

A TIP FOR IMPROVING DB2 PERFORMANCE

Ann Stephan

Texas Comptroller of Public Accounts, Austin, TX

When the Texas State Comptroller's office made the decision to switch from VSAM files to DB2, the User Programmers (as the SAS programmers are called) were hit hard. Most of our programs which previously accessed 1 VSAM or flat file now had to read multiple DB2 views. Since the "powers that be" were so afraid that the User Programmers' accessing DB2 would greatly impact the on-line production systems, the decision was made to set up a "Query" database which we could access without creating any problems with the Production database. Then, to make life even more interesting, they put a five-minute governor on the Query database. Five minutes may sound like a lot of processing time, but if you are reading 1 or more tables with 15 million + rows, -905 errors appear often. Consequently, we are always looking for ways to speed up the processing. One day while surfing the net, looking for items on DB2 performance, I ran across this Usage Note:

USAGE NOTE - V6-ENGINE-6053

Improving performance for DB2 Pass-Through to remote subsystems

Products: ACCDB2

To improve performance for the SQL Procedure Pass-Through facility when working with remote tables, specify the FOR FETCH ONLY clause at the end of the Pass-Through query. For example:

```
proc sql;
select * from
    connection to DB2( select * from table FOR FETCH ONLY );
```

This will cause DB2 to make requests for blocks of data through VTAM, rather than single row requests, and can drastically reduce the CPU time required for the remote request.

I was not sure what "remote" files meant, but I thought I would give the "for fetch only" a try. Here are my results:

	CPU usage without "For Fetch Only"	CPU usage with "For Fetch Only"
(See Examples 1A & B for code)	110.30 CPU seconds	32.76 CPU seconds
(See Examples 2A & B for code)	345.78 CPU seconds	89.46 CPU seconds
(See Examples 3A & B for code)	341.66 CPU seconds	87.33 CPU seconds

I later found out that "for fetch only" works for us because the Query database is treated as remote because the read has to go across the DDF(distributed data facility).

EX AMPLE 1A:

```

7 DATA _NULL_; FILE TEMPSQL;
8 COUNT = 1; SET = 1;
9 CHUNK=2000;
10 DO COUNT = 1 TO CHUNK;
11 SET AUDBAS.AUDBAS3 END=LAST;
12 IF COUNT = 1 THEN DO;
13 PUT 'CREATE TABLE JUNK' SET ' AS SELECT * FROM CONNECTION TO DB2';
14 PUT '(SELECT PYR_TP_ID, PYR_NET_PMT_AM, PYR_PDO_END_DT';
15 PUT ' FROM TAXQ.VPAYREV01 WHERE PART_ID IN (';
16 END;
17 IF NOT LAST THEN PUT PART_ID',';
18 IF LAST THEN DO;
19 CALL SYMPUT('NUMSETS',LEFT(PUT(SET,6)));
20 PUT PART_ID") AND PYR_TAX_ID = 26 AND ";
21 PUT "PYR_DTL_TY = 'P' AND ";
22 PUT "PYR_PDO_OBLG_TY = 4 AND ";
23 PUT "PYR_NET_PMT_AM = 0);";
24 COUNT = CHUNK +1;
25 END;
26 IF COUNT=CHUNK THEN DO;
27 PUT PART_ID") AND PYR_TAX_ID = 26 AND ";
28 PUT "PYR_DTL_TY = 'P' AND ";
29 PUT "PYR_PDO_OBLG_TY = 4 AND ";
30 PUT "PYR_NET_PMT_AM = 0);";
31 COUNT=0;
32 SET+1;
33 END;
34 END;
35 RUN;
36 DATA _NULL_; FILE TEMP2;
37 %PUT &NUMSETS;
38 DO N=1 TO &NUMSETS;
39 DSN='JUNK'||TRIM(LEFT(N));
40 PUT 'PROC APPEND BASE=JUNK DATA=' DSN ';;
41 END;
42 RUN;
43 PROC SQL;
44 CONNECT TO DB2 (SSID=DSNQ);
45 %INCLUDE TEMPSQL ;
15112 %PUT &SQLXMSG;
15113 %PUT &SQLXRC;
15114 DISCONNECT FROM DB2;
15116 QUIT;

```

NOTE: The SAS session used 110.30 CPU seconds and 5045K.

NOTE: SAS Institute Inc., SAS Campus Drive, Cary, NC USA 27513-2414

EXAMPLE 1B

```

6      DATA _NULL_;      FILE TEMPSQL;
7      COUNT = 1; SET = 1;
8      CHUNK=2000;
9      DO COUNT = 1 TO CHUNK;
10     SET AUDBAS.AUDBAS3 END=LAST;
11     IF COUNT = 1 THEN DO;
12     PUT 'CREATE TABLE JUNK' SET ' AS SELECT * FROM CONNECTION TO DB2';
13     PUT '(SELECT PYR_TP_ID, PYR_NET_PMT_AM, PYR_PDO_END_DT';
14     PUT ' FROM TAXQ.VPAYREV01 WHERE PART_ID IN (';
15     END;
16     IF NOT LAST THEN PUT PART_ID',';
17     IF LAST THEN DO;
18     CALL SYMPUT('NUMSETS',LEFT(PUT(SET,6)));
19     PUT PART_ID") AND PYR_TAX_ID = 26 AND ";
20     PUT "PYR_DTL_TY = 'P' AND ";
21     PUT "PYR_PDO_OBLG_TY = 4 AND ";
22     PUT "PYR_NET_PMT_AM = 0 FOR FETCH ONLY);";
23     COUNT = CHUNK +1;
24     END;
25     IF COUNT=CHUNK THEN DO;
26     PUT PART_ID") AND PYR_TAX_ID = 26 AND ";
27     PUT "PYR_DTL_TY = 'P' AND ";
28     PUT "PYR_PDO_OBLG_TY = 4 AND ";
29     PUT "PYR_NET_PMT_AM = 0 FOR FETCH ONLY);";
30     COUNT=0;
31     SET+1;      END;      END;
34     RUN;
35     DATA _NULL_;      FILE TEMP2;
36     %PUT &NUMSETS;
37     DO N=1 TO &NUMSETS;
38     DSN='JUNK'||TRIM(LEFT(N));
39     PUT 'PROC APPEND BASE=JUNK DATA=' DSN ';';
40     END;
43     RUN;
46     PROC SQL;
47     CONNECT TO DB2 (SSID=DSNQ);
48     %INCLUDE TEMPSQL ;
15111    %PUT &SQLXMSG;
15112    %PUT &SQLXRC;
15113    DISCONNECT FROM DB2;
15115    QUIT;
15117    %INCLUDE TEMP2 ;

```

NOTE: The SAS session used 32.76 CPU seconds and 5045K.

NOTE: SAS Institute Inc., SAS Campus Drive, Cary, NC USA 27513-2414

EXAMPLE 2A:

```
8      PROC SQL;
9      CONNECT TO DB2 (SSID=DSNQ);
10     CREATE TABLE RTRN AS
11     SELECT * FROM CONNECTION TO DB2
12     (SELECT TP_ID, TAX_ID, OBLG_END_DT
13     FROM TAXQ.VRTRN_AUDBAS01
14     WHERE TAX_ID = 26
15     AND TP_ID < 15000000000);
```

NOTE: Table WORK.RTRN created, with 1187869 rows and 3 columns.

```
16     %PUT &SQLXMSG;
17     %PUT &SQLXRC;
18     DISCONNECT FROM DB2;
19     QUIT;
```

NOTE: The PROCEDURE SQL used 345.78 CPU seconds and 2797K.

EXAMPLE 2B:

```
5      PROC SQL;
6      CONNECT TO DB2 (SSID=DSNQ);
7      CREATE TABLE RTRN AS
8      SELECT * FROM CONNECTION TO DB2
9      (SELECT TP_ID, TAX_ID, OBLG_END_DT
10     FROM TAXQ.VRTRN_AUDBAS01
11     WHERE TAX_ID = 26
12     AND TP_ID < 15000000000
13     FOR FETCH ONLY);
```

NOTE: Table WORK.RTRN created, with 1187869 rows and 3 columns.

```
14     %PUT &SQLXMSG;
15     %PUT &SQLXRC;
16     DISCONNECT FROM DB2;
17     QUIT;
```

NOTE: The PROCEDURE SQL used 89.46 CPU seconds and 2797K.

EXAMPLE 3A:

```
6      PROC SQL;
7          CONNECT TO DB2 (SSID=DSNQ);
8          CREATE TABLE PMTS AS
9              SELECT * FROM CONNECTION TO DB2
10                 (SELECT PYR_TP_ID, PYR_TAX_ID, PYR_PDO_END_DT
11                    FROM TAXQ.VPAYREV01
12                     WHERE PYR_TAX_ID = 26
13                      AND PYR_TP_ID < 15000000000);
```

NOTE: Table WORK.PMTS created, with 1154713 rows and 3 columns.

```
14      %PUT &SQLXMSG;
15      %PUT &SQLXRC;
16      DISCONNECT FROM DB2;
17      QUIT;
```

NOTE: The PROCEDURE SQL used 341.66 CPU seconds and 2797K.

EXAMPLE 3B:

```
5      PROC SQL;
6          CONNECT TO DB2 (SSID=DSNQ);
7          CREATE TABLE PMTS AS
8              SELECT * FROM CONNECTION TO DB2
9                 (SELECT PYR_TP_ID, PYR_TAX_ID, PYR_PDO_END_DT
10                    FROM TAXQ.VPAYREV01
11                     WHERE PYR_TAX_ID = 26
12                      AND PYR_TP_ID < 15000000000
13                      FOR FETCH ONLY);
```

NOTE: Table WORK.PMTS created, with 1154713 rows and 3 columns.

```
14      %PUT &SQLXMSG;
15      %PUT &SQLXRC;
16      DISCONNECT FROM DB2;
17      QUIT;
```

NOTE: The PROCEDURE SQL used 87.33 CPU seconds and 2797K.