CCES Oracle Data Pump

History seeding data selection requirements

Export from on-premise ODS

- Select All PAID claim versions for the last 3 years in a single Oracle DataPump export run, plus make sure we exclude “bad/unlinked" claim version based on list of claim IDs provided by Aetna (via sql loader).
- Pulling all paid versions results in additional records but the number of extras records is manageable plus the query was more efficient
    - Number of paid claims **without** using current_flag is 593,265,306 records (took 11 hours to run, 40M extra or 6.8% of the total)
    - Number of paid claims **with** current_flag = Y is 553,495,546 records (took 12 hours to run)
- Target only required claim tables, a full schema export is not required
- Provide Aetna with export instructions and Oracle DataPump parameter file
- The plan is to tun this process over several iterations perhaps once for each year
    - Use a from and to date (range currently commented out)
    - Truncate CCES_CLAIM_VERSION before each run

Import into cloud ODS

- Post import, set CURRENT_FLAG=Y for paid claims that have CURRENT_FLAG=N for all its versions
- Make this update on the claim record where IC_CLAIM_VERSION_ID is the highest (meaning latest since PK is based on Oracle sequence)

Export Steps

(1) Update the BATCH table to exclude seeded claims from the ongoing/daily CCES paid feed

- Claims that are exported as part of initial data seeding **must be excluded** from future exports done via the new CCES batch export task.
- The batch export task targets batch records with a BATCH.CCES_EXPORT_STATUS **of NOT NULL**. We will update this column to a non-null value (ie: "Seeded") just prior to the data pump export.
- An outage is required to make this update post CXT 6.1.1/CCES upgrade and just prior to data pump export
- This step is dependent on the ODS being upgraded to 6.1.1/CCES
    - If you are **testing** with a cloned 5x schema, just add the column to the BATCH table manually CCES_EXPORTSTUS, VARCHAR2(255 BYTE)

```
UPDATE BATCH
SET CCES_EXPORTSTATUS = 'SEEDED'
WHERE WORKFLOW        = 'PAID'
AND (STATE            = 'COMPLETED'
OR STATE              = 'CLOSED')
```

(2) Create objects

```
CREATE TABLE CCES_CLAIM_VERSION
  (
    "IC_CLAIM_VERSION_ID" NUMBER(18,0) NOT NULL,
    "IC_BATCH_ID"         NUMBER(18,0) NOT NULL,
    "IC_CLAIM_ID"         NUMBER(18,0) NOT NULL,
    "CLAIM_ID"            VARCHAR2(42 BYTE) NOT NULL
  );

CREATE TABLE CCES_CLAIM_ID_EXCLUDED
  ( "CLAIM_ID" VARCHAR2(42 BYTE) NOT NULL
  );

CREATE INDEX cces_claim_id_indx ON cces_claim_version(claim_id);
CREATE INDEX cces_ic_claim_version_id_indx ON
cces_claim_version(ic_claim_version_id);
CREATE INDEX cces_ic_batch_id_indx ON cces_claim_version(ic_batch_id);
CREATE INDEX cces_ic_claim_id_indx ON cces_claim_version(ic_claim_id);
CREATE OR REPLACE PROCEDURE PopulatDrivingTable IS
CURSOR c_cur IS
  SELECT CV.IC_CLAIM_VERSION_ID, CV.IC_BATCH_ID, CV.IC_CLAIM_ID,
CV.CLAIM_ID
  FROM CLAIM_VERSION CV, BATCH B
  WHERE CV.IC_PURGE_DATE >= TO_TIMESTAMP('02/01/2014','mm/dd/yyyy')  --
FROM_DATE
  AND CV.IC_PURGE_DATE <  TO_TIMESTAMP('02/01/2016','mm/dd/yyyy')    --
TO_DATE
    AND CV.VERSION_NAME     = 'PAID'
    AND B.IC_BATCH_ID       = CV.IC_BATCH_ID
    AND B.CCES_EXPORTSTATUS = 'SEEDED'
    AND NOT EXISTS (SELECT null FROM CCES_CLAIM_ID_EXCLUDED EX WHERE
EX.CLAIM_ID = CV.CLAIM_ID);
  --
  TYPE table_info IS  TABLE OF c_cur%ROWTYPE INDEX BY PLS_INTEGER;
  l_table_info table_info;
BEGIN
  OPEN c_cur;
   LOOP
     FETCH c_cur  BULK COLLECT INTO l_table_info LIMIT 1000;
     EXIT WHEN  l_table_info.COUNT = 0;
     FOR indx IN 1 .. l_table_info.COUNT  LOOP
    INSERT INTO CCES_CLAIM_VERSION values l_table_info(indx);
     END LOOP;
  commit;
   END LOOP;
  CLOSE c_cur;
  commit;
END PopulatDrivingTable;
/
show err
```

(3) Populate CCES_CLAIM_ID_EXCLUDED table using SQL Loader **(Aetna to provide actual data)**

Command

```
sqlldr userid=[database_connect] control=CCES_CLAIM_ID_EXCLUDED.ctl
data=claims.txt

Sample:
sqlldr
userid='AETNA_HE_TPP1/iclaim@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=N
DHDFIAT1.mckesson.com)(PORT=1521))(CONNECT_DATA=(SERVER=DEDICATED)(SERVI
CE_NAME=TRRNLAAE_GEN)))' control='c:\loader\CCES_CLAIM_ID_EXCLUDED.ctl'
data='c:\loader\claim.txt'
```

Control File

```
load data
infile 'claims.txt'
truncate
into table CCES_CLAIM_ID_EXCLUDED
(claim_id char(42))
```

Sample Data File

```
10
11
12
13
14
15
```

(4) Execute "PopulatDrivingTable" procedure to populate the CCES_CLAIM_VERSION table

- Double check the date range and be sure if is correct. Plan it to export one year at a time, start with year 1.

(5) Run data pump export

```
DIRECTORY=DPDIR
PARALLEL=4
FILESIZE=2GB
DUMPFILE=<FILENAME>_%U.dmp
CONTENT=DATA_ONLY
LOGFILE<FILENAME>.log
EXCLUDE=STATISTICS
TABLES=<schema_name>.BATCH,<schema_name>.CLAIM,<schema_name>.CLAIM_VERSI
ON,<schema_name>.CLAIM_LINE, <schema_name>.MEMBER,
<schema_name>.CLAIM_SET, <schema_name>.CLAIM_SET_CLAIM_VERSION
QUERY=(<schema_name>.BATCH:"where iC_BATCH_ID in (SELECT DISTINCT
IC_BATCH_ID FROM <schema_name>.CCES_CLAIM_VERSION)")
QUERY=(<schema_name>.CLAIM_VERSION:"where IC_CLAIM_VERSION_ID in (SELECT
IC_CLAIM_VERSION_ID FROM <schema_name>.CCES_CLAIM_VERSION)")
QUERY=(<schema_name>.CLAIM_LINE:"where IC_CLAIM_VERSION_ID in (SELECT
IC_CLAIM_VERSION_ID FROM <schema_name>.CCES_CLAIM_VERSION)")
QUERY=(<schema_name>.CLAIM:"where IC_CLAIM_ID in (SELECT IC_CLAIM_ID
FROM <schema_name>.CCES_CLAIM_VERSION)")
QUERY=(<schema_name>.CLAIM_SET:"where IC_CLAIM_ID in (SELECT DISTINCT
IC_BATCH_ID FROM <schema_name>.CCES_CLAIM_VERSION)")
QUERY=(<schema_name>.CLAIM_SET_CLAIM_VERSION:"where IC_CLAIM_VERSION_ID
in (SELECT IC_CLAIM_VERSION_ID FROM <schema_name>.CCES_CLAIM_VERSION)")
```

(6) Repeat the export process for years 2 and 3

- Repeat the process for year 2
  - Truncate the CCES_CLAIM_VERSION
  - Alter the "PopulatDrivingTable" procedure, changing the to/from dates
  - Execute "PopulatDrivingTable" procedure
  - Run data pump export
- Repeat the process for year 3
  - Truncate the CCES_CLAIM_VERSION
  - Alter the "PopulatDrivingTable" procedure, changing the to/from dates
  - Execute "PopulatDrivingTable" procedure
  - Run data pump export

(7) Encrypt and compress the export files

- Use PGP or GPG if encryption is required
- If removable media/drive is already encrypted, just compress using zip, no PGP or GPG is required

Import Steps

(1) Disable all constraints in **target** schema (CLAIM_SET_CLAIM_VERSION intentionally left out since it is a referenced constraint)

```
BEGIN
  FOR con IN
  (
  SELECT table_name, constraint_name, status FROM user_constraints
  WHERE table_name IN
('BATCH','CLAIM','CLAIM_LINE','CLAIM_VERSION','CLAIM_SET','MEMBER','BATC
H')
  and constraint_type = 'R' and status = 'ENABLED' -- R is FK
  )
  LOOP
    EXECUTE immediate 'alter table '||con.table_name||' disable
constraint  '||con.constraint_name||'';
  END LOOP;
END;
/
```

(2) Runs data pump import

(3) Enable constraints

```
BEGIN
  FOR con IN
  (
  SELECT table_name, constraint_name FROM user_constraints
  WHERE table_name IN
('BATCH','CLAIM','CLAIM_LINE','CLAIM_VERSION','MEMBER','BATCH','CLAIM_SE
T')
  and constraint_type = 'R' and status = 'DISABLED'
  )
  LOOP
    EXECUTE immediate 'alter table '||con.table_name||' enable
constraint  '||con.constraint_name||'';
  END LOOP;
END;
/
```

(4) Repair paid claim data (setting of current flag = Y where needed)

- Create and execute the "UpdateCurrentFlag" procedure

```
CREATE OR REPLACE PROCEDURE UpdateCurrentFlag IS
  CURSOR c_cur IS
  SELECT max(v1.ic_claim_version_id)
  FROM claim_version v1
  WHERE v1.current_flag <> 'Y'
    AND not exists (SELECT null
                 FROM claim_version v2
     WHERE v2.ic_claim_id = v1.ic_claim_id
       AND v2.current_flag = 'Y');

  l_cnt  number := 0;
  l_ic_claim_version_id  number;
BEGIN
  OPEN c_cur;
   LOOP
     FETCH c_cur  INTO l_ic_claim_version_id;
     EXIT WHEN  c_cur%NOTFOUND;
     l_cnt := l_cnt + 1;
  UPDATE claim_version
  SET current_flag = 'Y'
  WHERE ic_claim_version_id = l_ic_claim_version_id;
  IF l_cnt = 1000 THEN
    commit;
    l_cnt := l_cnt + 1;
  END IF;
   END LOOP;
  CLOSE c_cur;
  commit;
END UpdateCurrentFlag;
/
show err
```

(5) Reset Sequences

```
DECLARE
  maxval          INT;
  seqval          INT;
  l_seq_val       INTEGER;
  l_seq_val_after INTEGER;
  l_seq_val_final INTEGER;
BEGIN
  SELECT MAX(BATCH.IC_BATCH_ID) INTO maxval FROM BATCH WHERE
BATCH.IC_BATCH_ID <> 9999999991;
  l_seq_val := IC_BATCH_ID_SEQ.nextval;
  dbms_output.put_line('IC_BATCH_ID_SEQ Max Table Value: '||maxval||',
Current Seq Value: '||l_seq_val);
  EXECUTE immediate 'alter sequence IC_BATCH_ID_SEQ increment by '||
maxval;
  l_seq_val_after := IC_BATCH_ID_SEQ.nextval;
  dbms_output.put_line('IC_BATCH_ID_SEQ After Increment:
'||l_seq_val_after);
  EXECUTE immediate 'alter sequence IC_BATCH_ID_SEQ increment by 1 ';
  l_seq_val_final := IC_BATCH_ID_SEQ.nextval;
  dbms_output.put_line('IC_BATCH_ID_SEQ New Sequence:
'||l_seq_val_final);
  --
  SELECT MAX(CLAIM.IC_CLAIM_ID) INTO maxval FROM CLAIM;
  l_seq_val := IC_CLAIM_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_ID_SEQ Max Table Value: '||maxval||',
Current Seq Value: '||l_seq_val);
  EXECUTE immediate 'alter sequence IC_CLAIM_ID_SEQ increment by '||
maxval;
  l_seq_val_after := IC_CLAIM_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_ID_SEQ After Increment:
'||l_seq_val_after);
  EXECUTE immediate 'alter sequence IC_CLAIM_ID_SEQ increment by 1 ';
  l_seq_val_final := IC_CLAIM_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_ID_SEQ New Sequence:
'||l_seq_val_final);
  --
  SELECT MAX(CLAIM_VERSION.IC_CLAIM_VERSION_ID) INTO maxval FROM
CLAIM_VERSION;
  l_seq_val := IC_CLAIM_VERSION_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_VERSION_ID_SEQ Max Table Value:
'||maxval||', Current Seq Value: '||l_seq_val);
  EXECUTE immediate 'alter sequence IC_CLAIM_VERSION_ID_SEQ increment by
'|| maxval;
  l_seq_val_after := IC_CLAIM_VERSION_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_VERSION_ID_SEQ After Increment:
'||l_seq_val_after);
  EXECUTE immediate 'alter sequence IC_CLAIM_VERSION_ID_SEQ increment by
1 ';
  l_seq_val_final := IC_CLAIM_VERSION_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_VERSION_ID_SEQ New Sequence:
'||l_seq_val_final);
  --
  SELECT MAX(CLAIM_LINE.IC_CLAIM_LINE_ID) INTO maxval FROM CLAIM_LINE;
  l_seq_val := IC_CLAIM_LINE_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_LINE_ID_SEQ Max Table Value:
'||maxval||', Current Seq Value: '||l_seq_val);
  EXECUTE immediate 'alter sequence IC_CLAIM_LINE_ID_SEQ increment by
'|| maxval;
  l_seq_val_after := IC_CLAIM_LINE_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_LINE_ID_SEQ After Increment:
'||l_seq_val_after);
  EXECUTE immediate 'alter sequence IC_CLAIM_LINE_ID_SEQ increment by 1
';
```

```
  l_seq_val_final := IC_CLAIM_LINE_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_LINE_ID_SEQ New Sequence:
'||l_seq_val_final);
  --
  SELECT MAX(MEMBER.IC_MEMBER_ID) INTO maxval FROM MEMBER;
  l_seq_val := IC_MEMBER_ID_SEQ.nextval;
  dbms_output.put_line('IC_MEMBER_ID_SEQ Max Table Value: '||maxval||',
Current Seq Value: '||l_seq_val);
  EXECUTE immediate 'alter sequence IC_MEMBER_ID_SEQ increment by '||
maxval;
  l_seq_val_after := IC_MEMBER_ID_SEQ.nextval;
  dbms_output.put_line('IC_MEMBER_ID_SEQ After Increment:
'||l_seq_val_after);
  EXECUTE immediate 'alter sequence IC_MEMBER_ID_SEQ increment by 1 ';
  l_seq_val_final := IC_MEMBER_ID_SEQ.nextval;
  dbms_output.put_line('IC_MEMBER_ID_SEQ New Sequence:
'||l_seq_val_final);
  --
  SELECT MAX(CLAIM_SET.IC_CLAIM_SET_ID) INTO maxval FROM CLAIM_SET;
  l_seq_val := IC_CLAIM_SET_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_SET_ID_SEQ Max Table Value:
'||maxval||', Current Seq Value: '||l_seq_val);
  EXECUTE immediate 'alter sequence IC_CLAIM_SET_ID_SEQ increment by '||
maxval;
  l_seq_val_after := IC_CLAIM_SET_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_SET_ID_SEQ After Increment:
'||l_seq_val_after);
  EXECUTE immediate 'alter sequence IC_CLAIM_SET_ID_SEQ increment by 1
';
  l_seq_val_final := IC_CLAIM_SET_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLAIM_SET_ID_SEQ New Sequence:
'||l_seq_val_final);
  --
  SELECT MAX(CLAIM_SET_CLAIM_VERSION.IC_CLAIM_SET_CLAIM_VERSION_ID) INTO
maxval FROM CLAIM_SET_CLAIM_VERSION;
  l_seq_val := IC_CLM_SET_CLM_VSN_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLM_SET_CLM_VSN_ID_SEQ Max Table Value:
'||maxval||', Current Seq Value: '||l_seq_val);
  EXECUTE immediate 'alter sequence IC_CLM_SET_CLM_VSN_ID_SEQ increment
by '|| maxval;
  l_seq_val_after := IC_CLM_SET_CLM_VSN_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLM_SET_CLM_VSN_ID_SEQ After Increment:
'||l_seq_val_after);
  EXECUTE immediate 'alter sequence IC_CLM_SET_CLM_VSN_ID_SEQ increment
by 1 ';
  l_seq_val_final := IC_CLM_SET_CLM_VSN_ID_SEQ.nextval;
  dbms_output.put_line('IC_CLM_SET_CLM_VSN_ID_SEQ New Sequence:
'||l_seq_val_final);
```

```
END;
/
```

(6) Collect statistics