_id02(+)

AND ppi.problem\_investigation\_id IN

(SELECT DISTINCT pbi\_id FROM pm\_ids\_to\_pull

)

ORDER BY ppi.problem\_investigation\_id;

COMMIT;

-- 7. update pbi association flags

DECLARE

CURSOR pbi\_ticket\_cur

IS

SELECT pbi\_id FROM pm\_stage\_data ORDER BY pbi\_id;

CURSOR pbi\_ASsoc\_cur(i\_pbi\_id IN VARCHAR2)

IS

SELECT form\_name01 ,

form\_name02 ,

request\_id01 ,

request\_id02 ,

request\_description01 ,

ASsociation\_type01

## Simple DB Link Example

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

**FROM aradmin.pbm\_investigation\_ASsociations@DTVREMP1.CISCO.COM pia**

WHERE request\_id02 IS NOT NULL

-- AND substr(request\_id02, length(request\_id02)-3) = 341;

AND request\_id02 = i\_pbi\_id;

BEGIN

## Nested Cursor

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

-- for each record

**FOR pbi\_ticket\_rec IN pbi\_ticket\_cur**

**LOOP**

**-- grab all the Associations**

**FOR pbi\_ASsoc\_rec IN pbi\_ASsoc\_cur(pbi\_ticket\_rec.pbi\_id)**

**LOOP**

**-- for each ASsociation**

**-- if the pbi wAS created by an incident**

**IF pbi\_ASsoc\_rec.ASsociation\_type01 = '31000' AND SUBSTR(pbi\_ASsoc\_rec.request\_id01,1,3) = 'INC' THEN**

**UPDATE pm\_stage\_data**

**SET pbi\_created\_by\_inc\_flag = 'Y'**

**WHERE pm\_stage\_data.pbi\_id = pbi\_ASsoc\_rec.request\_id02;**

**END IF;**

**-- if the pbi wAS created by an change**

**IF pbi\_ASsoc\_rec.ASsociation\_type01 = '31000' AND SUBSTR(pbi\_ASsoc\_rec.request\_id01,1,3) = 'CRQ' THEN**

**UPDATE pm\_stage\_data**

**SET pbi\_created\_by\_crq\_flag = 'Y'**

**WHERE pm\_stage\_data.pbi\_id = pbi\_ASsoc\_rec.request\_id02;**

**END IF;**

**-- if the pbi created a change request**

**IF pbi\_ASsoc\_rec.ASsociation\_type01 = '30000' AND SUBSTR(pbi\_ASsoc\_rec.request\_id01,1,3) = 'CRQ' THEN**

**UPDATE pm\_stage\_data**

**SET pbi\_created\_crq\_flag = 'Y'**

**WHERE pm\_stage\_data.pbi\_id = pbi\_ASsoc\_rec.request\_id02;**

**END IF;**

**-- if the pbi created a solution**

**IF pbi\_ASsoc\_rec.ASsociation\_type01 = '30000' AND SUBSTR(pbi\_ASsoc\_rec.request\_id01,1,3) = 'SDB' THEN**

**UPDATE pm\_stage\_data**

**SET pbi\_created\_sdb\_flag = 'Y'**

**WHERE pm\_stage\_data.pbi\_id = pbi\_ASsoc\_rec.request\_id02;**

**END IF;**

**END LOOP;**

**END LOOP;**

END;

-- Solution tickets

-- 17. delete sdb record ids and records FROM lASt pull

METRIC\_UTILITIES.TRUNC\_TABLE('pm\_sbd\_ids\_to\_pull');

METRIC\_UTILITIES.TRUNC\_TABLE('pm\_sdb\_stage\_data');

-- 18. get sdb ids to pull this time

INSERT

INTO pm\_sbd\_ids\_to\_pull(sdb\_id)

SELECT sdb.solution\_databASe\_id

FROM aradmin.pbm\_solution\_databASe@DTVREMP1.CISCO.COM sdb

WHERE metric\_utilities.convert\_to\_date(sdb.lASt\_modified\_date) > i\_start\_date;

-- 19. insert into sdb\_stage\_data

INSERT

INTO pm\_sdb\_stage\_data

(

sdb\_id,

submit\_date,

company,

site,

ASsigned\_group,

ASsignee,

status,

data\_source

)

SELECT solution\_databASe\_id sdb\_id,

metric\_utilities.convert\_to\_date(sdb.submit\_date) AS submit\_date,

sdb.company,

sdb.site,

sdb.ASsigned\_group,

NVL(sdb.ASsignee, 'Not ASsigned'),

DECODE (sdb.status, 0, 'Proposed', 1, 'Enabled', 2, 'Offline', 3, 'Obsolete', 4, 'Archive', 5, 'Delete') AS status,

'DTV'

FROM aradmin.pbm\_solution\_databASe@DTVREMP1.CISCO.COM sdb

WHERE sdb.solution\_databASe\_id IN

(SELECT DISTINCT sdb\_id FROM pm\_sbd\_ids\_to\_pull

);

o\_pm\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_pm\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL' ,'PM\_GET\_DATA\_DTV ERROR', SYSDATE, v\_msg);

END pm\_get\_data\_dtv;

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

-- Update the history table

PROCEDURE pm\_update\_history\_dtv(

o\_pm\_status OUT VARCHAR)

IS

v\_msg VARCHAR2(1000);

BEGIN

DELETE

FROM pm\_history\_data

WHERE data\_source = 'DTV'

AND pbi\_id IN

(SELECT pbi\_id FROM pm\_stage\_data

);

COMMIT;

-- 9. insert into pm\_history

INSERT

INTO pm\_history\_data

(

pbi\_id,

submit\_date,

lASt\_modified\_date,

status,

priority,

status\_reASon,

company,

site,

ASsigned\_group,

ASsignee,

categorization\_tier1,

categorization\_tier2,

categorization\_tier3,

product\_categorization\_tier1,

product\_categorization\_tier2,

product\_categorization\_tier3,

-- calculated dates

lASt\_touch\_date,

closed\_date,

resolved\_date,

-- calculated times

ttr\_seconds,

ttc\_seconds,

mtttouch\_seconds,

-- calculated flags

closed\_AS\_duplicate\_flag,

pbi\_created\_by\_inc\_flag,

pbi\_created\_by\_crq\_flag,

pbi\_created\_crq\_flag,

pbi\_created\_sdb\_flag,

-- calculated counts

num\_touches,

num\_escalations,

num\_crq\_ASsociations,

num\_inc\_ASsociations,

num\_sdb\_ASsociations,

num\_other\_ASsociations,

data\_source

)

SELECT pbi\_id,

submit\_date,

lASt\_modified\_date,

status,

priority,

status\_reASon,

company,

site,

ASsigned\_group,

ASsignee,

categorization\_tier1,

categorization\_tier2,

categorization\_tier3,

product\_categorization\_tier1,

product\_categorization\_tier2,

product\_categorization\_tier3,

-- calculated dates

lASt\_touch\_date,

closed\_date,

resolved\_date,

-- calculated times

ttr\_seconds,

ttc\_seconds,

mtttouch\_seconds,

-- calculated flags

closed\_AS\_duplicate\_flag,

pbi\_created\_by\_inc\_flag,

pbi\_created\_by\_crq\_flag,

pbi\_created\_crq\_flag,

pbi\_created\_sdb\_flag,

-- calculated counts

num\_touches,

num\_escalations,

num\_crq\_ASsociations,

num\_inc\_ASsociations,

num\_sdb\_ASsociations,

num\_other\_ASsociations,

data\_source

FROM pm\_stage\_data;

-- 20. delete FROM pm\_sdb\_history

DELETE

FROM pm\_sdb\_history\_data

WHERE data\_source = 'DTV'

AND sdb\_id IN

(SELECT sdb\_id FROM pm\_sdb\_stage\_data

);

-- 21. insert into sdb\_history

INSERT

INTO pm\_sdb\_history\_data

(

sdb\_id,

submit\_date,

company,

site,

ASsigned\_group,

ASsignee,

status,

data\_source

)

SELECT sdb\_id,

submit\_date,

company,

site,

ASsigned\_group,

ASsignee,

status,

data\_source

FROM pm\_sdb\_stage\_data;

COMMIT;

o\_pm\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_pm\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL' ,'PM\_UPDATE\_HISTORY ERROR', SYSDATE, v\_msg);

END pm\_update\_history\_dtv;

-- Rolls up Problem Managment data for reporting

PROCEDURE pm\_rollup\_data\_dtv(

o\_pm\_status OUT VARCHAR)

IS

v\_msg VARCHAR2(1000);

v\_status VARCHAR2(15);

BEGIN

metric\_utilities.refresh\_view('pm\_rollup\_company\_site\_VW','A', v\_status );

metric\_utilities.refresh\_view('pm\_rollup\_team\_ASsignee\_VW','A', v\_status );

metric\_utilities.refresh\_view('pm\_rollup\_tkt\_metrics\_VW','AF', v\_status );

metric\_utilities.refresh\_view('pm\_rollup\_tkt\_data\_VW','A', v\_status );

metric\_utilities.refresh\_view('pm\_sdb\_rollup\_comp\_site\_VW','A', v\_status );

metric\_utilities.refresh\_view('pm\_sdb\_rollup\_team\_ASsn\_VW','A', v\_status );

metric\_utilities.refresh\_view('pm\_rollup\_sdb\_tkt\_data\_VW','A', v\_status );

o\_pm\_status := 'SUCCESS';

## Exception Sample

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

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_pm\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL' ,'PM\_ROLLUP\_DATA ERROR', SYSDATE, v\_msg);

END pm\_rollup\_data\_dtv;

--Gets stage data FROM Change Managment bASe tables

PROCEDURE cm\_get\_data\_dtv(

i\_start\_date IN DATE ,

i\_stop\_date IN DATE ,

o\_cm\_status OUT VARCHAR)

IS

v\_msg VARCHAR2(1000);

BEGIN

metric\_utilities.log\_writer('CM','cm\_dtv\_get\_stage\_data procedure start',SYSDATE,NULL) ;

METRIC\_UTILITIES.TRUNC\_TABLE('cm\_change\_id\_list');

--SELECT \* FROM cm\_change\_id\_list;

INSERT

INTO cm\_change\_id\_list (request\_id)

(SELECT cic.infrAStructure\_change\_id

FROM aradmin.chg\_infrAStructure\_change@DTVREMP1.CISCO.COM cic

WHERE metric\_utilities.convert\_to\_date(cic.lASt\_modified\_date) BETWEEN i\_start\_date AND i\_stop\_date

UNION

SELECT ca.request\_id02

FROM aradmin.chg\_ASsociations@DTVREMP1.CISCO.COM ca

WHERE metric\_utilities.convert\_to\_date(ca.modified\_date) BETWEEN i\_start\_date AND i\_stop\_date

UNION

SELECT cw.infrAStructure\_change\_id

FROM aradmin.chg\_worklog@DTVREMP1.CISCO.COM cw

WHERE metric\_utilities.convert\_to\_date(cw.lASt\_modified\_date) BETWEEN i\_start\_date AND i\_stop\_date);

METRIC\_UTILITIES.TRUNC\_TABLE('cm\_stage\_data');

COMMIT;

-- Fill CM\_STAGE\_DATA

INSERT

INTO cm\_stage\_data

(

-- ATOMIC FIELDS

infrAStructure\_change\_id ,

ASchg ,

ASgrp ,

ASsigned\_to ,

ASsignee\_id ,

ASsignee\_id\_ASsignee ,

ASsignee\_id\_manager ,

business\_justification ,

categorization\_tier\_1 ,

categorization\_tier\_2 ,

categorization\_tier\_3 ,

change\_request\_status ,

change\_revierwer\_name ,

change\_subtype ,

chgimp ,

chgimpgrp ,

customer\_area\_code ,

customer\_company ,

customer\_corporate\_id ,

customer\_department ,

customer\_extension ,

customer\_first\_name ,

customer\_lASt\_name ,

customer\_local\_phone ,

customer\_organization ,

customer\_phone\_number ,

department ,

first\_name ,

impact ,

lASt\_modified\_by ,

lASt\_name ,

location\_company ,

organization ,

owner\_group ,

owner\_support\_organization ,

phone\_number ,

product\_cat\_tier\_1\_2\_ ,

product\_cat\_tier\_2\_\_2\_ ,

product\_cat\_tier\_3\_\_2\_ ,

product\_categorization\_tier\_1 ,

product\_categorization\_tier\_2 ,

product\_categorization\_tier\_3 ,

product\_model\_version\_\_2\_ ,

product\_name ,

product\_name\_\_2\_ ,

region ,

request\_id ,

requester\_contacted ,

site ,

site\_group ,

site\_id ,

submitter ,

support\_group\_name ,

support\_group\_name2 ,

support\_organization ,

support\_organization2 ,

total\_time\_spent ,

urgency

-- Calculated Values

,

change\_timing\_value ,

customer\_email ,

description ,

priority ,

risk\_level ,

status ,

status\_reASon

-- Atomic Dates

,

acknowledgment\_end\_date ,

actual\_end\_date ,

actual\_start\_date ,

completed\_date ,

earliest\_start\_date ,

lASt\_modified\_date ,

next\_target\_date ,

requested\_end\_date ,

requested\_start\_date ,

resolution\_end\_date ,

response\_target\_date ,

rfc\_date ,

scheduled\_end\_date ,

scheduled\_start\_date ,

submit\_date

-- Calculated Dates

,

completed\_date\_h ,

closed\_date\_h ,

month\_completed\_h ,

month\_opened

-- Calculated Flags

,

crq\_created\_by\_inc ,

chg\_type\_is\_change\_flag ,

chg\_is\_rejected\_flag ,

chg\_cancelled\_rescheduled\_flag ,

chg\_is\_emergency\_flag ,

chg\_is\_backed\_out\_flag ,

chg\_is\_successful\_flag ,

chg\_is\_expedited\_flag

-- Calculated Counts

,

num\_inc\_FROM\_this\_crq

-- Calculated Times

,

time\_to\_complete\_days

-- Data Source

,

data\_source

)

SELECT

-- Atomic Fields

cic.infrAStructure\_change\_id ,

cic.ASchg ,

cic.ASgrp ,

cic.ASsigned\_to ,

cic.ASsignee\_id ,

cic.ASsignee\_id\_ASsignee ,

cic.ASsignee\_id\_manager ,

cic.business\_justification ,

cic.categorization\_tier\_1 ,

cic.categorization\_tier\_2 ,

cic.categorization\_tier\_3 ,

cic.change\_request\_status ,

cic.change\_revierwer\_name ,

cic.change\_subtype ,

cic.chgimp ,

cic.chgimpgrp ,

cic.customer\_area\_code ,

cic.customer\_company ,

cic.customer\_corporate\_id ,

cic.customer\_department ,

cic.customer\_extension ,

cic.customer\_first\_name ,

cic.customer\_lASt\_name ,

cic.customer\_local\_phone ,

cic.customer\_organization ,

cic.customer\_phone\_number ,

cic.department ,

cic.first\_name ,

cic.impact ,

cic.last\_modified\_by ,

cic.last\_name ,

cic.location\_company ,

cic.organization ,

cic.owner\_group ,

cic.owner\_support\_organization ,

cic.phone\_number ,

cic.product\_cat\_tier\_1\_2\_ ,

cic.product\_cat\_tier\_2\_\_2\_ ,

cic.product\_cat\_tier\_3\_\_2\_ ,

cic.product\_categorization\_tier\_1 ,

cic.product\_categorization\_tier\_2 ,

cic.product\_categorization\_tier\_3 ,

cic.product\_model\_version\_\_2\_ ,

cic.product\_name ,

cic.product\_name\_\_2\_ ,

cic.region ,

cic.request\_id ,

cic.requester\_contacted ,

cic.risk\_level ,

cic.site ,

cic.site\_group ,

cic.site\_id ,

cic.submitter ,

cic.support\_group\_name ,

cic.support\_organization ,

cic.support\_organization2 ,

cic.total\_time\_spent ,

cic.urgency

-- Calculated Values

,

metric\_utilities.enum\_values(1200,1000000568,cic.change\_timing) AS change\_timing\_value ,

REPLACE(REPLACE(cic.customer\_internet\_e\_mail,chr(10),' '),chr(13),' ') AS customer\_email ,

REPLACE(REPLACE(cic.description,chr(10),' '),chr(13),' ') AS description ,

metric\_utilities.enum\_values(1200,1000000164,cic.priority) AS priority ,

metric\_utilities.enum\_values(1200,1000000180,cic.risk\_level) AS risk\_level ,

metric\_utilities.enum\_values(1200,7,cic.change\_request\_status) AS status ,

metric\_utilities.enum\_values(1200,1000000150,cic.status\_reASon) AS status\_reASon

-- Atomic Dates

,

metric\_utilities.convert\_to\_date(cic.acknowledgment\_end\_date)AS acknowledgment\_end\_date ,

metric\_utilities.convert\_to\_date(cic.actual\_end\_date) AS actual\_end\_date ,

metric\_utilities.convert\_to\_date(cic.actual\_start\_date)AS actual\_start\_date ,

metric\_utilities.convert\_to\_date(cic.completed\_date) AS completed\_date ,

metric\_utilities.convert\_to\_date(cic.earliest\_start\_date) AS earliest\_start\_date ,

metric\_utilities.convert\_to\_date(cic.lASt\_modified\_date) AS lASt\_modified\_date ,

metric\_utilities.convert\_to\_date(cic.next\_target\_date) AS next\_target\_date ,

metric\_utilities.convert\_to\_date(cic.requested\_end\_date) AS requested\_end\_date ,

metric\_utilities.convert\_to\_date(cic.requested\_start\_date) AS requested\_start\_date ,

metric\_utilities.convert\_to\_date(cic.resolution\_end\_date) AS resolution\_end\_date ,

metric\_utilities.convert\_to\_date(cic.response\_target\_date) AS response\_target\_date ,

metric\_utilities.convert\_to\_date(cic.rfc\_date) AS rfc\_date ,

metric\_utilities.convert\_to\_date(cic.scheduled\_end\_date) AS scheduled\_end\_date ,

metric\_utilities.convert\_to\_date(cic.scheduled\_start\_date) AS scheduled\_start\_date ,

metric\_utilities.convert\_to\_date(cic.submit\_date)AS submit\_date

-- Calculated Dates

,

(SELECT metric\_utilities.convert\_to\_date(t10)

FROM aradmin.H1200@DTVREMP1.CISCO.COM h

WHERE h.entryid = cic.request\_id

) AS completed\_date\_h ,

(SELECT metric\_utilities.convert\_to\_date(t11)

FROM aradmin.H1200@DTVREMP1.CISCO.COM h

WHERE h.entryid = cic.request\_id

) AS closed\_date\_h ,

TO\_CHAR(

(SELECT metric\_utilities.convert\_to\_date(t10)

FROM aradmin.H1200@DTVREMP1.CISCO.COM h

WHERE h.entryid = cic.request\_id

) ,'MONTH') AS month\_completed\_h ,

TO\_CHAR(metric\_utilities.convert\_to\_date(cic.submit\_date), 'MONTH') AS month\_opened

-- Calculated Flags

,

ca\_set\_flag.flag AS crq\_created\_by\_INC ,

CASE

WHEN cic.change\_type = 2000

THEN 'Y'

ELSE 'N'

END AS chg\_type\_is\_change\_flag ,

CASE

WHEN cic.change\_type= 2000

AND change\_request\_status = 9

THEN 'Y'

ELSE 'N'

END AS chg\_is\_rejected\_flag ,

CASE

WHEN cic.change\_request\_status = 12

AND status\_reASon = 3000

THEN 'Y'

ELSE 'N'

END AS chg\_cancelled\_rescheduled\_flag ,

CASE

WHEN cic.change\_timing = 1000

THEN 'Y'

ELSE 'N'

END AS chg\_is\_emergency\_flag ,

CASE

WHEN cic.change\_request\_status = 11

AND status\_reASon = 8000

THEN 'Y'

ELSE 'N'

END AS chg\_is\_backed\_out\_flag ,

CASE

WHEN cic.change\_request\_status = 11

AND status\_reASon = 5000

THEN 'Y'

ELSE 'N'

END AS chg\_is\_successful\_flag ,

CASE

WHEN cic.change\_timing = 2000

THEN 'Y'

ELSE 'N'

END AS chg\_is\_expedited\_flag

-- Calculated Counts

,

NVL(count\_incidents.num\_inc\_FROM\_this\_CRQ,0) AS num\_inc\_FROM\_this\_CRQ

-- Calculated Times

,

(CASE

WHEN cic.change\_request\_status = 10

THEN metric\_utilities.convert\_to\_date(cic.completed\_date) - metric\_utilities.convert\_to\_date(cic.submit\_date)

END) AS time\_to\_complete\_days ,

'DTV'

FROM aradmin.chg\_infrAStructure\_change@DTVREMP1.CISCO.COM cic

-- count\_incidents created by this change

,

(SELECT cic.infrAStructure\_change\_id AS infrAStructure\_change\_id,

COUNT(ca.request\_id02) AS num\_inc\_FROM\_this\_CRQ

FROM aradmin.chg\_infrAStructure\_change@DTVREMP1.CISCO.COM cic,

aradmin.chg\_ASsociations@DTVREMP1.CISCO.COM ca

WHERE cic.infrAStructure\_change\_id = ca.request\_id02

AND ca.ASsociation\_type01 =30000

AND SUBSTR(ca.request\_id01,1,3) = 'INC'

GROUP BY cic.infrAStructure\_change\_id

) count\_incidents ,

(SELECT cic.infrAStructure\_change\_id AS infrAStructure\_change\_id,

MAX(ca\_flag.flag) flag

FROM aradmin.chg\_infrAStructure\_change@DTVREMP1.CISCO.COM cic,

(SELECT ca.request\_id02,

ca.request\_id01,

CASE

WHEN ca.ASsociation\_type01=31000

AND SUBSTR(ca.request\_id01,1,3) = 'INC'

THEN 'Y'

ELSE 'N'

END flag

FROM aradmin.chg\_ASsociations@DTVREMP1.CISCO.COM ca

) ca\_flag

WHERE cic.infrAStructure\_change\_id = ca\_flag.request\_id02

GROUP BY cic.infrAStructure\_change\_id

) ca\_set\_flag

WHERE cic.infrAStructure\_change\_id = ca\_set\_flag.infrAStructure\_change\_id(+)

AND cic.infrAStructure\_change\_id = count\_incidents.infrAStructure\_change\_id(+)

AND cic.infrAStructure\_change\_id IN

(SELECT DISTINCT(request\_id) FROM cm\_change\_id\_list

);

COMMIT;

o\_cm\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_cm\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL' ,'CM\_DTV\_GET\_DATA ERROR', SYSDATE, v\_msg);

END cm\_get\_data\_DTV;

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

-- Update the Change Management History Table

PROCEDURE cm\_update\_history\_dtv(

o\_cm\_status OUT VARCHAR)

IS

v\_msg VARCHAR2(1000);

BEGIN

metric\_utilities.log\_writer('CM','cm\_dtv\_update\_history\_data procedure start',SYSDATE,NULL);

DELETE

FROM cm\_history\_data

WHERE data\_source = 'DTV'

AND infrAStructure\_change\_id IN

(SELECT infrAStructure\_change\_id FROM cm\_stage\_data

);

COMMIT;

INSERT

INTO CM\_HISTORY\_DATA

(

-- ATOMIC FIELDS

INFRASTRUCTURE\_CHANGE\_ID ,

ASCHG ,

ASGRP ,

ASSIGNED\_TO ,

ASSIGNEE\_ID ,

ASSIGNEE\_ID\_ASSIGNEE ,

ASSIGNEE\_ID\_MANAGER ,

BUSINESS\_JUSTIFICATION ,

CATEGORIZATION\_TIER\_1 ,

CATEGORIZATION\_TIER\_2 ,

CATEGORIZATION\_TIER\_3 ,

CHANGE\_REQUEST\_STATUS ,

CHANGE\_REVIERWER\_NAME ,

CHANGE\_SUBTYPE ,

CHGIMP ,

CHGIMPGRP ,

CUSTOMER\_AREA\_CODE ,

CUSTOMER\_COMPANY ,

CUSTOMER\_CORPORATE\_ID ,

CUSTOMER\_DEPARTMENT ,

CUSTOMER\_EXTENSION ,

CUSTOMER\_FIRST\_NAME ,

CUSTOMER\_LAST\_NAME ,

CUSTOMER\_LOCAL\_PHONE ,

CUSTOMER\_ORGANIZATION ,

CUSTOMER\_PHONE\_NUMBER ,

DEPARTMENT ,

FIRST\_NAME ,

IMPACT ,

LAST\_MODIFIED\_BY ,

LAST\_NAME ,

LOCATION\_COMPANY ,

ORGANIZATION ,

OWNER\_GROUP ,

OWNER\_SUPPORT\_ORGANIZATION ,

PHONE\_NUMBER ,

PRODUCT\_CAT\_TIER\_1\_2\_ ,

PRODUCT\_CAT\_TIER\_2\_\_2\_ ,

PRODUCT\_CAT\_TIER\_3\_\_2\_ ,

PRODUCT\_CATEGORIZATION\_TIER\_1 ,

PRODUCT\_CATEGORIZATION\_TIER\_2 ,

PRODUCT\_CATEGORIZATION\_TIER\_3 ,

PRODUCT\_MODEL\_VERSION\_\_2\_ ,

PRODUCT\_NAME ,

PRODUCT\_NAME\_\_2\_ ,

REGION ,

REQUEST\_ID ,

REQUESTER\_CONTACTED ,

SITE ,

SITE\_GROUP ,

SITE\_ID ,

SUBMITTER ,

SUPPORT\_GROUP\_NAME ,

SUPPORT\_GROUP\_NAME2 ,

SUPPORT\_ORGANIZATION ,

SUPPORT\_ORGANIZATION2 ,

TOTAL\_TIME\_SPENT ,

URGENCY

-- Calculated Values

,

CHANGE\_TIMING\_VALUE ,

CUSTOMER\_EMAIL ,

DESCRIPTION ,

PRIORITY ,

RISK\_LEVEL ,

STATUS ,

STATUS\_REASON

-- Atomic Dates

,

ACKNOWLEDGMENT\_END\_DATE ,

ACTUAL\_END\_DATE ,

ACTUAL\_START\_DATE ,

COMPLETED\_DATE ,

EARLIEST\_START\_DATE ,

LAST\_MODIFIED\_DATE ,

NEXT\_TARGET\_DATE ,

REQUESTED\_END\_DATE ,

REQUESTED\_START\_DATE ,

RESOLUTION\_END\_DATE ,

RESPONSE\_TARGET\_DATE ,

RFC\_DATE ,

SCHEDULED\_END\_DATE ,

SCHEDULED\_START\_DATE ,

SUBMIT\_DATE

-- Calculated Dates

,

COMPLETED\_DATE\_H ,

CLOSED\_DATE\_H ,

MONTH\_COMPLETED\_H ,

MONTH\_OPENED

-- Calculated Flags

,

CRQ\_CREATED\_BY\_INC ,

CHG\_TYPE\_IS\_CHANGE\_FLAG ,

CHG\_IS\_REJECTED\_FLAG ,

CHG\_CANCELLED\_RESCHEDULED\_FLAG ,

CHG\_IS\_EMERGENCY\_FLAG ,

CHG\_IS\_BACKED\_OUT\_FLAG ,

CHG\_IS\_SUCCESSFUL\_FLAG ,

CHG\_IS\_EXPEDITED\_FLAG

-- Calculated Counts

,

NUM\_INC\_FROM\_THIS\_CRQ

-- Calculated Times

,

TIME\_TO\_COMPLETE\_DAYS

-- Data Source

,

DATA\_SOURCE

)

SELECT

-- ATOMIC FIELDS

INFRASTRUCTURE\_CHANGE\_ID ,

ASCHG ,

ASGRP ,

ASSIGNED\_TO ,

ASSIGNEE\_ID ,

ASSIGNEE\_ID\_ASSIGNEE ,

ASSIGNEE\_ID\_MANAGER ,

BUSINESS\_JUSTIFICATION ,

CATEGORIZATION\_TIER\_1 ,

CATEGORIZATION\_TIER\_2 ,

CATEGORIZATION\_TIER\_3 ,

CHANGE\_REQUEST\_STATUS ,

CHANGE\_REVIERWER\_NAME ,

CHANGE\_SUBTYPE ,

CHGIMP ,

CHGIMPGRP ,

CUSTOMER\_AREA\_CODE ,

CUSTOMER\_COMPANY ,

CUSTOMER\_CORPORATE\_ID ,

CUSTOMER\_DEPARTMENT ,

CUSTOMER\_EXTENSION ,

CUSTOMER\_FIRST\_NAME ,

CUSTOMER\_LAST\_NAME ,

CUSTOMER\_LOCAL\_PHONE ,

CUSTOMER\_ORGANIZATION ,

CUSTOMER\_PHONE\_NUMBER ,

DEPARTMENT ,

FIRST\_NAME ,

IMPACT ,

LAST\_MODIFIED\_BY ,

LAST\_NAME ,

LOCATION\_COMPANY ,

ORGANIZATION ,

OWNER\_GROUP ,

OWNER\_SUPPORT\_ORGANIZATION ,

PHONE\_NUMBER ,

PRODUCT\_CAT\_TIER\_1\_2\_ ,

PRODUCT\_CAT\_TIER\_2\_\_2\_ ,

PRODUCT\_CAT\_TIER\_3\_\_2\_ ,

PRODUCT\_CATEGORIZATION\_TIER\_1 ,

PRODUCT\_CATEGORIZATION\_TIER\_2 ,

PRODUCT\_CATEGORIZATION\_TIER\_3 ,

PRODUCT\_MODEL\_VERSION\_\_2\_ ,

PRODUCT\_NAME ,

PRODUCT\_NAME\_\_2\_ ,

REGION ,

REQUEST\_ID ,

REQUESTER\_CONTACTED ,

SITE ,

SITE\_GROUP ,

SITE\_ID ,

SUBMITTER ,

SUPPORT\_GROUP\_NAME ,

SUPPORT\_GROUP\_NAME2 ,

SUPPORT\_ORGANIZATION ,

SUPPORT\_ORGANIZATION2 ,

TOTAL\_TIME\_SPENT ,

URGENCY

-- Calculated Values

,

CHANGE\_TIMING\_VALUE ,

CUSTOMER\_EMAIL ,

DESCRIPTION ,

PRIORITY ,

RISK\_LEVEL ,

STATUS ,

STATUS\_REASON

-- Atomic Dates

,

ACKNOWLEDGMENT\_END\_DATE ,

ACTUAL\_END\_DATE ,

ACTUAL\_START\_DATE ,

COMPLETED\_DATE ,

EARLIEST\_START\_DATE ,

LAST\_MODIFIED\_DATE ,

NEXT\_TARGET\_DATE ,

REQUESTED\_END\_DATE ,

REQUESTED\_START\_DATE ,

RESOLUTION\_END\_DATE ,

RESPONSE\_TARGET\_DATE ,

RFC\_DATE ,

SCHEDULED\_END\_DATE ,

SCHEDULED\_START\_DATE ,

SUBMIT\_DATE

-- Calculated Dates

,

COMPLETED\_DATE\_H ,

CLOSED\_DATE\_H ,

MONTH\_COMPLETED\_H ,

MONTH\_OPENED

-- Calculated Flags

,

CRQ\_CREATED\_BY\_INC ,

CHG\_TYPE\_IS\_CHANGE\_FLAG ,

CHG\_IS\_REJECTED\_FLAG ,

CHG\_CANCELLED\_RESCHEDULED\_FLAG ,

CHG\_IS\_EMERGENCY\_FLAG ,

CHG\_IS\_BACKED\_OUT\_FLAG ,

CHG\_IS\_SUCCESSFUL\_FLAG ,

CHG\_IS\_EXPEDITED\_FLAG

-- Calculated Counts

,

NUM\_INC\_FROM\_THIS\_CRQ

-- Calculated Times

,

TIME\_TO\_COMPLETE\_DAYS

-- Data Source

,

DATA\_SOURCE

FROM cm\_stage\_data;

COMMIT;

metric\_utilities.log\_writer('CM','cm\_dtv\_update\_history\_data procedure end',SYSDATE,NULL);

o\_cm\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_cm\_status := 'FAILURE';

metric\_utilities.log\_writer('CM','cm\_dtv\_get\_stage\_data procedure ERROR',SYSDATE, v\_msg);

END cm\_update\_history\_dtv;

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

-- Rolls up Change Managment data for reporting

PROCEDURE cm\_rollup\_data\_dtv(

o\_cm\_status OUT VARCHAR)

IS

v\_msg VARCHAR2(1000);

v\_status VARCHAR2(15);

BEGIN

metric\_utilities.refresh\_snapshot( 'CM\_ROLLUP\_OPENCOUNTSVW', v\_status );

metric\_utilities.refresh\_snapshot( 'CM\_ROLLUP\_CLOSEDCOUNTSVW', v\_status );

metric\_utilities.refresh\_snapshot( 'CM\_ROLLUP\_OPENCOUNTSBYEMPVW',v\_status );

metric\_utilities.refresh\_snapshot( 'CM\_ROLLUP\_CLOSEDCOUNTSBYEMPVW', v\_status );

o\_cm\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

v\_msg := SQLERRM;

o\_cm\_status := 'FAILURE';

metric\_utilities.log\_writer('ALL' ,'CM\_ROLLUP\_DATA ERROR', SYSDATE, v\_msg);

END cm\_rollup\_data\_dtv;

END metric\_rollups\_dtv;

/