



Explain Yourself and Improve DB2 Performance!

—

Jim Dee

Chief Architect for DB2, BMC

IBM Information Champion

2016

Dec / 2016



Key Points

Understand new EXPLAIN capabilities, added in DB2 10 and 11, and how to exploit them.

Explore the ability to save access path history, and how to use it to analyze trends and to detect unwanted access path changes.

Understand the mechanism of SYSSTATEFEEDBACK and how it can be used to guide RUNSTATS executions.

Discuss selectivity profiles and how to use them to improve the operation of the DB2 optimizer.

SQL Tuning – like Whac-a-Mole?



There's always a new problem

As soon as you think you have them all fixed, a new one pops up

EXPLAIN Tables

PLAN_TABLE	DSN_FUNCTION_TABLE
DSN_STATEMNT_TABLE	DSN_STATEMENT_CACHE_TABLE
DSN_QUERYINFO_TABLE	DSN_PREDICAT_TABLE
DSN_STRUCT_TABLE	DSN_PGROUP_TABLE
DSN_PTASK_TABLE	DSN_FILTER_TABLE
DSN_DETCOST_TABLE	DSN_SORT_TABLE
DSN_SORTKEY_TABLE	DSN_PGRANGE_TABLE
DSN_VIEWREF_TABLE	DSN_QUERY_TABLE
DSN_VIRTUAL_INDEXES	DSN_COLDIST_TABLE
DSN_KEYTGTDIST_TABLE	DSN_VIRTUAL_KEYTARGETS
DSN_PREDICATE_SELECTIVITY	DSN_STAT_FEEDBACK

New EXPLAIN Features

Access path stability

- SYSPACKCOPY added in DB2 10
- APCOMPARE and APREUSE added in DB2 10
- APREUSE(WARN) added in DB2 11 – allows statement level control
- APREUSESOURCE added in DB2 12
- Ability to FREE INACTIVE, PREVIOUS, or ORIGINAL added in DB2 12

New columns were added to the DSN_VIRTUAL_INDEXES table, and a new DSN_VIRTUAL_KEYTARGETS table was added

- Support for Index on Expression, XML indexes, Index Include Column, and null suppressed indexes

New EXPLAIN Features

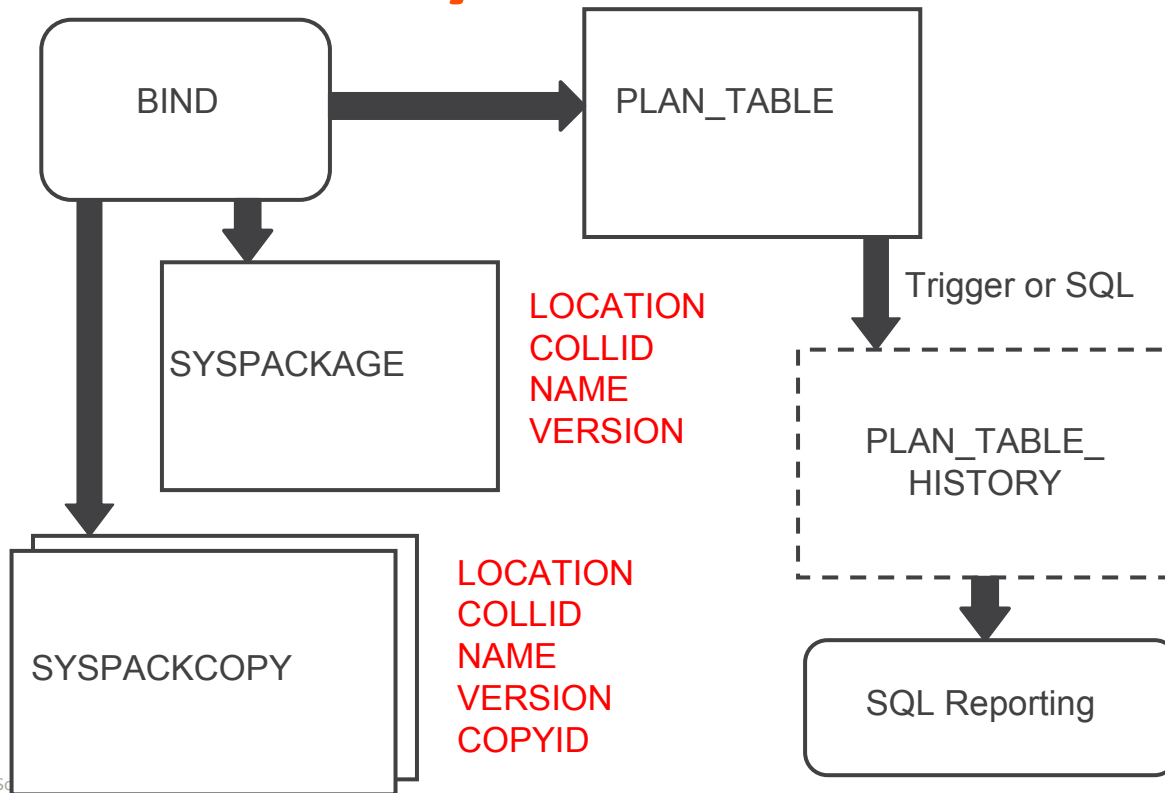
EXPLAIN (ONLY)

EXPLAIN PACKAGE

– EXPLAIN STABILIZED DYNAMIC QUERY

SYSDYNQRY table added in DB2 12 to hold stabilized access path for dynamic SQL

EXPLAIN History



What Can I Get Out of My EXPLAIN Table?

```
SELECT SUBSTR(PL.COLLDID,1,10) AS COLLDID,  
       SUBSTR(PL.PROGNAME,1,10) AS PROGNAME,  
       DATE(PL.EXPLAIN_TIME) AS DATE,  
       TIME(PL.EXPLAIN_TIME) AS TIME,  
       COUNT(PL.QUERYNO) AS "STMT COUNT",  
       DEC(SUM(ST.TOTAL_COST),8,2) AS "TOTAL COST"  
FROM SJD.PLAN_TABLE PL,  
     SJD.DSN_STATEMNT_TABLE ST  
WHERE PL.PROGNAME = ST.PROGNAME  
     AND PL.COLLDID = ST.COLLDID  
     AND PL.EXPLAIN_TIME = ST.EXPLAIN_TIME  
     AND PL.QUERYNO = ST.QUERYNO  
GROUP BY PL.COLLDID, PL.PROGNAME, PL.EXPLAIN_TIME  
ORDER BY PL.PROGNAME;
```


Sample Results for a Very Simple Example

COLLID	PROGNAME	DATE	TIME	STMT COUNT	TOTAL COST
MYCOLL	MYPACK	12/08/2016	11.19.38	5	10.55
MYCOLL	MYPACK	12/08/2016	17.36.17	8	19.03

DSNE610I NUMBER OF ROWS DISPLAYED IS 2

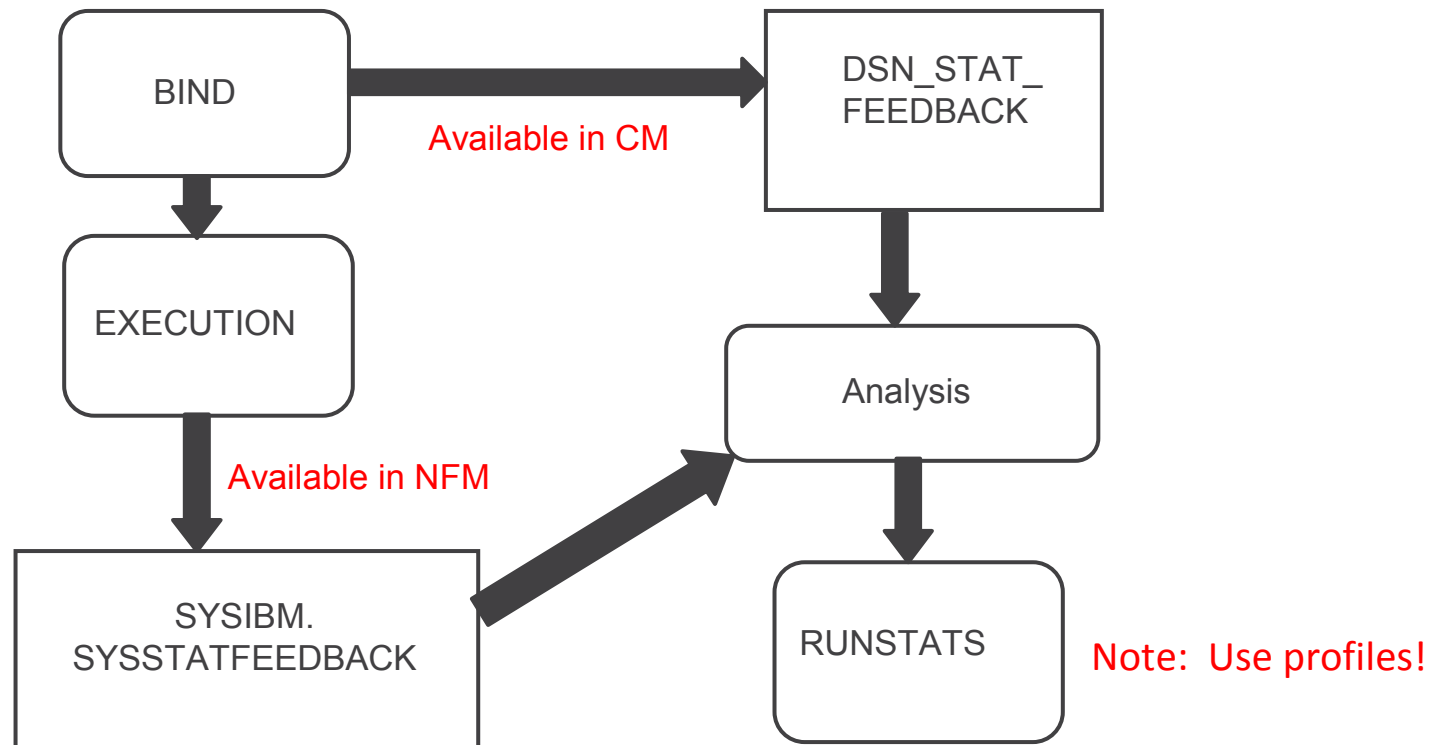
Can I Detect Changed Costs?

```
SELECT ST1.QUERYNO AS QUERYNO,  
       COALESCE(DEC(ST1.TOTAL_COST,8,2),0) AS "OLD COST",  
       COALESCE(DEC(ST2.TOTAL_COST,8,2),0) AS "NEW COST"  
FROM SJD.DSN_STATEMNT_TABLE ST1 FULL JOIN  
     SJD.DSN_STATEMNT_TABLE ST2  
ON   ST1.QUERYNO = ST2.QUERYNO  
WHERE ST1.COLLID = 'MYCOLL' AND ST2.COLLID = 'MYCOLL'  
      AND ST1.PROGNAME = 'MYPACK'  
      AND ST2.PROGNAME = 'MYPACK'  
      AND DATE(ST1.EXPLAIN_TIME) = '2016-12-01'  
      AND TIME(ST1.EXPLAIN_TIME) = '17.36.17'  
      AND DATE(ST2.EXPLAIN_TIME) = '2016-12-08'  
      AND TIME(ST2.EXPLAIN_TIME) = '13.05.14'  
      AND ST1.TOTAL_COST <> ST2.TOTAL_COST  
ORDER BY QUERYNO;
```

Can I Detect Changed Costs?

```
-----+-----+-----+-----+-----  
      QUERYNO          OLD COST          NEW COST  
-----+-----+-----+-----+-----  
          353              .15              .11  
          360             8.07             16.24  
DSNE610I NUMBER OF ROWS DISPLAYED IS 2
```

Optimizer Help for RUNSTATS



The Stats Feedback Tables

DSN_STATS_FEEDBACK

- QUERYNO
- APPLNAME
- PROGNAME
- COLLID
- GROUP_MEMBER
- EXPLAIN_TIME
- SECTNOI
- VERSION
- Most of SYSSTATFEEDBACK
Columns

SYSSTATFEEDBACK

- TBCREATOR
- TBNAME
- IXCREATOR
- IXNAME
- COLNAME
- NUMCOLUMNS
- COLGROUPCOLNO
- TYPE
- DBNAME
- TSNAME
- REASON
- BLOCK_RUNSTATS
- REMARKS
- LASTDATE

The Stats Feedback Columns Of Interest

COLNAME

- Name of the single column associated with the recommendation

NUMCOLUMNS

- Number of columns if the recommendation is for more than one column

COLGROUPCOLNO

- Binary field: array of halfwords which are COLNO references

TYPE

- CHAR(1), representing the type of statistic to collect
- 'C' (cardinality), 'F' (frequency), 'H' (histogram), 'I' (index), 'T' (table)

REASON

- CHAR(8), representing the reason for the recommendation
- 'BASIC', 'KEYCARD', 'LOWCARD', 'NULLABLE', 'DEFAULT', 'RANGEPRD', 'PARALLEL', 'CONFLICT', 'COMPFFIX'

How To Get Stats Help

To see DSN_STAT_FEEDBACK results, you must create the table before running EXPLAIN

- Available in DB2 11 CM
- DSNSAMP(DSNTESC)

To see SYSSTATFEEDBACK, you must ensure that the STATSFDBK_SCOPE zparm is set to something other than NONE

- STATSINT zparm – REAL TIME STATS field on DSNTIPO (Operator Functions) – specifies time in minutes between externalizations
- ACCESS DB MODE(STATS) updates SYSSTATFEEDBACK from memory – externalizes RTS also

What If You Want to Try This Gradually?

You can use the `STATSFDBK_SCOPE` zparm to restrict stats feedback collection

- NONE, STATIC, DYNAMIC, or ALL (default)
- Specified in the STATISTICS FEEDBACK field in install panel DSNTIPO (Operator Functions)

You can use the `STATS_FEEDBACK` column in `SYSTABLES` to disable statistics feedback for the table

- Can set it to “N” or “Y”

You can set the `BLOCK_RUNSTATS` column to ‘Y’

- Process that generates RUNSTATS must honor this

Other Considerations For Stats Feedback

Stats Feedback doesn't work in certain cases:

- VOLATILE
- DGTT
- CGTT

Use of Hints, APREUSE, APCOMPARE may interfere during EXPLAIN

Enhanced in DB2 12!

If new ZPARM STATFDBK_PROFILE = 'Y'

- RUNSTATS profiles are updated automatically at the same time as SYSTATFEEDBACK
- SYSIBM.SYSTABLES_PROFILES

Profile is created before updating, if it does not exist

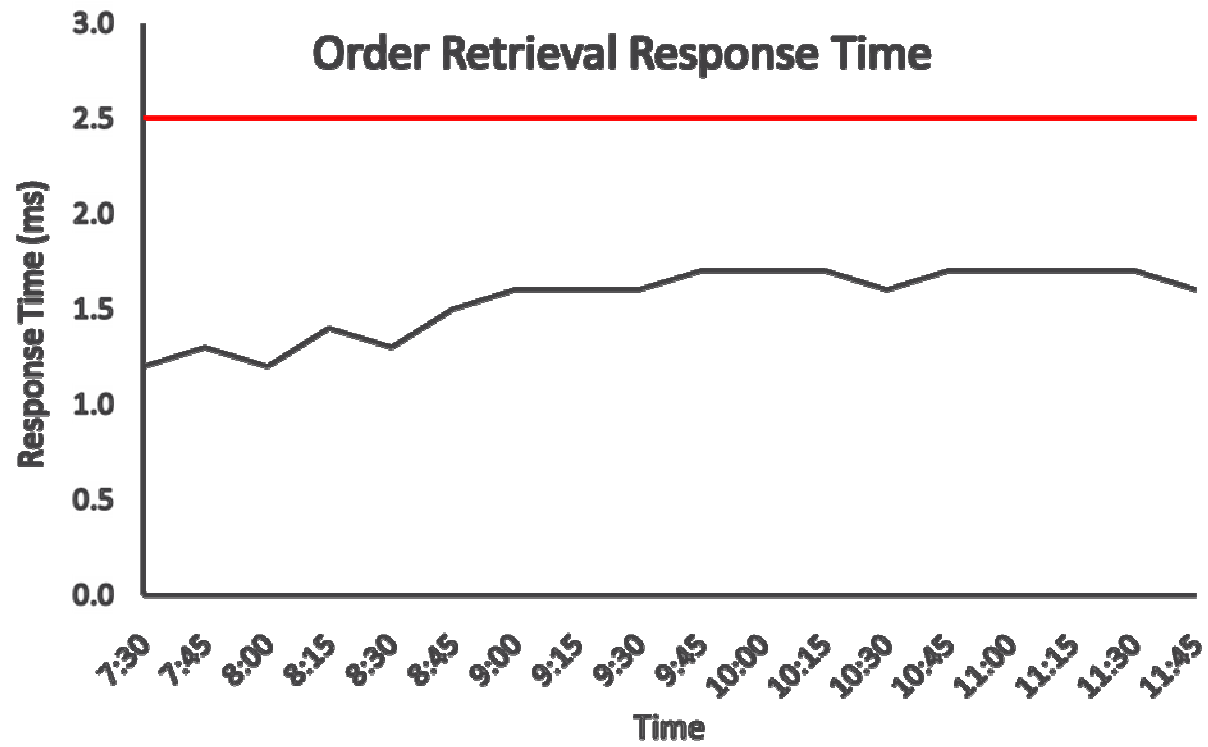
REASON value of 'STALE' added to SYSTATFEEDBACK

RUNSTATS removes unused multi-column frequency stats

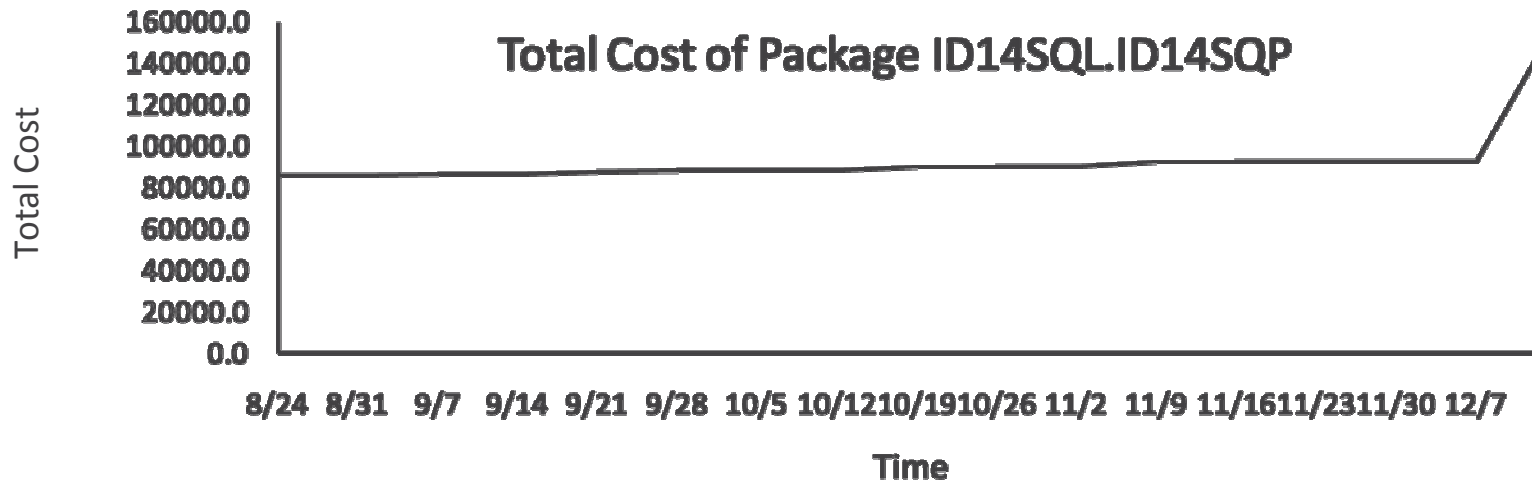
Selectivity Profiles



Is There a Problem?



Maybe We Should Check Our EXPLAIN History



COLLID	PROGNAME	DATE	TIME	STMT COUNT	TOTAL COST
ID14SQL	ID14SQP	11/30/2016	05:58 PM	9	92741.17
ID14SQL	ID14SQP	12/07/2016	05:39 PM	9	92741.17
ID14SQL	ID14SQP	12/13/2016	06:03 PM	10	148921.15

What SQL Is It?

```
SELECT QUERYNO, TOTAL_COST, EXPLAIN_TIME
       FROM SJD.DSN_STATEMNT_TABLE
       WHERE PROGNAME = 'ID14SQP'
              AND COLLID = 'ID14SQL'
       ORDER BY TOTAL_COST DESC, EXPLAIN_TIME DESC;
```

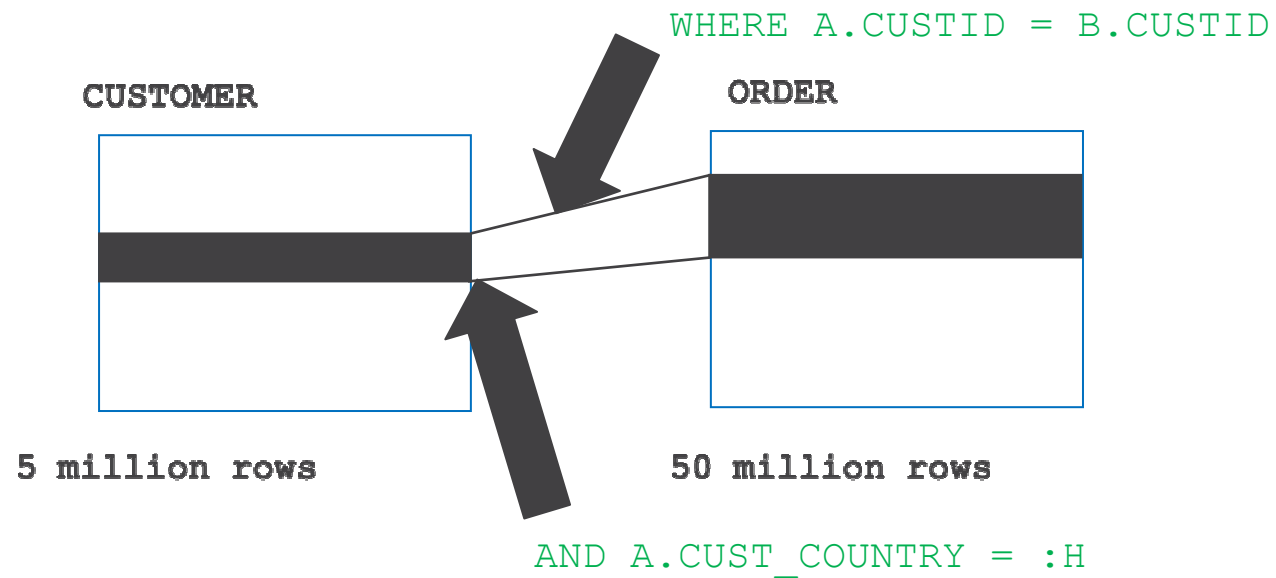
```
-----+-----+-----+-----+-----+-----+-----
QUERYNO                TOTAL_COST  EXPLAIN_TIME
-----+-----+-----+-----+-----+-----+-----
      88  +0.5616997729372477E+05  2016-12-13-18.03.04.993751
     131  +0.1453335050780900E+01  2016-12-13-18.03.04.993751
     131  +0.1453335050780900E+01  2016-12-07-16.55.51.641367
```

What SQL Is It?

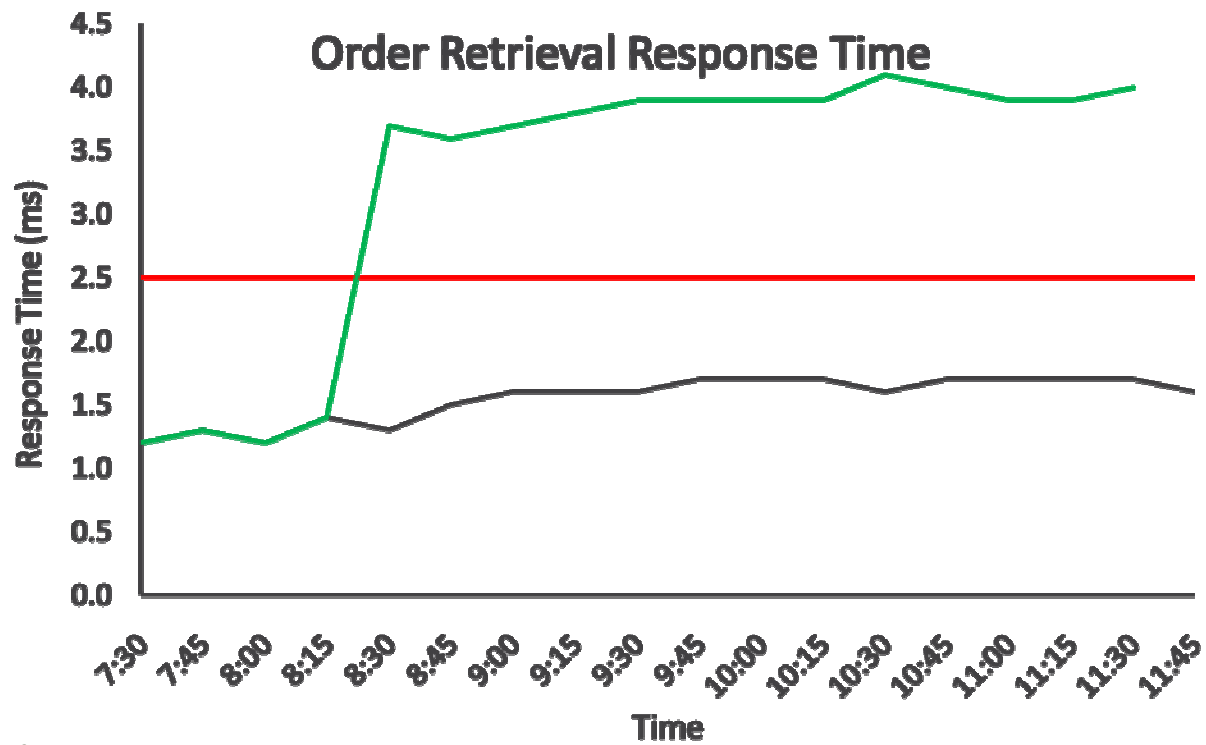
```
SELECT STATEMENT
      FROM SYSIBM.SYSPACKSTMT
      WHERE COLLID = 'ID14SQL' AND NAME = 'ID14SQP'
      AND QUERYNO = 88;
-----+-----+-----+-----+-----+-----+-----
STATEMENT
-----+-----+-----+-----+-----+-----+-----
DECLARE TOPTEN CURSOR FOR SELECT LASTNM , ADDR5 , ID , ORDCOUNT ,

SELECT LASTNM, ADDR5, ID, ORDCOUNT, GTOTAL
      FROM ( SELECT LASTNM, ADDR5, ID, COUNT(ORDNO) AS ORDCOUNT, SUM(TOTAL) AS GTOTAL
      FROM ( SELECT A.CUST_LASTNM AS LASTNM, A.CUST_ADDRESS AS ADDR5,
      A.CUSTID AS ID, B.ORDNO AS ORDNO,B.ORDER_TOTAL AS TOTAL
      FROM SJDID001.CUSTOMER A, SJDID001.ORDER B
      WHERE A.CUSTID = B.CUSTID AND A.CUST_COUNTRY = :H ) X
      GROUP BY LASTNM, ADDR5, ID ) Y
WHERE ORDCOUNT > :H AND GTOTAL > :H
ORDER BY GTOTAL DESC, ORDCOUNT DESC FOR FETCH ONLY OPTIMIZE FOR 10 ROWS
```

What is the SQL Doing?



Is There a Problem?



A Little More Research...

From PLAN_TABLE:

PLANNNO	METHOD	ACCESSTYPE	MATCHCOLS	PREFETCH	CREATOR	TNAME	TABNO
1	0	I	1	L	SJDID001	CUSTOMER	1
2	4	I	1	L	SJDID001	ORDER	2
3	3		0				0
4	3		0				0

DSNE610I NUMBER OF ROWS DISPLAYED IS 4

From DSN_PREDICAT_TABLE:

PREDNO	TYPE	FILTER_FACTOR	LHS	TEXT
1	AND	+0.10000000000000000E+01		("A"."CUSTID"="B".
2	EQUAL	+0.99999994290410541E-06	CUSTID	"A"."CUSTID"="B".
3	EQUAL	+0.5263157188892365E-02	CUST_COUNTRY	"A"."CUST_COUNTRY"
4	AND	+0.10000000000000000E+01		(COUNT("B"."ORDNO"
5	RANGE	+0.3333333134651184E+00	VALUE	COUNT("B"."ORDNO")
6	RANGE	+0.3333333134651184E+00	VALUE	SUM("B"."ORDER_TOT
7	COMPOUND	+0.99999994290410541E-06		COMPOUND: 2,3

A Little More Research...

```
SELECT SUBSTR(KY.COLNAME,1,18) AS COL, KY.COLSEQ,  
       CO.COLTYPE, CO.LENGTH  
FROM SYSIBM.SYSINDEXES IX, SYSIBM.SYSKEYS KY,  
     SYSIBM.SYSCOLUMNS CO  
WHERE IX.TBCREATOR = 'SJDID001' AND IX.TBNAME = 'CUSTOMER'  
      AND IX.NAME = 'CUSTADDR' AND IX.CREATOR = 'SJDID001'  
      AND IX.NAME = KY.IXNAME  AND IX.CREATOR = KY.IXCREATOR  
      AND IX.TBNAME = CO.TBNAME AND IX.TBCREATOR = CO.TBCREATOR  
      AND KY.COLNO = CO.COLNO  
ORDER BY KY.COLSEQ;
```

```
-----+-----+-----+-----+-----+-----+-----+  
COL              COLSEQ  COLTYPE  LENGTH  
-----+-----+-----+-----+-----+-----+  
CUST_COUNTRY           1  CHAR         2  
CUST_ADDRESS           2  VARCHAR       120  
DSNE610I NUMBER OF ROWS DISPLAYED
```

A Little More Research...

```
SELECT STATTIME, INT(FIRSTKEYCARD) AS FIRSTKEYCARD,  
       INT(FULLKEYCARD) AS FULLKEYCARD, DEC(CLUSTERRATIOF, 3, 1) AS CLUSTERRATIO  
FROM SYSIBM.SYSINDEXES WHERE CREATOR = 'SJDID001' AND NAME = 'CUSTADDR';
```

```
-----+-----+-----+-----+-----+-----  
STATTIME          FIRSTKEYCARD    FULLKEYCARD    CLUSTERRATIO  
-----+-----+-----+-----+-----+-----  
2016-11-28-08.32.13.545461          190          4975000          .3  
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
```

```
SELECT COUNT(*) FROM SJDID001.CUSTOMER WHERE CUST_COUNTRY = 'US';
```

```
-----+-----+-----+-----+-----  
4000000
```

```
SELECT COUNT(*) FROM SJDID001.CUSTOMER WHERE CUST_COUNTRY = 'CA';
```

```
-----+-----+-----+-----+-----  
950000
```

```
SELECT COUNT(*) FROM SJDID001.CUSTOMER WHERE CUST_COUNTRY = 'FR';
```

```
-----+-----+-----+-----+-----  
500
```

How Do You Correct This?

Three EXPLAIN tables involved

- PLAN_TABLE, DSN_PREDICAT_TABLE and DSN_PREDICATE_SELECTIVITY
- You should create these with a new schema, to separate them functionally
- DO NOT use “CREATE TABLE ...LIKE...” ... bad things happen
 - Use the DDL from DSNSAMP(DSNTESC)
 - Same warning applies to DSN_USERQUERY_TABLE

You can update DSN_PREDICATE_SELECTIVITY with the critical filter factor data

- A row for “CUST_COUNTRY = ‘US’, with WEIGHT = .5, SELECTIVITY = .80
- A row for “CUST_COUNTRY = ‘CA’, with WEIGHT = .1, SELECTIVITY = .19

ASSUMPTION must be set to ‘OVERRIDE’ for these rows

The optimizer will assume default filtering for all other predicates

What's in DSN_PREDICATE_SELECTIVITY now?

```

SELECT PREDNO, INSTANCE, SELECTIVITY, WEIGHT,
       SUBSTR(ASSUMPTION,1,8) AS ASSUM
FROM SJD.DSN_PREDICATE_SELECTIVITY
WHERE PROGNAME = 'ID14SQP'
       AND COLLID = 'ID14SQL'
       AND EXPLAIN_TIME = '2016-12-13-18.03.04.993751'
       AND QUERYNO = 88
ORDER BY PREDNO;

```

PREDNO	INSTANCE	SELECTIVITY	WEIGHT	ASSUM
1	0	+0.10000000000000000E+01	+0.100000000E+01	NORMAL
2	0	+0.9999994290410541E-06	+0.100000000E+01	NORMAL
3	0	+0.5263157188892365E-02	+0.100000000E+01	NORMAL
4	0	+0.10000000000000000E+01	+0.100000000E+01	NORMAL
5	0	+0.3333333134651184E+00	+0.100000000E+01	NORMAL
6	0	+0.3333333134651184E+00	+0.100000000E+01	NORMAL
7	0	+0.9999994290410541E-06	+0.100000000E+01	NORMAL

How do we update DSN_PREDICATE_SELECTIVITY?

```
INSERT INTO SJDUPD.DSN_PREDICATE_SELECTIVITY
SELECT * FROM SJD.DSN_PREDICATE_SELECTIVITY
  WHERE COLLID = 'ID14SQL' AND PROGNAME = 'ID14SQP'
     AND EXPLAIN_TIME = '2016-12-13-18.03.04.993751'
     AND QUERYNO = 88;
UPDATE SJDUPD.DSN_PREDICATE_SELECTIVITY
SET INSTANCE = 1, SELECTIVITY = .8, WEIGHT = .5,
  ASSUMPTION = 'OVERRIDE'
  WHERE COLLID = 'ID14SQL' AND PROGNAME = 'ID14SQP'
     AND EXPLAIN_TIME = '2016-12-13-18.03.04.993751'
     AND QUERYNO = 88 AND PREDNO = 3 AND INSTANCE = 0;
INSERT INTO SJDUPD.DSN_PREDICATE_SELECTIVITY
(QUERYNO, QBLOCKNO, APPLNAME, PROGNAME, SECTNOI, COLLID, VERSION, PREDNO,
 INSTANCE, SELECTIVITY, WEIGHT, ASSUMPTION, INSERT_TIME, EXPLAIN_TIME,
  REMARKS, EXPANSION_REASON)
SELECT (QUERYNO, QBLOCKNO, APPLNAME, PROGNAME, SECTNOI, COLLID, VERSION, PREDNO,
  2, .19, .1, ASSUMPTION, INSERT_TIME, EXPLAIN_TIME, REMARKS, EXPANSION_REASON)
  FROM SJDUPD.DSN_PREDICATE_SELECTIVITY
  WHERE COLLID = 'ID14SQL' AND PROGNAME = 'ID14SQP'
     AND EXPLAIN_TIME = '2016-12-13-18.03.04.993751'
     AND QUERYNO = 88 AND PREDNO = 3 AND INSTANCE = 1;
```

Final Step (Almost)

Now, you must insert a row into the DSN_USERQUERY_TABLE table, as for a hint (SQL is on the next slide but one)

- This is not created with the EXPLAIN tables...DSNSAMP(DSNTESH)

Copy QUERY_TEXT from SYSPACKSTMT

Set HINT_SCOPE to 1 (package level hint)

Set SELECTVTY_OVERRIDE to 'Y'

Final Step (Almost)

Set `ACCESSPATH_HINT` to 'N' (this is not a hint)

Set `OPTION_OVERRIDE` to 'N' (assuming you don't want to override a BIND option)

`BIND QUERY LOOKUP(NO) EXPLAININPUTSCHEMA('SJDUPD')`

- Populates the appropriate catalog tables

BIND for your package will now consider the new filter factors

- **IF** zparm `OPTHINTS` is YES

INSERT Into DSN_USERQUERY_TABLE

```
INSERT INTO SJD.DSN_USERQUERY_TABLE
  (QUERYNO, SCHEMA, HINT_SCOPE, QUERY_TEXT,
   COLLECTION, PACKAGE,
   SELECTVTY_OVERRIDE, ACCESSPATH_HINT, OPTION_OVERRIDE)
SELECT QUERYNO, 'SJD', 1, STATEMENT,
  COLLID, NAME, 'Y', 'N', 'N'
FROM SYSIBM.SYSPACKSTMT
WHERE COLLID = 'ID14SQL' AND NAME = 'ID14SQP'
  AND QUERYNO = 88;
```

READY

DSN SYSTEM(DJO1)

DSN

BIND QUERY LOOKUP(NO)

DSNT280I *DJO1 BIND QUERY FOR QUERYNO = 88 SUCCESSFUL

DSNT290I *DJO1 BIND QUERY COMMAND COMPLETED

DSN

END

READY

END

BIND QUERY Again

```
READY
  DSN SYSTEM(DJO1)
DSN
  BIND QUERY LOOKUP (YES)
DSNT280I *DJO1 LOOKUP QUERY FOR QUERYNO = 88 SUCCESSFUL
DSNT290I *DJO1 LOOKUP QUERY COMMAND COMPLETED
DSN
  END
READY
END
```

This proves that the catalog tables were updated successfully. The other tables updated are SYSIBM.SYSQUERYPRDICATION and SYSIBM.SYSQUERYSEL.

```
SELECT QUERYID, SUBSTR(SCHEMA,1,8) AS SCHEMA, USERFILTER,
       SUBSTR(OTHER_OPTION,1,16) AS OTHER_OPTION, PLAN_VALID, INVALID_REASON,
       SUBSTR(STMTTEXT,1,20) AS STMTTEXT
FROM SYSIBM.SYSQUERY
WHERE COLLECTION = 'ID14SQL' AND PACKAGE = 'ID14SQP';
```

```
+-----+-----+-----+-----+
|          QUERYID          | SCHEMA | USERFILTER | OTHER_OPTION |
+-----+-----+-----+-----+
|                          1 | SJD    |           |             |
```

And, Finally, We BIND Again...

```
READY
DSN SYSTEM(DJO1)
DSN
  BIND PACKAGE      (ID14SQL) MEMBER(ID14SQP)          OWNER      (SJD)
                    ISOLATION (CS) CURRENTDATA(NO)
DSNT297I *DJO1 DSNTBBP2 BIND WARNING FOR
          PACKAGE = DJO.ID14SQL.ID14SQP,
          USE OF SELECTIVITY OVERRIDES RESULTS IN:
          1 STATEMENTS WHERE SELECTIVITY OVERRIDES ARE FULLY
            APPLIED,
          0 STATEMENTS WHERE SELECTIVITY OVERRIDES ARE INVALID,
          3 STATEMENTS WHERE SELECTIVITY OVERRIDES ARE NOT FOUND.
DSNT254I *DJO1 DSNTBCM2 BIND OPTIONS FOR
```

PLANNO	METHOD	TABNO	ACCESSTYPE	MATCHCOLS	PREFETCH
1	0	1	I	0	S
2	2	2	R	0	S
3	3	0		0	

What Changed?

```
SELECT QUERYNO,  
       COALESCE(DEC(TOTAL_COST,8,0),0) AS "COST",  
       EXPLAIN_TIME  
       FROM SJD.DSN_STATEMNT_TABLE  
WHERE COLLID = 'ID14SQL'  
       AND PROGNAME = 'ID14SQP'  
ORDER BY EXPLAIN_TIME DESC;
```

```
-----+-----+-----+-----+  
QUERYNO      COST      EXPLAIN_TIME  
-----+-----+-----+-----+  
      88      434758      2016-12-14-09.03.04.993751  
      88      23184      2016-12-13-16.55.51.641367  
DSNE610I NUMBER OF ROWS DISPLAYED IS 2
```

Bibliography

“DB2 11 for z/OS Managing Performance”, SC-19-4060-05

“IBM DB2 11 for z/OS Performance Topics” (redbook), SG24-8222-0

“DB2 11 for z/OS SQL Reference”, SC19-4066-02

“DB2 11 for z/OS Installation and Migration Guide”, GC19-4056-05

Jim Dee

BMC Software

jim_dee@bmc.com

Stop Wasting Time With Utilities

Thank You

—
Bring IT to Life.™