



IDUG DB2 EMEA Tech Conference
Dublin, Ireland | November 2015

#IDUGDB2

More Sage Advice: Invaluable DB2 LUW Performance Insights from Around the Globe

Scott Richard Hayes, *DBI Software*

Session Code: D10

Wednesday, 18 November, 8:30-9:30

Platform: DB2 for Linux, UNIX, and Windows



How much does it weigh? TOTAL weight and RELATIVE weight...



3/12

The scale shows the total weight of a person with ugly toes.

If you were going hiking, how much might your back pack weigh?

What's in your backpack?

Does it weigh too much?

If you want the back pack to weigh less, and you have limited time, what might you remove?

We have a **WEIGHT** “Opportunity for Improvement”



When your database is carrying too much weight, forward motion stops! Or sputters forward at a snail's pace... and your phone invariably rings! “The database is slow!” Now you need to lessen the weight burden, and fast!!

Table Performance Analysis Table Rows Read per Transaction (TBRRTX) & WEIGHT

- Not every TX accesses every table, so we expect Rows Read/#TX to be a small average, normally < 10, and often 3 or less
 - TBRRTX tells you where you have Data Page scans occurring
 - > 10, likely opportunity for improvement
 - > 100, definitely opportunity for improvement
 - > 1,000, crisis! **DO NOT UPGRADE HARDWARE**
- In addition to the cost per TX, find the % of DB Rows Read (**Relative Weight**) by expressing Table Rows Read x 100 / Sum of all Rows Read.

4

-- Author: Scott.Hayes@DBIsoftware.com

-- Version: 3.0

-- Last Updated: 2014-03-06

-- Copyright 2014 DBI. All Rights Reserved.

-- Licensed for use by IDUG 2014 Attendees and DBI Authorized Customers Only

--

```
select substr(a.tabschema,1,20) as TABSCHEMA,  
       substr(a.tabname,1,25) as TABNAME,  
       a.rows_read as RowsRead,  
       CAST((((A.ROWS_READ) * 100.0)  
            / (Select (SUM(Z.ROWS_READ) + 1.0)  
                FROM SYSIBMADM.SNAPTAB Z  
                WHERE A.DBPARTITIONNUM = Z.DBPARTITIONNUM  
                )) AS DECIMAL(5,2)) AS PCT_DB_TB_ROWSREAD,  
       CAST( (a.rows_read / (b.commit_sql_stmts + b.rollback_sql_stmts + 1.0))  
            AS DECIMAL(13,3)) as TBRRTX
```

```
from SYSIBMADM.snaptab a,  
     SYSIBMADM.snapdb b  
where a.dbpartitionnum = b.dbpartitionnum  
order by a.rows_read desc fetch first 20 rows only;
```






Examples Table Relative Weights and Read I/O Costs

Table Workload from 10/31/13 9:25 AM to 10/31/13 12:25 PM

Schema	Table	Size (MB)	% Space	Rows Read	% Rows Read	Rows Read/Tx	Rows Read/Sec	Rows Written	% Rows Written	Rows Written/Tx
MCSADM	XPX_PROMOTION	59.414	0.000%	314,269,599	9.250%	3.88	28,315.13	0	0.000%	0.00
MCSADM	FX_PROMOTION	13,581.410	0.190%	252,701,117	7.430%	3.12	22,767.92	0	0.000%	0.00
MCSADM	TABSCHEMA									
MCSADM	TABNAME			ROWSREAD		PCT_DB_TB_ROWSREAD		TBRRTX		
MCSADM	DBI_POC			1325709286		99.23		322086.804		0.02
MCSADM	DBI_POC_DMS			10141825		0.75		2464.000		0.00
MCSADM	IDUG			15695		0.00		3.813		0.00
MCSADM	SVS1BM			559		0.00		0.135		0.03
MCSADM	SVS1BM			146		0.00		0.035		0.00
MCSADM	SVS1BM			133		0.00		0.032		0.00
MCSADM	SVS1BM			40		0.00		0.009		0.01
MCSADM	SVS1BM			35		0.00		0.008		0.00
MCSADM	SVS1BM			28		0.00		0.006		0.00
MCSADM	SVS1BM			18		0.00		0.004		0.04
MCSADM	SVS1BM			10		0.00		0.002		0.01
MCSADM	SVS1BM			8		0.00		0.001		0.00
MCSADM	SVS1BM			8		0.00		0.001		0.00
MCSADM	SVS1BM			8		0.00		0.001		0.05
MCSADM	SVS1BM			7		0.00		0.001		0.00
MCSADM	SVS1BM			6		0.00		0.001		0.00
MCSADM	SVS1BM			6		0.00		0.001		0.00
MCSADM	SVS1BM			4		0.00		0.000		0.00
MCSADM	SVS1BM			2		0.00		0.000		0.00
MCSADM	SVS1BM			2		0.00		0.000		0.00
MCSADM	XCEFFM	46,277.750	0.660%	43,625,040	1.280%	0.54	3,930.54	41,319	0.110%	0.00

-- Author: Scott.Hayes@DBIsoftware.com

-- Version: 3.0

-- Last Updated: 2014-03-06

-- Copyright 2014 DBI. All Rights Reserved.

-- Licensed for use by IDUG 2014 Attendees and DBI Authorized Customers Only

--

select substr(a.tabschema,1,20) as TABSCHEMA,

substr(a.tabname,1,25) as TABNAME,

a.rows_read as RowsRead,

CAST((((A.ROWS_READ) * 100.0)

/ (Select (SUM(Z.ROWS_READ) + 1.0)

FROM SYSIBMADM.SNAPTAB Z

WHERE A.DBPARTITIONNUM = Z.DBPARTITIONNUM

)) AS DECIMAL(5,2)) AS PCT_DB_TB_ROWSREAD,

CAST((a.rows_read / (b.commit_sql_stmts + b.rollback_sql_stmts + 1.0))

AS DECIMAL(13,3)) as TBRRTX

```
from SYSIBMADM.snaptab a,  
     SYSIBMADM.snapdb b  
where a.dbpartitionnum = b.dbpartitionnum  
order by a.rows_read desc fetch first 20 rows only;
```

http://www.dbisoftware.com/blog/db2_performance.php?id=123

- **How DB2 Sees the SQL Workload:**

Select c1, c2, c4 from tbl where c5 = '0360' cpu=.1
100's of SQL statements per second...

SQL Snapshot shows 19 different statements!
WRONG ANSWER!

Relative
Costs

- **How the DBA needs to see the SQL Workload:**

SQL Statement	Count	TotCPU	CPU%
Select c1, c2, c4 from tbl where c5 = '?'	16	1.6	66.6
Select c1, c2, c4 from tbl where c5 > '?'	2	.6	25.0
Select c1, c2, c4 from tbl where c8 = '?'	1	.2	8.33
Totals:	19	2.4	100.00

US Patent # 6,772,411

"Costly SQL" – Aggregate Costs with Relative WEIGHTS

SQL Cost aggregation is imperative to successfully understanding statement workload costs. In the absence of aggregating statement costs, you are merely hunting elephants. When costs are aggregated, your analysis may likely provide shocking and invaluable insights into performance. More help on this topic is available at <http://www.dbisoftware.com/help/index.php>

SQL WEIGHTS

Aggregated, Concentrated Costs, & their WEIGHTS

- Now that you know the TABLES with the heaviest WEIGHTS, what is the heavy SQL driving I/O to the heavy tables?
 - STMT_TEXT like %TABLE_NAME% has some limitations
- What are the HEAVIEST SQL – By table? Across the DB?
 - CPU %
 - Rows Read %
 - Logical Reads %
 - Physical Reads %
 - Rows Written %
 - Execution Time %
 - Sort Time %

7

For relative weights to be invaluable, you have to know the total aggregate costs of execution independent of any literal values that might be present in the SQL. For static SQL, and dynamic SQL with parameter markers, or if you have the STMT_CONC = LITERALS database configuration parameter set, cost aggregation is done correctly by DB2. But, the majority of SQL in the world is dynamic with literals, and only a very small percentage (about 10% in a recent DB2Night Show audience survey) of DB2 customers use the statement concentrator due to potential adverse side effects.

SQL HEAVY WEIGHTS by CPU Time (microseconds)

```

SELECT
  CAST( ( ( (A.TOTAL_USR_CPU_TIME * 1000000) + A.TOTAL_USR_CPU_TIME_MS
    + (A.TOTAL_SYS_CPU_TIME * 1000000) + A.TOTAL_SYS_CPU_TIME_MS )
    / A.NUM_EXECUTIONS )
    AS DECIMAL (15,0)) AS AVG_CPU_TIME_MS,
  CAST (A.NUM_EXECUTIONS AS INTEGER) AS NUM_EXECS,
  CAST(((
    ((A.TOTAL_USR_CPU_TIME * 1000000) + A.TOTAL_USR_CPU_TIME_MS
    + (A.TOTAL_SYS_CPU_TIME * 1000000) + A.TOTAL_SYS_CPU_TIME_MS)
    * 100.0)
    / (select (SUM(B.TOTAL_USR_CPU_TIME) * 1000000)
    + (SUM(B.TOTAL_SYS_CPU_TIME) * 1000000)
    + SUM(B.TOTAL_USR_CPU_TIME_MS)
    + SUM(B.TOTAL_SYS_CPU_TIME_MS)
    + 1.0
    FROM SYSIBMADM.SNAPDYN_SQL B
    WHERE A.DBPARTITIONNUM = B.DBPARTITIONNUM
    )) AS DECIMAL(5,2)) AS PCT_CPU_TIME,
  SUBSTR(A.STMT_TEXT,1,100) AS CPU_SUCKING_SQL
FROM SYSIBMADM.SNAPDYN_SQL A
WHERE A.NUM_EXECUTIONS > 0
ORDER BY A.DBPARTITIONNUM ASC, 3 DESC, 1 DESC FETCH FIRST 25 ROWS ONLY;

```

- SQL Analysis - SQL with Execution Times: SQLCPUTIME.SQL
- Results best viewed with command Window 200 characters wide
- Author: Scott.Hayes@DBIsoftware.com
- Version: 1.0
- Last Updated: 2014-03-12
- Copyright 2014 DBI. All Rights Reserved.
- Licensed for use by paid IDUG 2014 Attendees and Authorized DBI Customers Only

```

SELECT
  CAST((( (A.TOTAL_USR_CPU_TIME * 1000000) +
A.TOTAL_USR_CPU_TIME_MS
    + (A.TOTAL_SYS_CPU_TIME * 1000000) +
A.TOTAL_SYS_CPU_TIME_MS )
    / A.NUM_EXECUTIONS )
    AS DECIMAL (15,0)) AS AVG_CPU_TIME_MS,
  CAST (A.NUM_EXECUTIONS AS INTEGER) AS NUM_EXECS,
  CAST(((
    ((A.TOTAL_USR_CPU_TIME * 1000000) +
A.TOTAL_USR_CPU_TIME_MS

```

```

+ (A.TOTAL_SYS_CPU_TIME * 1000000) + A.TOTAL_SYS_CPU_TIME_MS)
* 100.0)
/ (Select (SUM(B.TOTAL_USR_CPU_TIME) * 1000000) +
(SUM(B.TOTAL_SYS_CPU_TIME) * 1000000)
+ SUM(B.TOTAL_USR_CPU_TIME_MS) +
SUM(B.TOTAL_SYS_CPU_TIME_MS) + 1.0
FROM SYSIBMADM.SNAPDYN_SQL B
WHERE A.DBPARTITIONNUM = B.DBPARTITIONNUM
)) AS DECIMAL(5,2)) AS PCT_CPU_TIME,
SUBSTR(A.STMT_TEXT,1,100) AS CPU_SUCKING_SQL
FROM SYSIBMADM.SNAPDYN_SQL A
WHERE A.NUM_EXECUTIONS > 0
ORDER BY A.DBPARTITIONNUM ASC, 3 DESC, 1 DESC FETCH FIRST 25 ROWS
ONLY;

```

SQL HEAVY WEIGHTS by CPU Time (microseconds) - Examples

Brother-Panther@ - db2admin@win7srv1:50000/PRODD891

File Edit View Tools Reports Window Help

Statement Performance for WIN7SRV1:50000/PRODD891

Statement Workload from 3/12/14 12:00 AM to 3/12/14 3:00 AM

Follow Up	Stmt ID	Verb	Type	Exec Time (sec)	Avg Exec Time (sec)	% Exec Time	# Execs	CPU Time (sec)	Avg CPU Time (sec)	CPU Cost (\$)	% CPU Time
	644F60DF8...	SELECT	DYNAMIC	2.097 979211	0.925035	19.160%	2,268	886.959273	0.391076	\$1,773.9185	25.292%
	64D382EB4...	SELECT	DYNAMIC	1.761 873208	0.789724	16.090%	2,231	747.728393	0.335154	\$1,495.4568	21.322%
	75829C047...	SELECT	DYNAMIC	2.368 219343	0.986758	21.628%	2,400	571.852871	0.238272	\$1,143.7057	16.307%

AVG_CPU_TIME_MS	NUM_EXECS	PCT_CPU_TIME	CPU_SWCHKING_SQL
134767.	36	14.98	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = '204.62.53.126' fetch first 10
128845.	27	10.48	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = '625.net.csl.searches.com' fet
1575610.	1	4.83	select dayofweek(hittimestamp), count(*) from dbipoc_website_data_t4 group by dayofweek(hittimestamp) order by
131467.	9	2.68	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = 'cpovill.net.csl.searches.com'
128343.	7	2.58	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = '89.132.29.77' fetch first 10
179881.	6	2.52	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = 'ops188.searches.com' fetch
674884.	2	2.46	select domainname, count(*) from DBIPOC_WEBSITE_DATA_T4 group by domainname order by 2 desc fetch first 10 row
111880.	6	2.46	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = '65.55.213.36' fetch first 10
157921.	4	1.95	select protocol, targetfile from DBIPOC_FAILED_HITS_UW where domainname = '61.136.154.215' fetch first 10 row
128701.	4	1.58	select protocol, targetfile from DBIPOC_FAILED_HITS_UW where domainname = 'ip-224-26.csmay.net' fetch first 1
135781.	2	1.28	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = '204.62.53.127' fetch first 10
124881.	4	1.53	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = 'custome-GDI-41-281.negored.n
158881.	3	1.38	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = 'ec2-74-127-111-17.compute-1.
130880.	1	1.19	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = 'ec2-67-62-48-105.compute-1.
179481.	1	1.10	select protocol, authid, dayofweek(hittimestamp), targetfile from DBIPOC_FAILED_HITS_UW where domainname = 'so
148281.	2	0.91	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = '204.62.53.129' fetch first 10
148281.	2	0.91	select protocol, targetfile from DBIPOC_FAILED_HITS_UW where domainname = '61.136.154.78' fetch first 10 row
265281.	1	0.91	select month(hittimestamp), count(*) from dbipoc_website_data_t4 group by month(hittimestamp) order by 1 fetch
132680.	1	0.81	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = '288.111.154.249' fetch first
249681.	1	0.76	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = 'asnbot-55-55-165-16.search.ms
124881.	2	0.76	select actionverb, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where domainname = 'ec2-57-245-41-52.compute-1.
234882.	1	0.71	select ipaddr, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where hittimestamp = '2009-03-11-11-45-43.000000' ord
218482.	1	0.67	select ipaddr, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where hittimestamp = '2009-08-05-05-04-52.000000' ord
218482.	1	0.67	select ipaddr, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where hittimestamp = '2009-10-04-21-57-08.000000' ord
218482.	1	0.67	select ipaddr, targetfile from DBIPOC_SUCCESSFUL_HITS_UW where hittimestamp = '2009-03-13-20-13-36.000000' ord

25 record(s) selected.

- SQL Analysis - SQL with Execution Times: SQLCPUTIME.SQL
- Results best viewed with command Window 200 characters wide
- Author: Scott.Hayes@DBISoftware.com
- Version: 1.0
- Last Updated: 2014-03-12
- Copyright 2014 DBI. All Rights Reserved.
- Licensed for use by paid IDUG 2014 Attendees and Authorized DBI Customers Only

SELECT

```

CAST( ( ( (A.TOTAL_USR_CPU_TIME * 1000000) +
A.TOTAL_USR_CPU_TIME_MS
+ (A.TOTAL_SYS_CPU_TIME * 1000000) +
A.TOTAL_SYS_CPU_TIME_MS )
/ A.NUM_EXECUTIONS )
AS DECIMAL (15,0)) AS AVG_CPU_TIME_MS,
CAST (A.NUM_EXECUTIONS AS INTEGER) AS NUM_EXECS,
CAST(((
((A.TOTAL_USR_CPU_TIME * 1000000) +
A.TOTAL_USR_CPU_TIME_MS

```

```

+ (A.TOTAL_SYS_CPU_TIME * 1000000) + A.TOTAL_SYS_CPU_TIME_MS)
* 100.0)
/ (Select (SUM(B.TOTAL_USR_CPU_TIME) * 1000000) +
(SUM(B.TOTAL_SYS_CPU_TIME) * 1000000)
+ SUM(B.TOTAL_USR_CPU_TIME_MS) +
SUM(B.TOTAL_SYS_CPU_TIME_MS) + 1.0
FROM SYSIBMADM.SNAPDYN_SQL B
WHERE A.DBPARTITIONNUM = B.DBPARTITIONNUM
)) AS DECIMAL(5,2)) AS PCT_CPU_TIME,
SUBSTR(A.STMT_TEXT,1,100) AS CPU_SUCKING_SQL
FROM SYSIBMADM.SNAPDYN_SQL A
WHERE A.NUM_EXECUTIONS > 0
ORDER BY A.DBPARTITIONNUM ASC, 3 DESC, 1 DESC FETCH FIRST 25 ROWS
ONLY;

```

SQL HEAVY WEIGHTS by Rows Read

```
SELECT CAST (A.NUM_EXECUTIONS AS INTEGER) AS NUM_EXECS,
       CAST( (A.ROWS_READ + 0.001) / (A.NUM_EXECUTIONS + 0.001)
           AS DECIMAL (13,4)) AS AVG_ROWS_READ,
       CAST((((A.ROWS_READ) * 100.0)
           / (Select (SUM(B.ROWS_READ) + 1.0)
               FROM SYSIBMADM.SNAPDYN_SQL B
               WHERE A.DBPARTITIONNUM = B.DBPARTITIONNUM
               )) AS DECIMAL(5,2)) AS PCT_ROWS_READ,
       SUBSTR(A.STMT_TEXT,1,110) AS HEAVY_READER_SQL
FROM SYSIBMADM.SNAPDYN_SQL A
WHERE A.ROWS_READ > 0
      AND A.NUM_EXECUTIONS > 0
ORDER BY A.DBPARTITIONNUM ASC, 3 DESC, 2 DESC FETCH FIRST 25 ROWS ONLY;
```

10
10

```
-- SQL Analysis - Heavy Read I/O SQL: SQLROWSREAD.SQL
-- Results best viewed with command Window 200 characters wide
-- Author: Scott.Hayes@DBIsoftware.com
-- Version: 1.0
-- Last Updated: 2014-03-13
-- Copyright 2014 DBI. All Rights Reserved.
-- Licensed for use by paid IDUG 2014 Attendees and Authorized DBI Customers
Only
```

```
SELECT CAST (A.NUM_EXECUTIONS AS INTEGER) AS NUM_EXECS,
       CAST( (A.ROWS_READ + 0.001) / (A.NUM_EXECUTIONS + 0.001)
           AS DECIMAL (13,4)) AS AVG_ROWS_READ,
       CAST((((A.ROWS_READ) * 100.0)
           / (Select (SUM(B.ROWS_READ) + 1.0)
               FROM SYSIBMADM.SNAPDYN_SQL B
               WHERE A.DBPARTITIONNUM = B.DBPARTITIONNUM
               )) AS DECIMAL(5,2)) AS PCT_ROWS_READ,
       SUBSTR(A.STMT_TEXT,1,110) AS HEAVY_READER_SQL
FROM SYSIBMADM.SNAPDYN_SQL A
```

```
WHERE A.ROWS_READ > 0 AND A.NUM_EXECUTIONS > 0  
ORDER BY A.DBPARTITIONNUM ASC, 3 DESC, 2 DESC FETCH FIRST 25 ROWS  
ONLY;
```

SQL HEAVY WEIGHTS by Rows Read - Examples

```

Statement Performance for W07SRV15000/PRODD091
Statement Workload from 3/12/14 10:15 PM to 3/13/14 1:15 AM

NUM_EXECS  AVG_ROWS_READ  PCT_ROWS_READ  HEAVY_READER_SQL
-----
36 753328.0742 4.25 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = '204.62.53.126' fetch first 10
27 753321.0992 3.19 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = 'v35.nat.svl.searchme.com' fet
25 619573.2171 2.43 select ipaddr, actionverb, protocol from DBIPOC.WEBSITE_DATA_TB where domainname = 'v35.nat.svl.searchme.com'
18 753222.5727 1.18 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = 'elinet.edu.pl' fetch first 10
10 753273.6727 1.18 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'sowa.com.nv' fetch first 10 rows
9 753265.3839 1.06 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = 'craovi.nat.svl.searchme.com'
8 753254.8432 0.94 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = '61.135.134.215' fetch first 10 ro
8 753254.8432 0.94 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = '45.55.219.36' fetch first 10
7 753241.3942 0.82 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = '89.122.29.77' fetch first 10
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = '91.192.20.154' fetch first 10 row
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'msnbot-65-55-106-163.search.msn.c
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'ip-234-26.csmmap.net' fetch first
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'msnbot-65-55-211-64.search.msn.co
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'msnbot-65-55-106-114.search.msn.c
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = 'ops8-r106.searchme.com' fetch
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'msnbot-65-55-211-74.search.msn.co
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = '210.197.154.17' fetch first 1
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'msnbot-65-55-211-75.search.msn.co
6 753223.4629 0.70 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'msnbot-65-55-106-135.search.msn.c
5 753198.3685 0.59 select ipaddr, actionverb, protocol from DBIPOC.WEBSITE_DATA_TB where domainname = 'elinet.edu.pl'
5 753198.3685 0.59 select actionverb, targetfile from DBIPOC.SUCCESSFUL_HITS_WU where domainname = 'ec2-67-202-48-105.compute-1.a
4 753168.7180 0.47 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'ec2-194-129-111-17.compute-1.
4 753168.7180 0.47 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'msnbot-65-55-106-102.search.msn.c
4 753168.7180 0.47 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'msnbot-65-55-165-79.search.msn.co
4 753168.7180 0.47 select actionverb, targetfile from DBIPOC.FAILED_HITS_WU where domainname = 'baserver.proservis.net' fetch fir

25 record(s) selected.
  
```

- SQL Analysis - Heavy Read I/O SQL: SQLROWSREAD.SQL
- Results best viewed with command Window 200 characters wide
- Author: Scott.Hayes@DBISoftware.com
- Version: 1.0
- Last Updated: 2014-03-13
- Copyright 2014 DBI. All Rights Reserved.
- Licensed for use by paid IDUG 2014 Attendees and Authorized DBI Customers Only

```

SELECT CAST (A.NUM_EXECUTIONS AS INTEGER) AS NUM_EXECS,
       CAST( (A.ROWS_READ + 0.001) / (A.NUM_EXECUTIONS + 0.001)
           AS DECIMAL (13,4)) AS AVG_ROWS_READ,
       CAST((((A.ROWS_READ) * 100.0)
           / (Select (SUM(B.ROWS_READ) + 1.0)
           FROM SYSIBMADM.SNAPDYN_SQL B
           WHERE A.DBPARTITIONNUM = B.DBPARTITIONNUM
           )) AS DECIMAL(5,2)) AS PCT_ROWS_READ,
       SUBSTR(A.STMT_TEXT,1,110) AS HEAVY_READER_SQL
FROM SYSIBMADM.SNAPDYN_SQL A
  
```

```
WHERE A.ROWS_READ > 0 AND A.NUM_EXECUTIONS > 0  
ORDER BY A.DBPARTITIONNUM ASC, 3 DESC, 2 DESC FETCH FIRST 25 ROWS  
ONLY;
```



IDUG DB2 EMEA Tech Conference
Dublin, Ireland | November 2015

#IDUGDB2

“If you call someone and get voicemail,
LEAVE A MESSAGE!”
-Scott Hayes

#WISDOM



Doodle here.

#1 SOLUTION: Having the RIGHT indexes to lower execution cost!

- **Good Indexes:**
 - Are Used!
 - Have High FULLKEYCARD compared to TBCARD
 - > 5% giving a rid list of 20 or fewer RIDs for each distinct column value combination (ROT)
 - Satisfy the Sargable Predicates (\geq , \leq , $=$, $>$, $<$) of WHERE clauses
 - Reduce Scans & Sorts

	Yeast bread tips, 3 ·37
	Yeast Doughnuts, 3 ·55
	Breakfast Pizza, 8 ·155
	Brioche, 3 ·52
	British-Style Burgers, 11 ·243
	Broccoli
	Broccoli-Carrot Stir-Fry, 18 ·407
	Broccoli-Onion
44	Casserole*, 18 ·409
	Broccoli Oriental, 18 ·395
	Broccoli Puff, 8 ·153
	Broccoli-Stuffed Sole*, 9 ·171
	Cauliflower-Broccoli
	Bake*, 18 ·408
	Fettuccine with Broccoli and
	Cheese Sauce*, 8 ·158
	Golden Broccoli Bake*, 18 ·395
	Pasta Primavera, 12 ·268
	Broiled Coconut Topping, 4 ·90
	Broth
	Beef Broth, 17 ·372
	Chicken Broth, 17 ·373

Query SYSCAT.INDEXES LASTUSED column to see when an index was last used. This works reasonably well for DB2 9.7 FP5 and higher. The column is updated periodically and automatically by DB2.

#1 Solution – Part 2 NOT HAVING BAD (Low Cardinality) INDEXES!

```
select substr(a.tabschema,1,8) as schema,
       substr(a.tabname,1,20) as table,
       substr(a.indschema,1,8) as indschema,
       substr(a.indname,1,20) as index,
       a.fullkeycard as IXFULLKEYCARD,
       b.card as TBCARD,
       int((float(a.fullkeycard)/float(b.card)) * 100) as ratio,
       a.lastused as LAST_USED
from SYSCAT.INDEXES A inner join SYSCAT.TABLES B
  on A.tabschema = B.tabschema
  and A.tabname = B.tabname
where A.fullkeycard > 0
      and A.tabschema <> 'SYSIBM'
      and B.card > 100
      and A.uniquerule <> 'U'
      and int((float(a.fullkeycard)/float(b.card)) * 100) < 5
      and A.tabname in
      (SELECT C.TABNAME FROM sysibmadm.snaptab C
       order by C.ROWS_WRITTEN DESC fetch first 10 ROWS ONLY)
order by 7 ASC;
```

14
14

-- ANALYZE Indexes for Low IXCARD on High Write Tables:
IXLOWCARDV3.SQL

-- INDEXES must be HIGH QUALITY on Top 10 Write I/O Tables

-- Results best viewed with command Window ** 200 characters wide **

-- Author: Scott.Hayes@DBIsoftware.com

-- Version: 3.0

-- Last Updated: 2014-03-14

-- Copyright 2014 DBI. All Rights Reserved.

-- Licensed for use by paid IDUG Attendees and DBI Authorized Customers
Only

--

-- For the top 10 most highly written to tables, indentify the indexes having
very low cardinality

-- compared to the table cardinality.

```
select substr(a.tabschema,1,8) as schema, substr(a.tabname,1,20) as table,
       substr(a.indschema,1,8) as indschema, substr(a.indname,1,20) as index,
       a.fullkeycard as IXFULLKEYCARD, b.card as TBCARD,
```

```
        int((float(a.fullkeycard)/float(b.card)) * 100) as ratio,  
        a.lastused as LAST_USED  
from SYSCAT.INDEXES A inner join SYSCAT.TABLES B  
    on A.tabschema = B.tabschema  
    and A.tabname = B.tabname  
where A.fullkeycard > 0  
--    and A.tabschema <> 'SYSIBM'  
    and B.card > 100 and A.uniquerule <> 'U'  
    and int((float(a.fullkeycard)/float(b.card)) * 100) < 5  
    and A.tabname in  
        (SELECT C.TABNAME FROM sysibmadm.snaptab C  
         order by C.ROWS_WRITTEN DESC fetch first 10 ROWS ONLY)  
order by 7 ASC;
```

#1 Solution – Part 2 NOT HAVING BAD INDEXES! BAD EXAMPLE!

SCHEMA	TABLE	INDSCHEMA	INDEX	IXFULLKEYCARD	TBCARD	RATIO	LAST_USED
SYSTEM	SVSCOLUMNS	SYSTEM	INDCOLUMNS02	15	10648	0	01/01/0001
SYSTEM	SVSCOLUMNS	SYSTEM	INDCOLUMNS03	1	10648	0	01/01/0001
SYSTEM	SVSTABLES	SYSTEM	INDTABLES03	1	662	0	01/01/0001
SYSTEM	SVSTABLES	SYSTEM	INDTABLES04	1	662	0	01/01/0001
SYSTEM	SVSINDEXES	SYSTEM	INDINDEXES03	1	440	0	03/13/2014
SYSTEM	SVSTABLES	SYSTEM	INDTABLES07	1	662	0	01/01/0001
SYSTEM	SVSTABLES	SYSTEM	INDTABLES08	3	662	0	01/01/0001
SYSTEM	SVSTABLES	SYSTEM	INDTABLES09	1	662	0	09/04/2012
SYSTEM	SVSDATAPARTITIONS	SYSTEM	INDDATAPARTITIONS06	6	371	1	04/17/2012
SYSTEM	SVSTABLES	SYSTEM	INDTABLES02	7	662	1	01/01/0001
SYSTEM	SVSPPLAN	SYSTEM	INDPLAN02	6	555	1	04/17/2012
SYSTEM	SVSDATAPARTITIONS	SYSTEM	INDDATAPARTITIONS04	6	371	1	01/01/0001

12 record(s) selected.

-- ANALYZE Indexes for Low IXCARD on High Write Tables:
IXLOWCARDV3.SQL

-- INDEXES must be HIGH QUALITY on Top 10 Write I/O Tables

-- Results best viewed with command Window ** 200 characters wide **

-- Author: Scott.Hayes@DBIsoftware.com

-- Version: 3.0

-- Last Updated: 2014-03-14

-- Copyright 2014 DBI. All Rights Reserved.

-- Licensed for use by paid IDUG Attendees and DBI Authorized Customers
Only

--

-- For the top 10 most highly written to tables, identify the indexes having
very low cardinality

-- compared to the table cardinality.

```
select substr(a.tabschema,1,8) as schema, substr(a.tabname,1,20) as table,
       substr(a.indschema,1,8) as indschema, substr(a.indname,1,20) as index,
       a.fullkeycard as IXFULLKEYCARD, b.card as TBCARD,
```

```
int((float(a.fullkeycard)/float(b.card)) * 100) as ratio,  
a.lastused as LAST_USED  
from SYSCAT.INDEXES A inner join SYSCAT.TABLES B  
on A.tabschema = B.tabschema  
and A.tabname = B.tabname  
where A.fullkeycard > 0  
-- and A.tabschema <> 'SYSIBM'  
and B.card > 100 and A.uniquerule <> 'U'  
and int((float(a.fullkeycard)/float(b.card)) * 100) < 5  
and A.tabname in  
  (SELECT C.TABNAME FROM sysibmadm.snaptab C  
   order by C.ROWS_WRITTEN DESC fetch first 10 ROWS ONLY)  
order by 7 ASC;
```

More about INDEXES V10.1 & V10.5

- Jump Scans introduced in V10.1
 - A more “efficient” means for using sub-optimal indexes?
 - V9.7 to V10.1 upgrade of E-COM DB gave 13 out of 4,267 unique SQL patterns with lower LREAD costs and improved results.
 - Contrary to Marketing, Jump Scans are NOT a license to be lazy!
- V10.5
 - Indexes on Expressions! YAY! DB2 caught up to Oracle 8i (LOL)
 - **CREATE INDEX** EMPUP **ON** EMPLOYEE (**UPPER**(LASTNAME), EMPNO)
 - New RANDOM option specifies that index entries are to be kept in random order of the column values
 - Cannot be used with CLUSTER indexes, ALLOW REVERSE SCANS
 - Create Index MYScheme.MYIX on MYScheme.MYTable (MY_AWESOME_COLUMN **RANDOM**);
 - Might mitigate the need to use **PAGE SPLIT HIGH** if queries have only = predicates

16

Indexes on Expressions via IBM docs:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.sql.ref.doc/doc/r0000919.html>

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.admin.dboj.doc/doc/c0061101.html>

The RANDOM option is intended for use by pureScale customers that experience contention/hot spots in indexes on high keys. I don't see much else practical use for it since ASC or DESC sequences can help avoid sorts and improve efficiency for accessing ranges (>, <).

Is DB2 BLU Right for YOU? ORGANIZE BY COLUMN TABLES – “Load & Go”

- Do you have tables that are very heavily READ?
 - Do these have minimal, or zero, WRITE Activity?
 - Are they free from obnoxious unsupported data types like LOBs?
 - Does the database not violate any restrictions?
 - pureScale®, partitioned databases, DB w/out automatic storage
 - Tablespaces not enabled for reclaimable storage, ~~no~~ HADR
 - Must be UNICODE and IDENTITY or IDENTITY16
 - Prohibited: STMM automatic tuning of SORT memory
 - No use of DB2_COMPATIBILITY_VECTOR
 - Others:
<http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.admin.dboobj.doc/doc/c0060592.html>
 - Do the SQL queries reference very few columns, generally?
 - Do SQL queries do a lot of grouping, aggregation, summarizing?
- ... Then DB2 BLU 10.5 ORG by COL might be a great solution!

17

Updated IBM docs:

http://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.admin.dboobj.doc/doc/c0060592.html

Limitations:

http://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.admin.dboobj.doc/doc/c0061528.html

The DB2Night Show™ had DB2 BLU Fest during the fall (SEPT-OCT) of 2013. Check out recorded replays of these DB2 BLU shows at <http://www.DBISoftware.com/blog/db2nightshow.php>

#114, DB2 BLU Early Experiences and Best Practices (IBM Canada Lab)

#115, DB2 BLU & pureScale Real Life Successes and Advice (Kent Collins, DB2 User & Consultant)

#116, Intimate Details – DB2 BLU with IBM Master Inventors (IBM Almaden Lab)

#117, Feelin' BLU – The HOTTEST IBM DB2 BLU video ever (Scott Hayes, DBI, & Randall Ibbott, QBE)

Also of potential interest:

#118, 17 Laws of building very large databases! (Lee Goddard, DBI, formerly IBM)

Take a picture of your luggage before you fly-
- easy description when lost
- documents bag condition

● #WISDOM



How would you describe this piece of luggage?

Hell on wheels in a black box?

What other WEIGHTS can we look at?



IDENTIFYING HEAVY WEIGHTS

18

We've looked at the weight of tables plus SQL by CPU and Rows Read (Review from Sage Advice Part 1), and Part 1 also covered SQL weights by Logical Reads, Physical Reads, Rows Written, Execution Time, and Sort Time (see session D06 of IDUG NA Phoenix). Also, alternatively, watch a replay of The DB2Night Show Episode #142 to see and hear the entire presentation: <http://www.dbisoftware.com/blog/db2nightshow.php?id=528>

In this section of the presentation, we will contemplate some additional relative weights that merit analysis.

MON_GET_INDEXES

Caution: Provides Data *ONLY* Since Database Activation

TABSCHEMA	TABNAME	INDSHEMA	INDNAME	MEMBER	INDEX_ONLY_SCANS	INDEX_SCANS	PCT_INDEX_SCANS
SYSIBM	SYSTABLES	SYSIBM	INDTABLES01	0	13	658	50.46 ✓
SYSIBM	SYSSEVENTMONITORS	SYSIBM	INDEVENTMONITORS01	0	23	71	5.44
SYSIBM	SYSROUTINES	SYSIBM	INDROUTINES06	0	0	65	4.98
SYSIBM	SYSINDEXES	SYSIBM	INDINDEXES04	0	64	64	4.90
SYSIBM	SYSTABLES	SYSIBM	INDTABLES06	0	56	56	4.29
SYSIBM	SYSROUTINES	SYSIBM	INDROUTINES02	0	0	49	3.75
SYSIBM	SYSMODULES	SYSIBM	INDMODULES02	0	0	36	2.76
SYSIBM	SYSVARIABLES	SYSIBM	INDVARIABLES03	0	0	32	2.45
DB2ADMIN	ADVISE_INDEX	DB2ADMIN	IDX_I2	0	0	27	2.07
SYSIBM	SYSTASKS	SYSIBM	INDTASKS04	0	0	22	1.68
SYSIBM	SYSSECTION	SYSIBM	INDSECTION01	0	0	20	1.53
SYSIBM	SYSSTMT	SYSIBM	INDSTMT01	0	0	20	1.53
DB2ADMIN	EXPLAIN_STATEMENT	DB2ADMIN	STMT_I1	0	0	14	1.07
SYSIBM	SYSAUDITUSE	SYSIBM	INDAUDITUSE01	0	0	14	1.07
DB2ADMIN	ADVISE_INDEX	DB2ADMIN	IDX_ADVISE_IDX_RUN_	0	7	11	0.84
SYSIBM	SYSPLAN	SYSIBM	INDPLAN01	0	0	10	0.76
SYSIBM	SYSPLAN	SYSIBM	INDPLAN04	0	9	9	0.69

Query to find the heaviest weight INDEX_SCANS:

```
SELECT VARCHAR(T.TABSCHEMA, 18) AS TABSCHEMA,
       VARCHAR(T.TABNAME, 18) AS TABNAME,
       VARCHAR(SI.INDSHEMA, 18) AS INDSHEMA,
       VARCHAR(SI.INDNAME, 18) AS INDNAME,
       T.MEMBER,
       T.INDEX_ONLY_SCANS,
       T.INDEX_SCANS,
       CAST((((T.INDEX_SCANS) * 100) / (SELECT
(SUM(MI.INDEX_SCANS) + 1.0)
          FROM TABLE(MON_GET_INDEX('', -2)) as MI ))
          AS DECIMAL(5,2)) AS PCT_INDEX_SCANS
-- (MON_GET_INDEX('SCHEMA','TABLE', MEMBER))
FROM TABLE(MON_GET_INDEX('', -2)) as T,
       SYSCAT.INDEXES AS SI
WHERE T.TABSCHEMA = SI.TABSCHEMA AND
      T.TABNAME = SI.TABNAME AND
```

T.IID = SI.IID
ORDER BY INDEX_SCANS DESC

MON_GET_INDEXES

What is an INDEX_SCAN?

- An INDEX_SCAN means that DB2 “looked” at one or more pages of the index.
 - It does not tell you if the B-Tree was navigated (hop, hop, hop) or if ALL or MANY of the LEAF pages were scanned. **Bummer!**
 - To learn the behavior of index access, we have to look at Bufferpool_Logical_Index_Reads
 - Available at DB, BP, and TS level, but insufficient granularity unless you have your indexes isolated to separate tablespaces
 - Also available at the SQL performance level for individual statements!
 - AVG_IX_L_READS per Execution should be in range 1-100
 - If > 100, SQL is doing Index Leaf Page Scan, and Explain should reveal the Index being scanned
 - If 0, then SQL is doing a table scan!

SQL HEAVY WEIGHTS by INDEX LOGICAL READS:

```
SELECT CAST( (A.POOL_INDEX_L_READS + 0.01) /  
(A.NUM_EXECUTIONS + 0.01)  
AS DECIMAL (13,2)) AS IXLREAD_PER_EXEC,  
SUBSTR(A.STMT_TEXT,1,180) AS PROBABLE_LEAF_SCAN_SQL  
  
FROM SYSIBMADM.SNAPDYN_SQL A  
ORDER BY A.DBPARTITIONNUM ASC, 1 DESC FETCH FIRST 25 ROWS  
ONLY;
```

MON_GET_INDEXES

So what good is it? ***DANGER*** Finding Unused Indexes

Execute SQL: db2admin@WIN7SRV1:50000/PRODD891

```

SELECT VARCHA (T. TABSCHE
VARCHA (T. TABNAME
VARCHA (SI. INDSCH
VARCHA (SI. INDNAM
T. MEMBER,
T. INDEX_SCANS
-- (MON_GET_INDEX('SCHEM
FROM TABLE (MON_GET_IND
SYSCAT.INDEXES AS
WHERE T. TABSCHEMA = SI
T. TABNAME = SI. TABNA
T. IID = SI. IID AND
T. INDEX_SCANS = 0
ORDER BY INDEX_SCANS D
  
```

TABSCHEMA	TABNAME	INDSCHEMA	INDNAME	MEMBER	INDEX_SCANS
DB2ADMIN	ADVISE_MQT	DB2ADMIN	MQT_I1	0	0
DB2ADMIN	ADVISE_MQT	DB2ADMIN	MQT_I2	0	0
DB2ADMIN	ADVISE_PA...	DB2ADMIN	PRT_I1	0	0
SYSIBM	SYSTABLES	SYSIBM	INDTAB...	0	0
SYSIBM	SYSTABLES	SYSIBM	INDTAB...	0	0
SYSIBM	SYSTABLES	SYSIBM	INDTAB...	0	0
SYSIBM	SYSTABLES	SYSIBM	INDTAB...	0	0
SYSIBM	SYSTABLES	SYSIBM	INDTAB...	0	0
SYSIBM	SYSTABLES	SYSIBM	INDTAB...	0	0
SYSIBM	SYSCOLUMNS	SYSIBM	INDCOL...	0	0
SYSIBM	SYSCOLUMNS	SYSIBM	INDCOL...	0	0
SYSIBM	SYSVIEWS	SYSIBM	INDVIE...	0	0

DANGER Remember, data is only current as of db2start!! Don't know if I'd trust this unless DB2 has been up an entire year to observe all business cycles!

```

SELECT VARCHA(T.TABSCHEMA, 18) AS TABSCHEMA,
       VARCHA(T.TABNAME, 18) AS TABNAME,
       VARCHA(SI.INDSCHEMA, 18) AS INDSHEMA,
       VARCHA(SI.INDNAME, 18) AS INDNAME,
       T.MEMBER,
       T.INDEX_SCANS
-- (MON_GET_INDEX('SCHEMA','TABLE', MEMBER))
FROM TABLE(MON_GET_INDEX(",", -2)) as T,
       SYSCAT.INDEXES AS SI
WHERE T.TABSCHEMA = SI.TABSCHEMA AND
      T.TABNAME = SI.TABNAME AND
      T.IID = SI.IID AND
      T.INDEX_SCANS = 0
  
```

ORDER BY INDEX_SCANS DESC

#1 Solution – Part 2 – ENHANCED by MON_GET_INDEXES NOT HAVING BAD (Low Cardinality, Rare Use) INDEXES!

SCHEMA	TABLE	INDSCHEMA	INDEX	IXFULLKEYCARD	TBCARD	RATIO	LAST_USED	INDEX_SCANS
SYSIBM	SYSROUTINES	SYSIBM	INDROUTINES04	3	1036	0	0001-01-01	0
SYSIBM	SYSCOLUMNS	SYSIBM	INDCOLUMNS03	1	10648	0	0001-01-01	0
SYSIBM	SYSCOLUMNS	SYSIBM	INDCOLUMNS02	15	10648	0	0001-01-01	0
SYSIBM	SYSINDEXES	SYSIBM	INDINDEXES03	1	440	0	2015-03-04	8 ✓
SYSIBM	SYSTABLES	SYSIBM	INDTABLES09	1	662	0	2012-09-04	0
SYSIBM	SYSTABLES	SYSIBM	INDTABLES08	3	662	0	0001-01-01	0
SYSIBM	SYSTABLES	SYSIBM	INDTABLES07	1	662	0	0001-01-01	0
SYSIBM	SYSTABLES	SYSIBM	INDTABLES04	1	662	0	0001-01-01	0
SYSIBM	SYSTABLES	SYSIBM	INDTABLES03	1	662	0	0001-01-01	0
DBIPOC	WEBSITE_DATA_T...	IDUGNA15	WEBSTATUS	13	753349	0	0001-01-01	0
SYSIBM	SYSPLAN	SYSIBM	INDPLAN02	6	555	1	2012-04-17	0
SYSIBM	SYSDATAPARTITI...	SYSIBM	INDDATAPARTITIO...	6	371	1	2012-04-17	0
SYSIBM	SYSDATAPARTITI...	SYSIBM	INDDATAPARTITIO...	6	371	1	0001-01-01	0
SYSIBM	SYSTABLES	SYSIBM	INDTABLES02	7	662	1	0001-01-01	0
SYSIBM	SYSROUTINES	SYSIBM	INDROUTINES11	47	1036	4	2012-08-03	0

```

select substr(a.tabschema,1,8) as schema, substr(a.tabname,1,20) as table,
       substr(a.indschema,1,8) as indschema, substr(a.indname,1,20) as index,
       a.fullkeycard as IXFULLKEYCARD, b.card as TBCARD,
       int((float(a.fullkeycard)/float(b.card)) * 100) as ratio,
       a.lastused as LAST_USED, MI.INDEX_SCANS
from TABLE(MON_GET_INDEX(",",-2)) as MI, SYSCAT.INDEXES A,
SYSCAT.TABLES B
where A.tabschema = B.tabschema and MI.tabschema = A.tabschema
and A.tabname = B.tabname and MI.tabname = A.tabname
and MI.IID = A.IID
and MI.INDEX_SCANS < 100 and A.fullkeycard > 0
-- and A.tabschema <> 'SYSIBM'
and B.card > 100 and A.uniquerule <> 'U'
and int((float(a.fullkeycard)/float(b.card)) * 100) < 5
and A.tabname in
(SELECT C.TABNAME FROM sysibmadm.snaptab C
 order by C.ROWS_WRITTEN DESC fetch first 20 ROWS ONLY)

```

order by 7 ASC;

So, you found a heavy weight SQL statement, and you passed it to the Design Advisor (db2advis), and the Design Advisor suggests that you create 3, 5, 11, or 13 indexes for a solution!

HOW MANY?

REALLY?

ADVANCED INDEX BENEFIT ANALYSIS

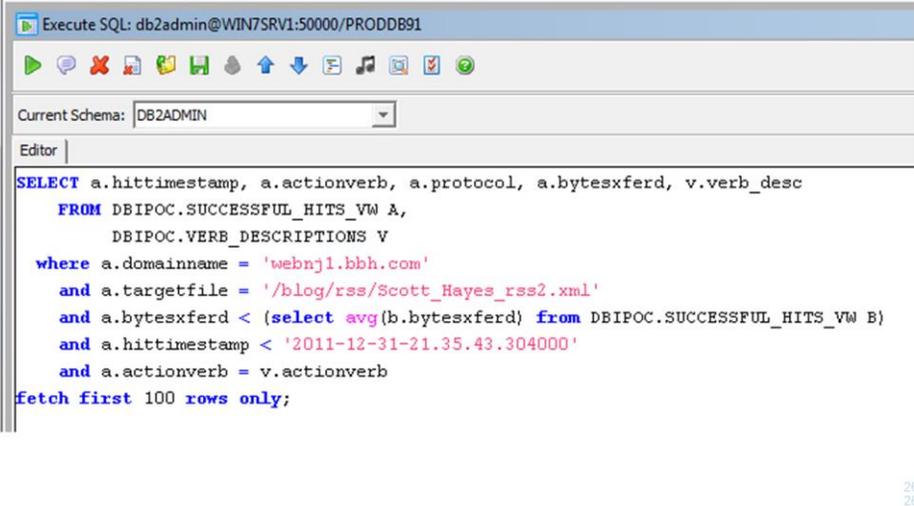
24

Sometimes you might think that db2advis uses drugs! It's awesome when db2advis recommends just one index that gives you a 99% cost reduction, but what about when db2advis wants you to create SEVERAL indexes? OMG!!!

DB2Night Show #168 – Handout Update (This slide not in replay video files)

- The SQL examples in upcoming slides will help you do Index Benefit Analysis, with addition and subtraction, for a SQL workload of ONE SQL statement. If you want to analyze workloads of multiple SQL statements, the math is a bit different (more complicated and more time consuming) and beyond the scope of this presentation.
- **Instead of doing the calculations and analysis by hand, which, according to our polling survey, takes people several minutes to hours, why not just get DBI tools for DB2 LUW instead? You will have complete analysis for one, or multiple, SQL statement workloads within seconds!**

Optimizing Index Solutions Solving a “Heavy” Query



The screenshot shows a DB2 SQL editor window titled "Execute SQL: db2admin@WIN7SRV1:50000/PRODDB91". The current schema is set to "DB2ADMIN". The editor contains the following SQL query:

```
SELECT a.hittimestamp, a.actionverb, a.protocol, a.bytesxferd, v.verb_desc
FROM DBIPOC.SUCCESSFUL_HITS_VW A,
     DBIPOC.VERB_DESCRIPTIONS V
where a.domainname = 'webnj1.bbh.com'
and a.targetfile = '/blog/rss/Scott_Hayes_rss2.xml'
and a.bytesxferd < (select avg(b.bytesxferd) from DBIPOC.SUCCESSFUL_HITS_VW B)
and a.hittimestamp < '2011-12-31-21.35.43.304000'
and a.actionverb = v.actionverb
fetch first 100 rows only;
```

The number 26 is visible in the bottom right corner of the editor window.

The “heavy” query:

```
SELECT a.hittimestamp, a.actionverb, a.protocol, a.bytesxferd, v.verb_desc
FROM DBIPOC.SUCCESSFUL_HITS_VW A,
     DBIPOC.VERB_DESCRIPTIONS V
where a.domainname = 'webnj1.bbh.com'
and a.targetfile = '/blog/rss/Scott_Hayes_rss2.xml'
and a.bytesxferd < (select avg(b.bytesxferd) from
DBIPOC.SUCCESSFUL_HITS_VW B)
and a.hittimestamp < '2011-12-31-21.35.43.304000'
and a.actionverb = v.actionverb
fetch first 100 rows only;
```

Optimizing Index Solutions

Advisor (db2advis) gives 5 Indexes!

5th Index

What is the benefit, or relative value, of Each Index?

185,651?

sigh

We can do better!

Table Schema	Table Name	Index Name	Index Columns
DBIPOC	HTML_STATUS_CODES	IDX1503091819500	+STATUS_DESC-STATUS_CODE
DBIPOC	WEBSITE_DATA_TB	IDX1503091819510	+WEBSTATUS+BYTESXFERD
DBIPOC	WEBSITE_DATA_TB	IDX1503091820070	+DOMAINNAME+TARGETFILE+BYTESXFERD+HITTIMESTAMP+PROTOCOL+ACTIONVERB+WEBSTATUS
DBIPOC	VERB_DESCRIPTIONS	IDX1503091820040	+ACTIONVERB+VERB_DESC
DBIPOC	HTML_STATUS_CODES	IDX1503091819490	+STATUS_CODE-STATUS_DESC

Results from DB2 9.7 FP3 on Windows...

The workload is in ADVISE_WORKLOAD with workload name:
DBI_Analysis_Workload_1425925159390

found 1 statements in the ADVISE_WORKLOAD table

Recommending indexes...

total disk space needed for initial set [23.849] MB

total disk space constrained to [1212.059] MB

Trying variations of the solution set.

5 indexes in current solution

[187411.2500] timerons (without recommendations)

[1760.0568] timerons (with current solution)

[99.06%] improvement

--

--

-- LIST OF RECOMMENDED INDEXES

-- =====

-- index[1], 0.013MB

```

CREATE INDEX "SYSTEM ". "IDX1503091819500" ON "DBIPOC
"."HTML_STATUS_CODES"
("STATUS_DESC" ASC, "STATUS_CODE" DESC) ALLOW REVERSE
SCANS COLLECT SAMPLED DETAILED STATISTICS;
COMMIT WORK ;
-- index[2], 19.657MB
CREATE INDEX "SYSTEM ". "IDX1503091819510" ON "DBIPOC
"."WEBSITE_DATA_TB"
("WEBSTATUS" ASC, "BYTESXFERD" ASC) ALLOW REVERSE
SCANS COLLECT SAMPLED DETAILED STATISTICS;
COMMIT WORK ;
-- index[3], 4.153MB
CREATE INDEX "SYSTEM ". "IDX1503091820070" ON "DBIPOC
"."WEBSITE_DATA_TB"
("DOMAINNAME" ASC, "TARGETFILE" ASC, "BYTESXFERD"
ASC, "HITTIMESTAMP" ASC, "PROTOCOL" ASC, "ACTIONVERB"
ASC, "WEBSTATUS" ASC) ALLOW REVERSE SCANS COLLECT SAMPLED
DETAILED STATISTICS;
COMMIT WORK ;
-- index[4], 0.013MB
CREATE INDEX "SYSTEM ". "IDX1503091820040" ON "DBIPOC
"."VERB_DESCRIPTIONS"
("ACTIONVERB" ASC, "VERB_DESC" ASC) ALLOW REVERSE
SCANS COLLECT SAMPLED DETAILED STATISTICS;
COMMIT WORK ;
-- index[5], 0.013MB
CREATE INDEX "SYSTEM ". "IDX1503091819490" ON "DBIPOC
"."HTML_STATUS_CODES"
("STATUS_CODE" ASC, "STATUS_DESC" DESC) ALLOW REVERSE
SCANS COLLECT SAMPLED DETAILED STATISTICS;
COMMIT WORK ;
--
--

```

-- RECOMMENDED EXISTING INDEXES

-- =====

--

--

-- UNUSED EXISTING INDEXES

-- =====

-- =====

--

-- =====ADVISOR DETAILED XML OUTPUT=====

-- == (Benefits do not include clustering recommendations) ==

--

--<?xml version="1.0"?>

--<design-advisor>

--<index>

--<identifier>

--<name>IDX1503091819500</name>

--<schema>SYSTEM </schema>

--</identifier>

--<table><identifier>

--<name>HTML_STATUS_CODES</name>

--<schema>DBIPOC </schema>

--</identifier></table>

--<statementlist>2</statementlist>

--<benefit>185651.193237</benefit>

--<overhead>0.000000</overhead>

--<diskspace>0.012719</diskspace>

--</index>

--<index>

--<identifier>

--<name>IDX1503091819510</name>

--<schema>SYSTEM </schema>

--</identifier>

```
--<table><identifier>
--<name>WEBSITE_DATA_TB</name>
--<schema>DBIPOC </schema>
--</identifier></table>
--<statementlist>2</statementlist>
--<benefit>185651.193237</benefit>
--<overhead>0.000000</overhead>
--<diskspace>19.657250</diskspace>
--</index>
--<index>
--<identifier>
--<name>IDX1503091820070</name>
--<schema>SYSTEM </schema>
--</identifier>
--<table><identifier>
--<name>WEBSITE_DATA_TB</name>
--<schema>DBIPOC </schema>
--</identifier></table>
--<statementlist>2</statementlist>
--<benefit>185651.193237</benefit>
--<overhead>0.000000</overhead>
--<diskspace>4.153344</diskspace>
--</index>
--<index>
--<identifier>
--<name>IDX1503091820040</name>
--<schema>SYSTEM </schema>
--</identifier>
--<table><identifier>
--<name>VERB_DESCRIPTIONS</name>
--<schema>DBIPOC </schema>
--</identifier></table>
```

```

--<statementlist>2</statementlist>
--<benefit>185651.193237</benefit>
--<overhead>0.000000</overhead>
--<diskspace>0.012719</diskspace>
--</index>
--<index>
--<identifier>
--<name>IDX1503091819490</name>
--<schema>SYSTEM </schema>
--</identifier>
--<table><identifier>
--<name>HTML_STATUS_CODES</name>
--<schema>DBIPOC </schema>
--</identifier></table>
--<statementlist>2</statementlist>
--<benefit>185651.193237</benefit>
--<overhead>0.000000</overhead>
--<diskspace>0.012719</diskspace>
--</index>
--<statement>
--<statementnum>2</statementnum>
--<statementtext>
-- SELECT a.hittimestamp, a.actionverb, a.protocol, a.bytesxferd,
-- v.verb_desc FROM DBIPOC.SUCCESSFUL_HITS_VW A,
-- DBIPOC.VERB_DESCRIPTIONS V where a.domainname
-- = 'webnj1.bbh.com' and a.targetfile = '/blog/rss/Scott_Hayes_rss2.xml'
-- and a.bytesxferd < (select avg(b.bytesxferd) from
-- DBIPOC.SUCCESSFUL_HITS_VW B) and a.hittimestamp
-- < '2011-12-31-21.35.43.304000' and a.actionverb
-- = v.actionverb fetch first 100 rows only
--</statementtext>
--<objects>

```

```
--<identifier>
--<name>HTML_STATUS_CODES</name>
--<schema>DBIPOC </schema>
--</identifier>
--<identifier>
--<name>VERB_DESCRIPTIONS</name>
--<schema>DBIPOC </schema>
--</identifier>
--<identifier>
--<name>WEBSITE_DATA_TB</name>
--<schema>DBIPOC </schema>
--</identifier>
--<identifier>
--<name>IDX1503091819490</name>
--<schema>SYSTEM </schema>
--</identifier>
--<identifier>
--<name>IDX1503091820040</name>
--<schema>SYSTEM </schema>
--</identifier>
--<identifier>
--<name>IDX1503091820070</name>
--<schema>SYSTEM </schema>
--</identifier>
--<identifier>
--<name>IDX1503091819510</name>
--<schema>SYSTEM </schema>
--</identifier>
--<identifier>
--<name>IDX1503091819500</name>
--<schema>SYSTEM </schema>
--</identifier>
```

```
--</objects>
--<benefit>185651.193237</benefit>
--<frequency>1</frequency>
--</statement>
--</design-advisor>
-- =====ADVISOR DETAILED XML OUTPUT=====
--
326 solutions were evaluated by the advisor
DB2 Workload Performance Advisor tool is finished.
```

Optimizing Index Solutions DB2 10.5 db2advise has a different solution set...

The screenshot shows the 'Design Analysis - LPAR21:60018/DBIPOCDB' window. The 'Analysis Report Output' pane displays the following text:

```

Recommending indexes...
total disk space needed for initial set [ 10.9
total disk space constrained to [ 61.9
Trying variations of the solution set.
  5 indexes in current solution
[81524.1953] timerons (without recommendation)
[376.8281] timerons (with current solution)
[99.54%] improvement
  
```

The 'Analysis Messages' pane shows:

```

Operating System = aix lpar21 1 6 000c77acd600
Execution Method = DBI
Advisor Command = db2advise -t 0 -a db2in105/***** -d DBIPOCDB -w DBI_A
Using user id as default schema name. Use -n option to specify schema
  
```

Below the panes is a table titled 'Recommended Indexes':

Table Schema	Table Name	Index Name	Index Columns
DBIPOC	HTML_STATUS_CODES	IDX1503091948080	+STATUS_DESC+STATUS_CODE
DBIPOC	HTML_STATUS_CODES	IDX1503091948080	+STATUS_DESC+STATUS_CODE
DBIPOC	WEBSITE_DATA_TB	IDX1503091948150	+WEBSTATUS-BYTESXFERD
DBIPOC	WEBB_DESCRIPTIONS	IDX1503091948270	+ACTIONWEBB-WEBB_DESC
DBIPOC	WEBSITE_DATA_TB	IDX1503091948290	+DOMAINNAME+TARGETFILE+BYTESXFERD+HITTIMESTAMP+PROTOCOL+ACTIONWEBB+WEBSTATUS

Page numbers 28 and 28 are visible in the bottom right corner of the screenshot.

Results from DB2 10.5.2 on AIX 6.1. We'll proceed with working in this environment because I like AIX better!

Optimizing Index Solutions Relative Benefit Value Analysis

- **Two Methods to Consider**
 - Index Addition – Add indexes one at a time to assess individual value
 - Index Subtraction – Subtract Indexes one at a time from the solution set to assess the value lost
- **Design Advisor can be overly aggressive on Index Only Access**
 - Sometimes additional columns are added to existing indexes to achieve IX Only access – we anticipate these will have less value
 - Give consideration to predicates involved when making final decisions on which indexes to implement

29

The IBM Design Advisor claims to have evaluated 326 solutions to arrive at the proposed solution set. Presuming disk storage is tight (when is it not? (rhetorical)) and time is limited, we will next look at the steps required to determine the relative weighted value of each proposed index.

Optimizing Index Solutions Index Addition 1

- Start with a clean Explain & Advise Environment
 - Delete from Explain_Instance
 - Delete from Advise_Index
- Explain the statement
 - `db2batch -d dbipocdb -f 3Table_Heavy_Query.sql -o e explain`

The “-o e explain” option causes db2batch to Explain the statements but NOT run them!

Optimizing Index Solutions Index Addition 2

- Find the original/"Before" Explain Cost

The screenshot shows two windows of a DB2 SQL editor. The left window displays the following SQL query:

```
select dec(total_cost,20,4) as before_total_cost,
       dec(io_cost,20,4) as io_cost,
       dec(comm_cost,20,4) as comm_cost
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR ;
```

The right window shows the results of the query in a table format:

BEFORE_TOTAL_COST	IO_COST	CPU_COST	COMM_COST
81524.1953	91161.0000	3404550912.0000	0.0000

31

```
select dec(total_cost,20,4) as before_total_cost,
       dec(io_cost,20,4) as io_cost, dec(CPU_cost,20,4) as cpu_cost,
       dec(Comm_cost,20,4) as comm_cost
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR ;
```

Optimizing Index Solutions Index Addition 3A

- Populate the ADVISE_INDEX table
 - `db2advis -t 0 -d DBIPOCDB -i 3Table_Heavy_Query.sql -p`

```
Trying var 5 index
[81524.19]
[376.8281]
[99.54%]

-- LIST OF RECOMMENDED INDEXES
=====
index[1],      0.013MB
CREATE INDEX "DB2IN105"."IDX1503092019020" ON "DBIPOC" "."HTML_STATUS_CODES"
("STATUS_DESC" ASC, "STATUS_CODE" ASC) ALLOW REVERSE
SCANS COLLECT SAMPLED DETAILED STATISTICS;
COMMIT WORK ;
index[2],      7.388MB
CREATE INDEX "DB2IN105"."IDX1503092019090" ON "DBIPOC" "."WEBSITE_DATA_TB"
("WEBSTATUS" ASC, "BYTESXFERD" DESC) ALLOW REVERSE
SCANS COLLECT SAMPLED DETAILED STATISTICS;
COMMIT WORK ;
index[3],      3.501MB
CREATE INDEX "DB2IN105"."IDX1503092019230" ON "DBIPOC" "."WEBSITE_DATA_TB"
("DOMAINNAME" ASC, "TARGETFILE" ASC, "BYTESXFERD"
ASC, "HITTIMESTAMP" ASC, "PROTOCOL" ASC, "ACTIONVERB"
ASC, "WEBSTATUS" ASC) ALLOW REVERSE SCANS COLLECT SAMPLED DETAILED STATISTICS;
COMMIT WORK ;
index[4],      0.013MB
CREATE INDEX "DB2IN105"."IDX1503092019210" ON "DBIPOC" "."VERB_DESCRIPTIONS"
("ACTIONVERB" ASC, "VERB_DESC" DESC) ALLOW REVERSE
SCANS COLLECT SAMPLED DETAILED STATISTICS;
COMMIT WORK ;
```

163 Solutions were evaluated by db2advis.

`db2advis -t 0 -d DBIPOCDB -i 3Table_Heavy_Query.sql`

Optimizing Index Solutions

Index Addition 3B – alternate method

- Populate the ADVISE_INDEX table - CLP
 - db2 "select current explain mode from sysibm.sysdummy1"
 - "NO"
 - db2 "set current explain mode recommend indexes"
 - db2 -stvf 3Table_Heavy_Query.sql
 - Does not execute the query!
 - db2 "set current explain mode NO"
 - So you can run queries again!

For this type of index benefit analysis, I actually prefer this CLP method over db2advis, though I do appreciate how db2advis provides nicely summarized outputs.

Optimizing Index Solutions ADVISE_INDEX Table 1

- USE_INDEX Column – the “magic”
 - ‘Y’ Index Recommended or Evaluated
 - ‘N’ Index not to be Recommended or Evaluated
 - ‘R’ An existing clustering RID index was recommended by Design Advisor to be unclustered – this is the case when a new clustering RID index is recommended for the table
 - ‘I’ Ignore an existing non-unique index for Evaluation. The EXISTS column should be ‘Y’ in this case or the index will not be ignored
- Several other interesting and helpful columns too
 - See sample query and results, next slide

34

Unclustered is a valid word, but spell check wanted it changed to uncluttered. Funny.

Learn about the ADVISE_INDEX table:

http://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.sql.ref.doc/doc/r0002417.html?cp=SSEPGG_10.5.0%2F2-12-13-0&lang=en


IDUG
 Leading the DB2 User Community since 1988

IDUG DB2 EMEA Tech Conference
 Dublin, Ireland | November 2015

 #IDUGDB2

Optimizing Index Solutions

ADVISE_INDEX Table 2

Execute SQL: db2in105@LPAR21:60018/DBIPOCDB

Brother-Panther@ - db2in105@LPAR21:60018/DBIREPOS

File Edit View Tools Reports Window Help

Execute SQL: db2in105@LPAR21:60018/DBIPOCDB

Connections: LPAR21:60018/DBIPOCDB

Current Schema: DB2IN105 Recent SQL: -delete from advise_index[1]

PROPOSED_INDEX	ON_TABLE	EXISTS	USE_INDEX	INDEX_COLS	NLEVELS	NLEAF	UNIQUERULE	FIRSTKEYCARD	FULLKEYCARD
IDX1503092345460	DBIPOC HTML_STATUS_CODES	N	Y	+STATUS_DESC+STATUS_CODE	2	3	D	38	38
IDX1503092345460	DBIPOC HTML_STATUS_CODES	N	Y	+STATUS_DESC+STATUS_CODE	2	3	D	38	38
IDX1503092345530	DBIPOC WEBSITE_DATA_TB	N	Y	+WEBSTATUS-BYTESINFERD	3	1891	D	10	189450
IDX1503092346050	DBIPOC VERB_DESCRIPTIONS	N	Y	+ACTIONVERB-VERB_DESC	2	3	D	12	12
IDX1503092346070	DBIPOC WEBSITE_DATA_TB	N	Y	+DOMAINNAME+TARGETFILE+BYTESINFERD+HITTIMESTAMP+PROTOCOL+ACTIONVERB+WEBSTATUS	3	896	D	134	134

```

select varchar(name,20) as PROPOSED_INDEX,
       concat(tbcreator, tbname) as ON_TABLE,
       EXISTS, USE_INDEX, varchar(colnames,80) as INDEX_COLS,
       NLEVELS, NLEAF, UNIQUERULE, FIRSTKEYCARD,
       FULLKEYCARD
from advise_index;
  
```

Optimizing Index Solutions

So, what are those proposed indexes worth?

- set current explain mode EVALUATE INDEXES
 - USE_INDEX = 'Y' for all Proposed Indexes
- \$ db2 -tvf 3Table_Heavy_Query.sql
- set current explain mode NO
- \$ db2 -tvf Query_In_Slide_Notes.sql
- 376 timerons
- Down from 81,524
 - 99.54% Reduced

PROPOSED_TOTAL_COST	IO_COST	CPU_COST	COMM_COST
376.8281	53.7631	36629316.0000	0.0000

36

Find the new query cost with the proposed indexes:

```
select dec(total_cost,20,4) as proposed_total_cost,
       dec(io_cost,20,4) as io_cost, dec(CPU_cost,20,4) as cpu_cost,
       dec(Comm_cost,20,4) as comm_cost
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions

Index Addition

- What is the value of each index individually, in isolation?
- Set USE_INDEX to 'N' for all Indexes
 - update advise_index set use_index='N';
- For each proposed index:
 - Set USE_INDEX to 'Y'
 - Update ADVISE_INDEX set USE_INDEX = 'Y' where NAME = 'IXNAME(N)'
 - set current explain mode EVALUATE INDEXES
 - db2 -tvf 3Table_Heavy_Query.sql
 - Retrieve the TOTAL_COST from EXPLAIN_OPERATOR table
 - db2 -tvf Query_In_Slide_Notes.sql
 - Compute Savings Percentage
 - Repeat!

37

```
select dec(total_cost,20,4) as add_IXNAME_total_cost
from Explain_Operator,
    (select max(explain_time) as maxtime
    from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Addition – 1st Index

```
$ db2 "set current explain mode NO"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='N'"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='Y' where name = 'IDX1503092345460'"
DB20000I The SQL command completed successfully.
$ db2 "set current explain mode EVALUATE INDEXES"
DB20000I The SQL command completed successfully.
$ db2 -tf 3Table_Heavy_Query.sql
SQL0217W The statement was not executed as only Explain information requests
are being processed.  SQLSTATE=01604
```

ORIGINAL_COST	ADD_IX1_TOTAL_COST	TIMERON_SAVINGS	VALUE_PCT
81524.1953	81524.1406	0.0547	0.0000670

38
38

Query to find cost savings:

```
select 81524.1953 as Original_Cost,
       dec(total_cost,20,4) as add_IX1_total_cost,
       81524.1953 - dec(total_cost,20,4) as timeron_savings,
       ((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 as value_pct
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Addition – 2nd Index

```
$ db2 "set current explain mode NO"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='N' where name = 'IDX1503092345460'"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='Y' where name = 'IDX1503092345530'"
DB20000I The SQL command completed successfully.
$ db2 "set current explain mode EVALUATE INDEXES"
DB20000I The SQL command completed successfully.
$ db2 -tf 3Table_Heavy_Query.sql
SQL0217W The statement was not executed as only Explain information requests
are being processed. SQLSTATE=01604
```

ORIGINAL_COST	ADD_IX2_TOTAL_COST	TIMERON_SAVINGS	VALUE_PCT
81524.1953	1844.3532	79679.8421	97.7376615

38
39

Query to find cost savings:

```
select 81524.1953 as Original_Cost,
       dec(total_cost,20,4) as add_IX2_total_cost,
       81524.1953 - dec(total_cost,20,4) as timeron_savings,
       ((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 as value_pct
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Addition – 3rd Index

```
$ db2 "set current explain mode NO"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='N' where name = 'IDX1503092345530'"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='Y' where name = 'IDX1503092346050'"
DB20000I The SQL command completed successfully.
$ db2 "set current explain mode EVALUATE INDEXES"
DB20000I The SQL command completed successfully.
$ db2 -tf 3Table_Heavy_Query.sql
SQL0217W The statement was not executed as only Explain information requests
are being processed. SQLSTATE=01604
```

ORIGINAL_COST	ADD_IX3_TOTAL_COST	TIMERON_SAVINGS	VALUE_PCT
81524.1953	81524.1953	0.0000	0E-7

40
40

Query to find cost savings:

```
select 81524.1953 as Original_Cost,
       dec(total_cost,20,4) as add_IX3_total_cost,
       81524.1953 - dec(total_cost,20,4) as timeron_savings,
       ((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 as value_pct
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Addition – 4th Index

```
$ db2 "set current explain mode NO"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='N' where name = 'IDX1503092346050'"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='Y' where name = 'IDX1503092346070'"
DB20000I The SQL command completed successfully.
$ db2 "set current explain mode EVALUATE INDEXES"
DB20000I The SQL command completed successfully.
$ db2 -tf 3Table_Heavy_Query.sql
SQL0217W The statement was not executed as only Explain information requests
are being processed. SQLSTATE=01604
```

ORIGINAL_COST	ADD_IX4_TOTAL_COST	TIMERON_SAVINGS	VALUE_PCT
81524.1953	6536.0610	74988.1343	91.9826733

41
41

Query to find cost savings:

```
select 81524.1953 as Original_Cost,
       dec(total_cost,20,4) as add_IX4_total_cost,
       81524.1953 - dec(total_cost,20,4) as timeron_savings,
       ((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 as value_pct
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Addition - Summary

Index Name	Timeron Savings	Value %
IDX1503092345460	0.0547	0.0000670
IDX1503092345530	79679.8421	97.7376615
IDX1503092346050	0.0000	0.0000000
IDX1503092346070	74988.1343	91.9826733
		189% 204018 %

**And the award for
LEAST valuable
index goes to...**

**And the award for
MOST valuable
index goes to...**

2nd Place MVI

By looking at proposed indexes in isolation, optimizer plans could change significantly, and some indexes may stand out as having very significant value. Wouldn't it be nice if the cost could be reduced by 189% ?!?!? We'd be getting back FREE resources from DB2 just by running the query! HA!

Optimizing Index Solutions

Index Subtraction

- If we take away just one index, how much value is lost?
 - Enables “cumulative effects” of indexes working together – TEAM!
- Set USE_INDEX to ‘Y’ for all Indexes
 - update advise_index set use_index='Y';
- Subtract each proposed index:
 - Set USE_INDEX to ‘N’
 - Update ADVISE_INDEX set USE_INDEX = ‘N’ where NAME = ‘IXNAME(N)’
 - set current explain mode EVALUATE INDEXES
 - db2 -tvf 3Table_Heavy_Query.sql
 - Retrieve the TOTAL_COST from EXPLAIN_OPERATOR table
 - db2 -tvf Query_In_Slide_Notes.sql
 - Compute Savings Percentage
 - Repeat!

43

```
select dec(total_cost,20,4) as subtract_IXNAME_total_cost
from Explain_Operator,
    (select max(explain_time) as maxtime
    from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Subtraction – 1st Index

```

$ db2 "set current explain mode NO"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='Y' " Use All
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index='N' where name = 'IDX1503092345460'"
DB20000I The SQL command completed successfully. -1
$ db2 "set current explain mode EVALUATE INDEXES"
DB20000I The SQL command completed successfully.
$ db2 -tf 3Table_Heavy_Query.sql
SQL0217W The statement was not executed as only Explain information requests
are being processed. SQLSTATE=01604
    
```

ORIGINAL_COST	SUBTRACT_IX1_TOTAL_COST	TIMERON_SAVINGS	REMAINING_VALUE_PCT	CONTRIBUTION_PCT
81524.1953	376.8834	81147.3119	99.5377036	0.4622964

44
44

Query to find cost savings:

```

select 81524.1953 as Original_Cost,
       dec(total_cost,20,4) as subtract_IX1_total_cost,
       81524.1953 - dec(total_cost,20,4) as timeron_savings,
       ((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 as
Remaining_value_pct,
       100.0 - (((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 ) as
Contribution_PCT
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
    
```

Optimizing Index Solutions Index Subtraction – 2nd Index

```
$ db2 "set current explain mode NO"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index = 'Y' where name = 'IDX1503092345460'"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index = 'N' where name = 'IDX1503092345530'"
DB20000I The SQL command completed successfully.
$ db2 "set current explain mode EVALUATE INDEXES"
DB20000I The SQL command completed successfully.
$ db2 -tf 3Table_Heavy_Query.sql
SQL0217W The statement was not executed as only Explain information requests
are being processed.  SQLSTATE=01604
```

ORIGINAL_COST	SUBTRACT_IX2_TOTAL_COST	TIMERON_SAVINGS	REMAINING_VALUE_PCT	CONTRIBUTION_PCT
81524.1953	6535.9990	74988.1963	91.9827494	8.0172506

45
45

```
select 81524.1953 as Original_Cost,
       dec(total_cost,20,4) as subtract_IX2_total_cost,
       81524.1953 - dec(total_cost,20,4) as timeron_savings,
       ((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 as
Remaining_value_pct,
       100.0 - (((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 ) as
Contribution_PCT
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Subtraction – 3rd Index

```
$ db2 "set current explain mode NO"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index = 'Y' where name = 'IDX1503092345530'"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index = 'N' where name = 'IDX1503092346050'"
DB20000I The SQL command completed successfully.
$ db2 "set current explain mode EVALUATE INDEXES"
DB20000I The SQL command completed successfully.
$ db2 -tf 3Table_Heavy_Query.sql
SQL0217W The statement was not executed as only Explain information requests
are being processed.  SQLSTATE=01604
```

ORIGINAL_COST	SUBTRACT_IX3_TOTAL_COST	TIMERON_SAVINGS	REMAINING_VALUE_PCT	CONTRIBUTION_PCT
81524.1953	376.8346	81147.3607	99.5377634	0.4622366

46
46

```
select 81524.1953 as Original_Cost,
       dec(total_cost,20,4) as subtract_IX3_total_cost,
       81524.1953 - dec(total_cost,20,4) as timeron_savings,
       ((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 as
Remaining_value_pct,
       100.0 - (((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 ) as
Contribution_PCT
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Subtraction – 4th Index

```
$ db2 "set current explain mode NO"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index = 'Y' where name = 'IDX1503092346050'"
DB20000I The SQL command completed successfully.
$ db2 "update advise_index set use_index = 'N' where name = 'IDX1503092346070'"
DB20000I The SQL command completed successfully.
$ db2 "set current explain mode EVALUATE INDEXES"
DB20000I The SQL command completed successfully.
$ db2 -tf 3Table_Heavy_Query.sql
SQL0217W The statement was not executed as only Explain information requests
are being processed. SQLSTATE=01604
```

ORIGINAL_COST	SUBTRACT_IX4_TOTAL_COST	TIMERON_SAVINGS	REMAINING_VALUE_PCT	CONTRIBUTION_PCT
81524.1953	1844.2912	79679.9041	97.7377376	2.2622624

47
47

```
select 81524.1953 as Original_Cost,
       dec(total_cost,20,4) as subtract_IX3_total_cost,
       81524.1953 - dec(total_cost,20,4) as timeron_savings,
       ((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 as
Remaining_value_pct,
       100.0 - (((81524.1953 - dec(total_cost,20,4)) * 100.0) / 81524.1953 ) as
Contribution_PCT
from Explain_Operator,
     (select max(explain_time) as maxtime
      from Explain_Operator) as b
where explain_time = b.maxtime
and operator_type = 'RETURN' with UR;
```

Optimizing Index Solutions Index Subtraction - Summary

Index Name	Timeron Savings	Remaining Value%	Contribution Value%
IDX1503092345460	81147.3119	99.5377036	0.4622964
IDX1503092345530	74988.1963	91.9827494	8.0172506
IDX1503092346050	81147.3607	99.5377634	0.4622366
IDX1503092346100	79679.9041	97.737737	2.2622624

And the award for
LEAST valuable
index goes to...

And the award for
MOST valuable
index goes to...

2nd Place MVI

We've now separated the "mice" from the "men". By addition and by subtraction, we know which indexes are the most valuable, providing the most benefit, and which are the least valuable. But wait, there's more...

Optimizing Index Solutions

Does a High Value Index have IX Access Only “Baggage”?

PROPOSED_INDEX	ON_TABLE	EXISTS	USE_INDEX	INDEX_COLS	NLEVELS	NLEAF	UNIQUE/RULE	FIRSTKEYCARD	FULLKEYCARD
IDX1503092345460	DBIPOC_HTML_STATUS_CODES	N	Y	+STATUS_DESC+STATUS_CODE	2	3	D	38	38
IDX1503092345530	DBIPOC_WEBSITE_DATA_TB	N	Y	+WEBSTATUS+BYTESXFERD	3	1891	D	10	189450
IDX1503092346050	DBIPOC_VERB_DESCRIPTIONS	N	Y	+ACTIONVERB+VERB_DESC	2	3	D	12	12
IDX1503092346070	DBIPOC_WEBSITE_DATA_TB	N	Y	+DOMAINNAME+TARGETFILE+BYTESXFERD+HITTIMESTAMP+PROTOCOL+ACTIONVERB+WEBSTATUS	3	896	D	134	134

RELOP_TYPE	HOW_APPLIED	PREDICATES
EQ	JOIN	(Q5.ACTIONVERB = Q7.ACTIONVERB)
LT	JOIN	(Q5.BYTESXFERD < (Q4.\$C0 / Q4.\$C1))
EQ	JOIN	(Q2.WEBSTATUS = Q1.STATUS_CODE)
EQ	START	(Q1.STATUS_DESC = 'OK. Request Fulfilled.')
EQ	STOP	(Q1.STATUS_DESC = 'OK. Request Fulfilled.')
EQ	START	(Q2.WEBSTATUS = Q1.STATUS_CODE)
EQ	STOP	(Q2.WEBSTATUS = Q1.STATUS_CODE)
LT	RESID	(Q5.BYTESXFERD < (Q4.\$C0 / Q4.\$C1))
EQ	JOIN	(Q5.WEBSTATUS = Q6.STATUS_CODE)
EQ	START	(Q6.STATUS_DESC = 'OK. Request Fulfilled.')
EQ	STOP	(Q6.STATUS_DESC = 'OK. Request Fulfilled.')
LT	SARG	(Q5.HITTIMESTAMP < '2011-12-31-21.35.43.304000000000')
EQ	SARG	(Q5.TARGETFILE = '/blog/rss/Scott_Hayes_rss2.xml')
EQ	SARG	(Q5.DOMAINNAME = 'webnj1.bbch.com')
EQ	START	(Q5.WEBSTATUS = Q6.STATUS_CODE)
EQ	STOP	(Q5.WEBSTATUS = Q6.STATUS_CODE)
EQ	START	(Q5.ACTIONVERB = Q7.ACTIONVERB)
EQ	STOP	(Q5.ACTIONVERB = Q7.ACTIONVERB)

Let's play Predicate BINGO!
\$ db2 -tvf Query_In_Notes.sql

VERB_DESC & PROTOCOL
are supporting IX Access Only

50
50

Query to look at EXPLAIN_PREDICATES table:

```
select a.relop_type, a.how_applied,
       varchar(a.predicate_text,100) as predicates
from explain_predicate a
where a.explain_time = (select max(b.explain_time) from explain_predicate
b);
```

VERB_DESC in Index IDX1503092346050 is giving IX Access Only but no predicate value

PROTOCOL in Index IDX1503092346070 is giving IX Access Only but no predicate value

Reserve IX Access Only for HEAVY SQL that is FREQUENTLY executed within important transactions.

Sometimes several extra columns might be added to an index to achieve IX Access Only – they could be, and probably should be, omitted, unless their inclusion also helps to increase the IX FULLKEYCARD (FULLKEYCARD

ideally > 5% of TBCARD)

Should a Row based table be converted to Column based? We've seen the guidelines and restrictions, but how about some automated analysis?

BLU DECISIONS

Feeling BLU? Have success stories or marketing propaganda peaked your interest? Do you have a Data Warehouse? Let's see if we can find some candidate tables for Column Organization...

Is BLU Organize By Column a good fit? Limitations & Exceptions Aside – Look For:

1. Tables that are heavily read with minimal write activity
 2. Not for operational tables where TXs retrieve only a few rows, usually via index – so we want TBSCANS
 3. No LOBs/LONGs or unsupported data types
 4. Queries generally reference very few columns
 5. Queries perform Grouping, Aggregation, & Summarization
- **Challenge: Can we smartly “automate” this analysis?**

I love a good challenge!

The image shows a screenshot of a SQL query in a database client. The query is designed to identify 'BLU Candidate Tables' based on several criteria. Callouts point to specific parts of the query:

- Queries Reference Few Columns:** Points to the `(mgt.NUM_COLUMNS_REFERENCED / (mgt.SECTION_EXEC_WITH_COL_REFERENCES + 0.01)) < 5` condition.
- Heavy Read Activity:** Points to the `mgt.rows_read > 1000000` condition.
- Want Table Scans:** Points to the `mgt.table_scans > 0` condition.
- with Minimal Write Activity:** Points to the `(mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted) < (mgt.rows_read / 1000)` condition.
- No Long or LOBs:** Points to the `mgt.LOB_OBJECT_L_PAGES is null and mgt.LONG_OBJECT_L_PAGES is null and mgt.XDA_OBJECT_L_PAGES is null` condition.

The query is as follows:

```
select varchar(mgt.tabschema,20) as tabschema, varchar(mgt.tabname,20) as
tabname,
mgt.member, mgt.tab_type, mgt.tab_organization, mgt.table_scans,
mgt.NUM_COLUMNS_REFERENCED, mgt.SECTION_EXEC_WITH_COL_REFERENCES,
mgt.rows_read,
(mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted) as rows_IUD,
(mgt.NUM_COLUMNS_REFERENCED /
mgt.SECTION_EXEC_WITH_COL_REFERENCES) as AVG_COLS_REFD
from table(mon_get_table('','-2)) as mgt
where mgt.tab_organization = 'R'
and mgt.tab_type = 'USER_TABLE'
and mgt.rows_read > 1000000
and (mgt.NUM_COLUMNS_REFERENCED /
(mgt.SECTION_EXEC_WITH_COL_REFERENCES + 0.01)) < 5
and mgt.table_scans > 0
-- AND condition below looks for IUD to be less than 0.1% of Rows Read, divide by 10000 for < 0.01%
and (mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted) < (mgt.rows_read / 1000)
and mgt.LOB_OBJECT_L_PAGES is null and mgt.LONG_OBJECT_L_PAGES is null and mgt.XDA_OBJECT_L_PAGES is null
order by rows_read desc;
```

```
select varchar(mgt.tabschema,20) as tabschema, varchar(mgt.tabname,20) as
tabname,
mgt.member, mgt.tab_type, mgt.tab_organization, mgt.table_scans,
mgt.NUM_COLUMNS_REFERENCED,
mgt.SECTION_EXEC_WITH_COL_REFERENCES,
mgt.rows_read,
(mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted) as rows_IUD,
(mgt.NUM_COLUMNS_REFERENCED /
mgt.SECTION_EXEC_WITH_COL_REFERENCES) as AVG_COLS_REFD
from table(mon_get_table("",-2)) as mgt
where mgt.tab_organization = 'R'
and mgt.tab_type = 'USER_TABLE'
and mgt.rows_read > 1000000
and (mgt.NUM_COLUMNS_REFERENCED /
(mgt.SECTION_EXEC_WITH_COL_REFERENCES + 0.01)) < 5
and mgt.table_scans > 0
-- AND condition below looks for IUD to be less than 0.1% of Rows Read,
divide by 10000 for < 0.01%
```

and (mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted) < (
mgt.rows_read / 1000)

and mgt.LOB_OBJECT_L_PAGES is null and mgt.LONG_OBJECT_L_PAGES is
null and mgt.XDA_OBJECT_L_PAGES is null

order by rows_read desc;

BLU Candidate Tables

5th Criteria: Queries perform Grouping, Aggregation, & Summarization

- Capture Query Workload (several hours or days)
- FILTER DB Workload to find GROUP BY & SQL w/ SORTS
 - Or Find SQL Impacting I/O to BLU_DECISION.SQL tables
 - Look for GROUP BY, SORTS, High Avg Rows Read
- Export Workload(s) to file(s) suitable for input to db2batch
- Use db2batch to Explain (only) the SQL in Workload(s)
- IF tables in EXPLAIN_OBJECT match tables found from previous BLU_DECISION.SQL query (intersection/inner join), THEN you've got some STRONG candidates for COLUMN ORGANIZED tables!

Where can we find a relationship between tables and SQL workloads? EXPLAIN! In particular, EXPLAIN_OBJECT table will tell you about tables, indexes, and other objects used to run a SQL query. This slide describes a methodology for discovering tables that are “victims” of queries that perform grouping, aggregation, and summarization. As always, individual results may vary.



IDUG
Leading the DB2 User
Community since 1988

IDUG DB2 EMEA Tech Conference
Dublin, Ireland | November 2015

 #IDUGDB2

BLU CANDIDATE TABLES Capture Query Workload

Brother-Panther@ - db2in105@LPAR2160018/DBIREPOS

File Edit View Tools Reports Window Help

1 week

Statement Workload from 3/1/15 3:30 PM to 3/6/15 6:30 PM

Last Refresh: 3/5/15 6:36 PM
Rows: 190

Follow Up	Stmt ID	Verb	Type	# Execs	Avg IX L Reads	IX Read Efficiency	Avg Rows Read	% Rows Read	Avg Rows Fetched	% CPU Time	Avg Exec Time (sec)	Avg Sort Time (ms)	% Sort Time	Avg CPU Time (sec)	CPU Time (%)
	00D62CEEA...	SELECT	DYNAMIC	2	0.000	90,551.429	633,860.000	0.118%	7.000	0.863%	8.253857	974.500	62.993%	2.008608	4.011
	CB778B621...	SELECT	DYNAMIC	2	0.000	23.123	633,860.000	0.118%	27,413.000	0.267%	6.188746	450.500	29.121%	0.622175	1.244
	78F000609...	SELECT	DYNAMIC	2	1,857.500	0.000	0.000	0.000%	22,561.000	0.112%	2.499976	57.500	3.717%	0.260755	0.521
	D91299236...	SELECT	DYNAMIC	293	3.000	1.000	5.000	0.000%	5.000	0.005%	0.001263	0.311	2.941%	0.000076	0.022
	614AE802B...	SELECT	DYNAMIC	400	0.000	539,455.319	633,860.000	23.611%	1.175	17.353%	3.003478	0.043	0.549%	0.202048	80.815
	016D26666...	SELECT	DYNAMIC	1	0.000	0.000	0.000	0.000%	3.000	0.000%	0.008280	7.000	0.226%	0.001038	0.001
	8AE3F658A...	SELECT	DYNAMIC	3	0.000	0.000	0.000	0.000%	1.667	0.001%	0.003917	2.333	0.226%	0.001123	0.003
	C4B307901...	SELECT	DYNAMIC	1	0.000	0.000	0.000	0.000%	2.000	0.000%	0.008252	7.000	0.226%	0.001007	0.001
	011C7DA2B...	UPDATE	DYNAMIC	2	2.500	4.500	4.500	0.000%	0.000	0.001%	0.015747	0.000	0.000%	0.002909	0.005
	053DF6966...	SELECT	DYNAMIC	8	2.500	2.417	3.625	0.000%	1.500	0.000%	0.000369	0.000	0.000%	0.000091	0.000
	061D8ECB3...	SELECT	DYNAMIC	1	1.000	1.000	451.000	0.000%	451.000	0.000%	0.047767	0.000	0.000%	0.001809	0.001
	07BE75FBD...	SELECT	DYNAMIC	1	0.000	1.001	1,064.000	0.000%	1,063.000	0.006%	3.422229	0.000	0.000%	0.028712	0.028
	07E42B209...	UPDATE	DYNAMIC	4	12.500	9.750	9.750	0.000%	0.000	0.005%	0.056797	0.000	0.000%	0.005741	0.022
	0A87C8D20...	SELECT	DYNAMIC	2	0.000	57,627.091	633,898.000	0.118%	11.000	0.144%	1.335034	0.000	0.000%	0.335567	0.671

There are commercially available tools that can help you capture and manage SQL workloads, or with greater difficulty, you could capture SQL from db2pd, db2top, or queries to MON_GET_PACKAGE_CACHE. For purposes of demonstrating the METHOD of this process, DBI's Brother-Panther is illustrated.


IDUG
 Leading the DB2 User Community since 1988

IDUG DB2 EMEA Tech Conference
 Dublin, Ireland | November 2015

 #IDUGDB2

BLU CANDIDATE TABLES

FILTER the Database Workload

Brother-Panther® - db2in105@LPAR21.60018/DBIPEPOS

File Edit View Tools Reports Window Help

Statement Performance for LPAR21.60018/DBIPEPOS

Statement Workload from 3/1/15 3:30 PM to 3/6/15 6:30 PM

Last Refresh: 3/5/15 6:36 PM
 Rows: 5

Follow Up	Stmt ID	Verb	Type	# Execs	Avg IX L Reads	IX Read Efficiency	Avg Rows Read	% Rows Read	Avg Rows Fetched	% CPU Time	Avg Exec Time (sec)	Avg Sort Time (ms)	% Sort Time	Avg CPU Time (sec)	CPU Time (sec)
	00D62CEEA...	SELECT	DYNAMIC	2	0.000	90,551.429	633,860.000	0.118%	7,000	0.863%	8.253857	974,500	62.993%	2.008608	4.01721
	CB778B621...	SELECT	DYNAMIC	2	0.000	23.123	633,860.000	0.118%	27,413.000	0.267%	6.188746	450,500	29.121%	0.622175	1.24434
	78F000609...	SELECT	DYNAMIC	2	1,857,500	0.000	0.000	0.000%	22,561.000	0.112%	2.499976	57,500	3.717%	0.260755	0.52151
	4898039D3...	SELECT	DYNAMIC	2	0.000	1.000	1.000	0.000%	1.000	0.000%	0.003431	0.000	0.000%	0.000087	0.00017
	B2BF8B24A...	SELECT	DYNAMIC	2	0.000	52,821.667	633,860.000	0.118%	12,000	0.202%	2.830543	0.000	0.000%	0.471327	0.94265

Workload Display Filter

Check all statement types to show. These settings are only applied once.

SELECT INSERT UPDATE DELETE DDL Other

Include SQL in Filter

Optionally include a SQL text fragment in the filter.

[GROUP BY]

OK Cancel

DBI's Brother-Panther allows you to filter workloads by types of SQL or SQL containing certain strings. Alternatively, in a command line interface, you might pass the SQL workload through grep or equivalent OS command.

BLU CANDIDATE TABLES

EXPORT filtered Database Workload to a File

The screenshot shows the IBM DB2 Performance Center interface on the left and a Notepad2 window on the right. The Performance Center window displays a table of statement performance metrics for database DBIPOCDB. The table has columns for Follow Up, Stmt ID, Verb, Type, and # Execs. The Notepad2 window shows the exported SQL workload, which includes performance comments and SQL statements. A red handwritten note '+3 more' with a downward arrow points to the end of the SQL statements in the Notepad2 window.

Follow Up	Stmt ID	Verb	Type	# Execs
	00D62CEEA...	SELECT	DYNAMIC	2
	CB778B621...	SELECT	DYNAMIC	2
	7BF000609...	SELECT	DYNAMIC	2
	4898039D3...	SELECT	DYNAMIC	2
	B2BF8B24A...	SELECT	DYNAMIC	2

```

1  --#COMMENT SQL workload for Database DBIPOCDB from 3/1/15 3:30 PM to 3/6/15
2  --#COMMENT Statement ID: 00D62CEEA067609E9251E3B27917F5BA3E197E0
3  --#COMMENT CPU Percent: 0.863%
4  --#COMMENT Rows Read Percent: 0.118%
5  --#COMMENT Avg Rows Read: 633860.0
6  --#COMMENT Avg IX Logical Reads: 0.0
7  --#COMMENT Avg Exec Time: 8.253857
8  --#COMMENT % Exec Time: 0.338
9  --#COMMENT SET FREQUENCY 2
10 SELECT DAYOFWEEK(HITTIMESTAMP), COUNT(*)
11     FROM DBIPOC.WEBSITE_DATA_TB
12 GROUP BY DAYOFWEEK(HITTIMESTAMP)
13 ORDER BY 1
14 ;
15 --#COMMENT Statement ID: CB778B621E9F08A592EFCF06275F8667CAE6C468
16 --#COMMENT CPU Percent: 0.267%
17 --#COMMENT Rows Read Percent: 0.118%
18 --#COMMENT Avg Rows Read: 633860.0
19 --#COMMENT Avg IX Logical Reads: 0.0
20 --#COMMENT Avg Exec Time: 6.188746
21 --#COMMENT % Exec Time: 0.254
22 --#COMMENT SET FREQUENCY 2
23 SELECT DOMAINNAME, COUNT(*)
24     FROM DBIPOC.WEBSITE_DATA_TB
25 GROUP BY DOMAINNAME
26 ORDER BY 2 DESC
27 ;
  
```

+3 more
↓

The filtered workload (all statements containing GROUP BY) is exported to a flat text file that is suitable for input to db2batch or db2advis. You should notice that there are also liberal comments that document the performance attributes of the exported statements.

BLU CANDIDATE TABLES

Use db2batch to Explain the Database Workload

- db2 “delete from db2in105.explain_instance”
- db2batch -d DBNAME -f groupbysql.txt -r rpt.txt -o e explain

```
* Total Entries:          5
* Total Time:            0.000000 seconds
* Minimum Time:         0.000000 seconds
* Maximum Time:         0.000000 seconds
* Arithmetic Mean Time: 0.000000 seconds
* Geometric Mean Time:  0.000000 seconds
-----
* Timestamp: Thu Mar 05 2015 19:14:09 CST
```

58

Begin by clearing out the Explain tables. Delete to Explain_Instance does a cascading delete to other Explain tables.

The “-o e explain” option instructs db2batch to Explain the statements but not run them!

BLU CANDIDATE TABLES

The Grand Finale – Drum Roll Please!



This is exciting!

IDUG
Leading the DB2 User
Community since 1988

IDUG DB2 EMEA Tech Conference
Dublin, Ireland | November 2015

#IDUGDB2

BLU Candidate Tables
Automating the Creation

Queries Reference Few Columns

Heavy Read Activity

with Minimal Write Activity

Want Table Scans

No Long or LOBs

Queries performing Sorts/Aggregation

TABLE_SCHEMA	TABLE_NAME	ROWS_READ	ROWS_IUD	AVG_COLS_REFD
DBIPOC	WEBSITE_DATA_TB	1496545572	0	3
DBIPOC	WEBSITE_DATA_TB	1496545572	0	3
DBIPOC	WEBSITE_DATA_TB	1496545572	0	3

```

select varchar(mgt.tabschema,20) as table_schema, varchar(mgt.tabname,20) as table_name,
mgt.rows_read, (mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted) as rows_IUD,
(mgt.NUM_COLUMNS_REFERENCED / mgt.SECTION_EXEC_WITH_COL_REFERENCES) as AVG_COLS_REFD
from table(mon_get_table('','',-2)) as mgt,
explain_object as obj
where obj.object_type = 'TA'
and obj.object_schema = mgt.tabschema
and obj.object_name = mgt.tabname
and mgt.tab_organization = 'R'
and mgt.tab_type = 'USER_TABLE'
and mgt.rows_read > 1000
and (mgt.NUM_COLUMNS_REFERENCED / (mgt.SECTION_EXEC_WITH_COL_REFERENCES + 0.01)) < 5
and mgt.table_scans > 0
-- AND condition below looks for IUD to be less than 0.1% of Rows Read,
and (mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted) < ( mgt.rows_read / 1000 )
and mgt.LOB_OBJECT_L_PAGES is null and mgt.LONG_OBJECT_L_PAGES is null and mgt.XDA_OBJECT_L_PAGES is null
order by varchar(mgt.tabschema,20) asc, varchar(mgt.tabname,20) asc;

```

```

select varchar(mgt.tabschema,20) as table_schema, varchar(mgt.tabname,20)
as table_name,
mgt.rows_read, (mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted)
as rows_IUD,
(mgt.NUM_COLUMNS_REFERENCED /
mgt.SECTION_EXEC_WITH_COL_REFERENCES) as AVG_COLS_REFD
from table(mon_get_table('','',-2)) as mgt,
explain_object as obj
where obj.object_type = 'TA'
and obj.object_schema = mgt.tabschema
and obj.object_name = mgt.tabname
and mgt.tab_organization = 'R'
and mgt.tab_type = 'USER_TABLE'
and mgt.rows_read > 1000
and (mgt.NUM_COLUMNS_REFERENCED /
(mgt.SECTION_EXEC_WITH_COL_REFERENCES + 0.01)) < 5
and mgt.table_scans > 0
-- AND condition below looks for IUD to be less than 0.1% of Rows Read,

```

divide by 10000 for < 0.01%

and (mgt.rows_inserted + mgt.rows_updated + mgt.rows_deleted) < (
mgt.rows_read / 1000)

and mgt.LOB_OBJECT_L_PAGES is null and mgt.LONG_OBJECT_L_PAGES is
null and mgt.XDA_OBJECT_L_PAGES is null

order by varchar(mgt.tabschema,20) asc, varchar(mgt.tabname,20) asc;

What have we learned?

- **For maximum beneficial performance impact, focus on weights in the DB2 workload**
- Put heavy weights on a diet to lighten the load
 - Beware of the weight of BAD Indexes – DROP them, or redefine them to make them less BAD
- Diet plan includes:
 - Optimal Index Design
 - EVALUATE INDEXES with ADDITION and SUBTRACTION
 - Use of BLU Column Organized Tables where appropriate

Learn more from this IDUG blog: <http://www.idug.org/p/bl/ar/blogaid=351>

Learn more from the DB2 LUW Performance blogs:
<http://www.dbisoftware.com/blog/>



IDUG DB2 EMEA Tech Conference
Dublin, Ireland | November 2015

#IDUGDB2

Scott Richard Hayes

DBI Software, @dbisoftware

sales@dbisoftware.com

Twitter: @srhayes



[D10] More Sage Advice: Invaluable
DB2 LUW Performance Insights from
Around the Globe

*Please fill out your session
evaluation before leaving!*



The end! Follow me on twitter at @srhayes! Follow DBI at @dbisoftware