Paper 221-29

DB2, SAS®, and You

Robert Ellsworth, Ellsworth Stewart Consulting Inc.
83 Bryant Road, Markham, Ontario, L3P 5Y8

ABSTRACT

Access DB2 using SAS, methods to get the best results for database access. We will cover tips and tricks to get the most out of SAS Access for DB2. The topography of your system dictates how your programs will perform. We look at many methods to access and process your data highlighting pros and cons.

INTRODUCTION

What is the best way to access your data? Let's look at several techniques to get the best results from DB2. Always use the power of the database engine. The database server is usually the biggest box available to your program. Database structures like case statements and key table can significantly improve query performance. Saving calculated results to use again and again cuts program execution time. Finally how can you use libname and views.

USING THE POWER OF THE DATABASE ENGINE

The topography of your system dictates how your programs will perform. The typical installation has a large multi-processor database server. The SAS server is generally smaller box. The SAS server is connected through a high-speed link to the database. Workstations are connected to the SAS server with a lower speed link. The database server has lots and lots of data proportional to its size.

So when looking to maximize performance you need to consider the topography. Moving large amount of data takes a lot of time even on a high-speed connection. The Database server has more horsepower than the SAS server or your workstation. Leading to the conclusion that you should do as much as possible on the Database server.

Use the power of the Database server. Let the Database do the heavy lifting.

Here is a simple example to illustrate.

USING THE SAS SERVER

```
Create table bal_dset select * from connection to db2(
   select balance,
          dlinq_ind
     from visa_acct);
Proc sort;
    by dlinq_ind;
Proc summary;
    var balance;
    by dlinq_ind;
    output out=result sum= n=cnt;
```

USING THE DATABASE

```
select * from connection to db2(
   select count(*) as cnt,
          sum(balance) as balance,
          dlinq_ind
     from visa_acct
    group by dlinq_ind);
```

The database already has the data organized to sum. Even if the SAS server was as fast as the database server transferring the data to the SAS server and sorting it is extra work, meaning extra time.

However any relation database has operations that it does not do well. Because of this there are things to avoid doing on the Database server. (eg. Joins of >4 large tables, left outer joins)

KEY TABLES

When running a report on a segment the population, key tables significantly improve performance. Key tables can

be used instead of complicated selection criteria, or used to avoid cross partition joins. This can reduce the number of tables in the query, and the number of rows evaluated for the predicate. Fewer tables and fewer test increase performance.

Sub-grouping can be placed in the key table to provide for totaling at custom levels without complicated predicate, case statements, multiple queries, or rollups on the SAS server.

Key tables can be loaded from selection query on the database, or constructed insert statements based on data from a flat file/dataset.

In following example we show how you can use a list of key in a dataset with a view of DB2 table and then with a key table. Both get the same results, however for large tables the key table will be much faster.

QUERY USING VIEW

```
Proc sql;
Create view acct as select * from connection to db2(
   select acct_no, sum(balance)
     from visa_acct v);
Data result;
   merge acct revolvers (in=a);
   if a;
Proc summary;
  var balance;
  output out=result sum= n=cnt;
```

BUILD AND USING KEY TABLE

```
Data _null_;
  file temp
  set select end=eof;
  if mod(I,100)=1 then
     put “proc sql; connect to db2(database=d3ca &password)”/
           “insert into key_table values(”;
  put “(” acct “, ‘revolvers’)” @;
  If eof or mod(I,100) = 0 then  put ‘);’
  else put ‘,’;
run;
%include temp;

Proc sql;
select * from connection to db2(
   select count(*), sum(balance)
     from visa_acct v,
          key_table k
    where v.acct_no = k.acct_no
      and k.grp = '  revolvers'  );
```

Some shops do not allow key table because user updates may fill the database and cause an outage. This can be avoided using tablespace that dedicated to the users and limited in size.

USING THE CASE STATEMENT

Totals based on indicator/attribute on the row, can cause multiple queries to obtain totals. Example delinquency totals. Case Statements allow you to do this in one pass, significantly improve query times.

Here is how the case statement can be used. Clearly 2 passes of the table will take longer than 1.

USING A CASE STATEMENT

```
select * from connection to db2(
    select count(*) as cnt,
```

```
            sum(balance) as balance,
            case when interest > 1 then '  REVL'   else '  SPND'   end
      from visa_acct
     group by case when interest > 1 then '  REVL'   else '  SPND'   end);
```

NOT USING A CASE STATEMENT

```
select * from connection to db2(
   select count(*) as cnt,
            sum(balance) as balance,
            '  REVL'
      from visa_acct
     where interest > 1
union
   select count(*) as cnt,
            sum(balance) as balance,
            '  SPND'
      from visa_acct
     where interest <= 1);
```

TIME SERIES REPORTING

For time series reports it is worth structuring the report to not recalculate the prior months information. This just wastes computing cycles. Even month over month comparison can be structured to only calculate the new information and compare it to pre-calculated information.

CREATE SET OF TIME SERIES DATASETS

```
Proc sql;
Create table rpt.bal_apr04 as select * from connection to db2(
   select count(*), sum(balance)
     from visa_acct v,
          key_table k
    where v.acct_no = k.acct_no
      and k.grp = '  revolvers'
      and me_dt = '2004-04-01');
data rpt.result;
   set rpt.bal_jan04 rpt.bal_feb04 rpt.bal_mar04 rpt.bal_apr04;
run;
```

USING LIBNAME REFERENCE TO DATABASE

DB2 can be setup to be accessed as a SAS library. This provides a more consistent method of working with data for the SAS programmer he doesn't need to know SQL.

When using the libname structure you have to be mindful of where the data is and how it is going to move as part of the query. With the libname construct the SAS system builds the SQL to access the data. Because the SAS system does not know as much about the data as you do it will not always build the best SQL.

USING LIBNAME ACCESS TO DATABASE

```
Libname datab db2 'database=d3ca';
Data result (keep acct_no balance);
   merge datab.acct revolvers (in=a);
   by acct_no;
   if a;
Proc summary;
  var balance;
  by acct_no;
  output out=result sum= n=cnt;
Proc summary;
  var balance;
  output out=result sum= n=cnt;
```

USING VIEWS INSTEAD OF TEMPORARY DATASETS

Some cases do call for dumping large amounts of data out of the database. The database may not be able to do the calculation you want. It may not be able perform the join logic required. The answer set may be too large without processing as it is built.

Here is an example of an answer set that is too large to handle without processing each row.

```
USING VIEW AS ANSWER SET TOO LARGE
Proc sql;
create view daily as select * from connection to db2(
  select acct_no, dt_ext, balance
    from daily
          order by acct_no dt_ext
 Create view monthly as select * from connection to db2(
  select acct_no, me_dt as dt_ext, balance
    from daily
          order by acct_no me_dt;
 data result;
  set monthly (in=m) daily (in=d);
  by acct_no dt_ext

  .. processing ..

  if last.acct_no;
```

CONCLUSION

There are many ways to get at your data. The best way is dictated by your data, system topography, and program requirements. Consider what is happening, how much data is moving, is there repeated calculation, and multiple database pass that can be avoided.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Robert Ellsworth
Ellsworth Stewart Consulting Inc.
83 Bryant Road
Markham, On., Canada
L3P 5Y8
Phone: 416-414-1172
Fax: 905-471-3298
Email: rob@ellsworthstewart.com
Web: www.ellsworthstewart.com