eSUG & CSUG

Rory Pittman

October 7th 2015 (eSUG)

October 8th 2015 (CSUG)

ATB Financial™

Example

- You have to pull 1 year of transaction data for an internal executive client ASAP.
- Monthly sandbox summary tables do not contain required information
- Determine what is needed is to summarize transactions by total sum and count of transactions per account per transaction type per month
- Data has to be pulled from two separates tables, one with recent data and another from an archive schema
- Variants of similar requests from other lines of business within in the next couple of weeks is highly probable, so want to write optimized code to pull data as quickly as possible
- TIMING FOR REQUEST IS AN ISSUE!

/* Data Step Method */

```
data Trxns;
     set archive.SOC_ACCOUNT_DET_TRXN_V (where = ('01SEP2014'd <= CREATION_DT <= '31MAR2015'd))
         schema.SOC_ACCOUNT_DET_TRXN_V (where = ('01APR2015'd <= CREATION_DT <= '31AUG2015'd));
     YEAR = year(CREATION_DT);
     MONTH = month(CREATION_DT);
     keep ACCT_OID TRXN_TYP_FK YEAR MONTH TRXN_AMT;
run;

proc sort data = Trxns; by ACCT_OID TRXN_TYP_FK YEAR MONTH; run;

proc summary data = Trxns;
     var TRXN_AMT;
     by ACCT_OID TRXN_TYP_FK YEAR MONTH;
     output out=Trxns_Summary (drop = _TYPE_ _FREQ_) n=TRXN_COUNT sum=TRXN_SUM;
run;
```

/* Data Step Execution Time */

```
NOTE: There were 258322493 observations read from the data set ARCHIVE.SOC_ACCOUNT_DET_TRXN_V.
      WHERE (CREATION_DT>='01SEP2014'D and CREATION_DT<='31MAR2015'D);
NOTE: There were 226288036 observations read from the data set SCHEMA.SOC_ACCOUNT_DET_TRXN_V.
      WHERE (CREATION_DT>='01APR2015'D and CREATION_DT<='31AUG2015'D);
NOTE: The data set WORK.TRXNS has 484610529 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           4:53:28.83
      cpu time            3:35:25.70

NOTE: There were 484610529 observations read from the data set WORK.TRXNS.
NOTE: The data set WORK.TRXNS has 484610529 observations and 5 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           5:03.88
      cpu time            8:35.81

NOTE: There were 484610529 observations read from the data set WORK.TRXNS.
NOTE: The data set WORK.TRXNS_SUMMARY has 55462958 observations and 6 variables.
NOTE: PROCEDURE SUMMARY used (Total process time):
      real time           5:25.21
      cpu time            5:21.50
```

/* Hash Table Method */

```
data _null_;
     if _n_ = 0 then set archive.SOC_ACCOUNT_DET_TRXN_V schema.SOC_ACCOUNT_DET_TRXN_V;
     if _n_ = 1 then do;
        declare hash a(dataset: "archive.SOC_ACCOUNT_DET_TRXN_V(where = ('01SEP2014'd <= CREATION_DT <= '31MAR2015'd)",
                       ordered:'yes');
          a.definekey('ACCT_DET_TRXN_ROW_ID');
          a.definedata('ACCT_OID','TRXN_TYP_FK','CREATION_DT','TRXN_AMT');
          a.definedone();              call missing(ACCT_OID,TRXN_TYP_FK,CREATION_DT,TRXN_AMT);
        declare hash s(dataset: "schema.SOC_ACCOUNT_DET_TRXN_V(where = ('01APR2015'd <= CREATION_DT <= '31AUG2015'd)",
                       ordered:'yes');
          s.definekey('ACCT_DET_TRXN_ROW_ID');
          s.definedata('ACCT_OID','TRXN_TYP_FK','CREATION_DT','TRXN_AMT');
          s.definedone();              call missing(ACCT_OID,TRXN_TYP_FK,CREATION_DT,TRXN_AMT);
        end;
        a.add ();    s.add ();     a.output(dataset: 'Trxns_ARC');   s.output(dataset: 'Trxns_SOC');
run;

data Trxns; set Trxns_ARC Trxns_SOC; where ACCT_OID is not null; YEAR = year(CREATION_DT); MONTH = month(CREATION_DT);
run;

proc sort data = Trxns; by ACCT_OID TRXN_TYP_FK YEAR MONTH; run;

proc summary data = Trxns;
     var TRXN_AMT;
     by ACCT_OID TRXN_TYP_FK YEAR MONTH;
        output out=Trxns_Summary (drop = _TYPE_ _FREQ_) n=TRXN_COUNT sum=TRXN_SUM;
run;
```

/* Hash Table Execution Time */

```
NOTE: There were 258322493 observations read from the data set ARCHIVE.SOC_ACCOUNT_DET_TRXN_V.
      WHERE (CREATION_DT>='01SEP2014'D and CREATION_DT<='31MAR2015'D);
NOTE: There were 226288036 observations read from the data set SCHEMA.SOC_ACCOUNT_DET_TRXN_V.
      WHERE (CREATION_DT>='01APR2015'D and CREATION_DT<='31AUG2015'D);
NOTE: DATA statement used (Total process time):
      real time           4:55:06.37
      cpu time            3:52:19.11

NOTE: The data set WORK.TRXNS has 484610529 observations and 6 variables.
NOTE: DATA statement used (Total process time):
      real time           5:06.59
      cpu time            4:51.46

NOTE: The data set WORK.TRXNS has 484610529 observations and 6 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           4:46.83
      cpu time            9:06.82


NOTE: The data set WORK.TRXNS_SUMMARY has 55462958 observations and 6 variables.
NOTE: PROCEDURE SUMMARY used (Total process time):
      real time           5:37.29
      cpu time            5:34.43
```

/* Proc SQL Method */

```
proc sql;
     create table Trxns_Summary as

     select distinct ACCT_OID, TRXN_TYP_FK, year(CREATION_DT) as YEAR, month(CREATION_DT) as MONTH,
            count(TRXN_AMT) as TRXN_COUNT, sum(TRXN_AMT) as TRXN_SUM
     from archive.SOC_ACCOUNT_DET_TRXN_V
     where '01SEP2014'd <= CREATION_DT <= '31MAR2015'd

     OUTER UNION CORR

     select distinct ACCT_OID, TRXN_TYP_FK, year(CREATION_DT) as YEAR, month(CREATION_DT) as MONTH,
            count(TRXN_AMT) as TRXN_COUNT, sum(TRXN_AMT) as TRXN_SUM
     from schema.SOC_ACCOUNT_DET_TRXN_V
     where '01APR2015'd <= CREATION_DT <= '31AUG2015'd

     group by ACCT_OID, TRXN_TYP_FK, calculated YEAR, calculated MONTH;
quit;
```

/* Proc SQL Execution Time */

- NOTE: The query requires remerging summary statistics back with the original data.
- NOTE: Table WORK.TRXNS_SUMMARY created, with 55462958 rows and 6 columns.

- NOTE: PROCEDURE SQL used (Total process time):
- real time 55:32.75
- cpu time 17:42.28

/* DB2 Pass-Through Example */

```
proc sql;
      connect to db2 (user=_______ database=_______ password=_______);

      execute(      CREATE TABLE "Z_SCHEMA"."TRXNS" AS (
                    SELECT ACCT_OID, TRXN_TYP_FK, YEAR(CREATION_DT) AS YEAR, MONTH(CREATION_DT) AS MONTH,
                          COUNT(TRXN_AMT) AS TRXN_COUNT, SUM(TRXN_AMT) AS TRXN_SUM
                    FROM "SCHEMA"."SOC_ACCOUNT_DET_TRXN_V"
                    GROUP BY ACCT_OID, TRXN_TYP_FK, CREATION_DT, TRXN_AMT) WITH NO DATA       ) by db2;

      execute(      INSERT INTO "Z_SCHEMA"."TRXNS" (ACCT_OID, TRXN_TYP_FK, YEAR, MONTH, TRXN_COUNT, TRXN_SUM)
                    SELECT ACCT_OID, TRXN_TYP_FK, YEAR(CREATION_DT), MONTH(CREATION_DT), COUNT(TRXN_AMT), SUM(TRXN_AMT)
                    FROM "ARCHIVE"."SOC_ACCOUNT_DET_TRXN_V"
                    WHERE CREATION_DT BETWEEN DATE '2014-09-01' AND DATE '2015-03-31'
                    GROUP BY ACCT_OID, TRXN_TYP_FK, YEAR(CREATION_DT), MONTH(CREATION_DT)
                    ORDER BY ACCT_OID, TRXN_TYP_FK , YEAR(CREATION_DT), MONTH(CREATION_DT)    ) by db2;

      execute(      INSERT INTO "Z_SCHEMA"."TRXNS" (ACCT_OID, TRXN_TYP_FK, YEAR, MONTH, TRXN_COUNT, TRXN_SUM)
                    SELECT ACCT_OID, TRXN_TYP_FK, YEAR(CREATION_DT), MONTH(CREATION_DT), COUNT(TRXN_AMT), SUM(TRXN_AMT)
                    FROM "SCHEMA"."SOC_ACCOUNT_DET_TRXN_V"
                    WHERE CREATION_DT BETWEEN DATE '2015-04-01' AND DATE '2015-08-31'
                    GROUP BY ACCT_OID, TRXN_TYP_FK, YEAR(CREATION_DT), MONTH(CREATION_DT)
                    ORDER BY ACCT_OID, TRXN_TYP_FK , YEAR(CREATION_DT), MONTH(CREATION_DT)    ) by db2;

      execute(      GRANT SELECT ON TABLE "Z_SCHEMA"."TRXNS" TO PUBLIC    ) by db2;

      disconnect from db2;
quit;
```

/* DB2 Pass-Through Execution Time */

```
• NOTE: PROCEDURE SQL used (Total process time):
•       real time           49:07.67
•       cpu time            0.01 seconds
•
```

```
proc sql;
     create table Trxns as
     select *
     from Z_schema.TRXNS;
quit;
```

```
• NOTE: Table WORK.TRXNS created, with 55462958 rows and 6 columns.
• NOTE: PROCEDURE SQL used (Total process time):
•       real time           2:37.82
•       cpu time            1:32.62
```

SQL Pass-Through Facility Structure

```
proc sql;
     connect to db2 (user=_______ database=_______ password=_______);

     execute( SQL Code for Database ) by db2;

     execute( SQL Code for Database ) by db2;

     disconnect from db2;
quit;
```

For connecting to Microsoft SQL Server, would use:

```
connect to sqlsvr (datasrc=_______ user=_______ password=_______);
```

For connecting to Oracle, would use:

```
connect to oracle (user=_______ password=_______ path=_______);
```

For connecting to other supported types of databases using SAS SQL Pass-Through facility, visit:
https://support.sas.com/documentation/cdl/en/acreldb/63647/HTML/default/viewer.htm#acreldbwhatsnew902.htm

Concluding Remarks

- SQL pass-through facility works well with pulling vast amounts of data from a data warehouse
- For general use, Proc SQL is probably the best method to use by default

- For large tables, improvements in data retrieval speed could be materialized if large tables are structured to be column-organized instead of row-organized (if pulling millions of records but yet only require a few columns)
- Indexing data warehouse tables based on unique identifiers will greatly speed up data pulls
- Speed of data pull will depend on how data warehouse is connected and configured to SAS Server

- Using a macro do loop to pull one month of data at a time could improve data retrieval speed
- Load times and usage of SAS Server and data warehouse can effect retrieval times, so may be optimal to run large data pull scripts during evening or early morning hours