DB2 Monitoring Essentials

Philip K. Gunning
Gunning Technology Solutions, LLC

Session Code: C2
Date and Time of Presentation: May 2, 2013, 1:30 – 2:30 pm | Platform: DB2 for LUW

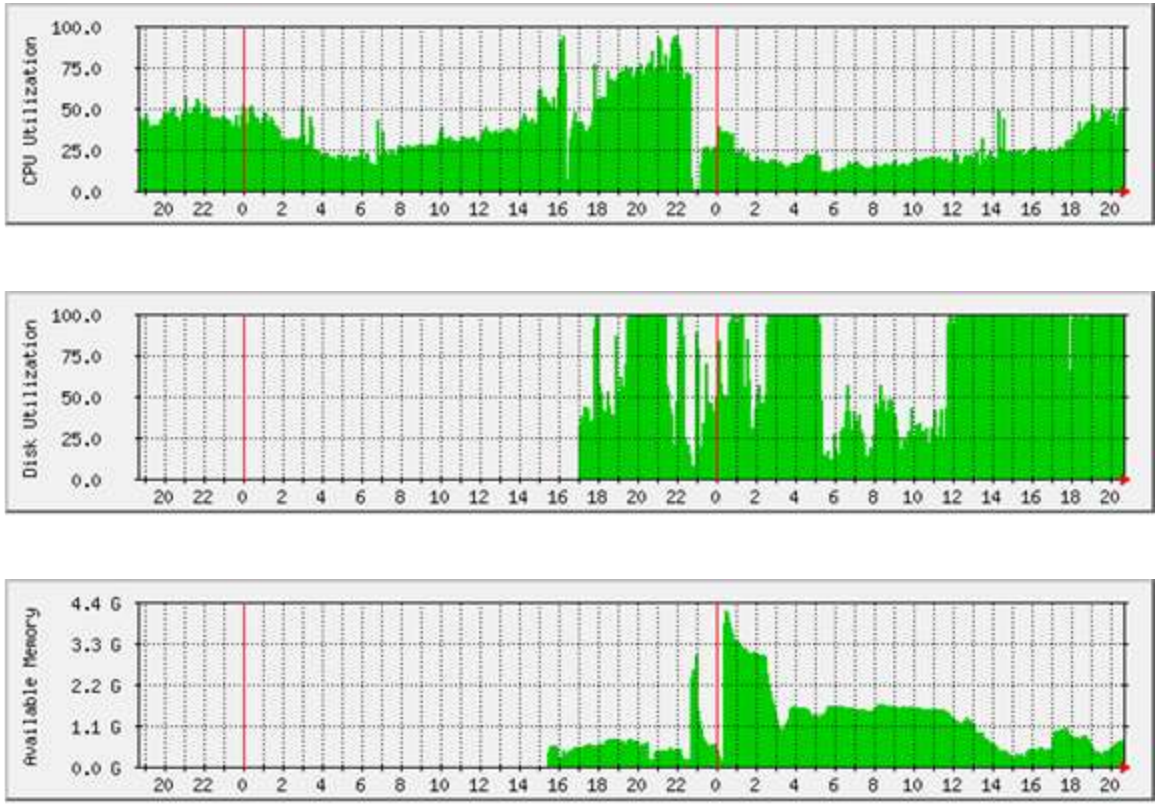
Overview

- Where is the problem? DB2 or OS?
- Isolate the problem
  - Where's the bottleneck?
  - CPU
  - IO
  - Memory
- Check key metrics and parameters
- Highlight key snapshots, table functions, db2pd output, and new MONREPORT reporting module

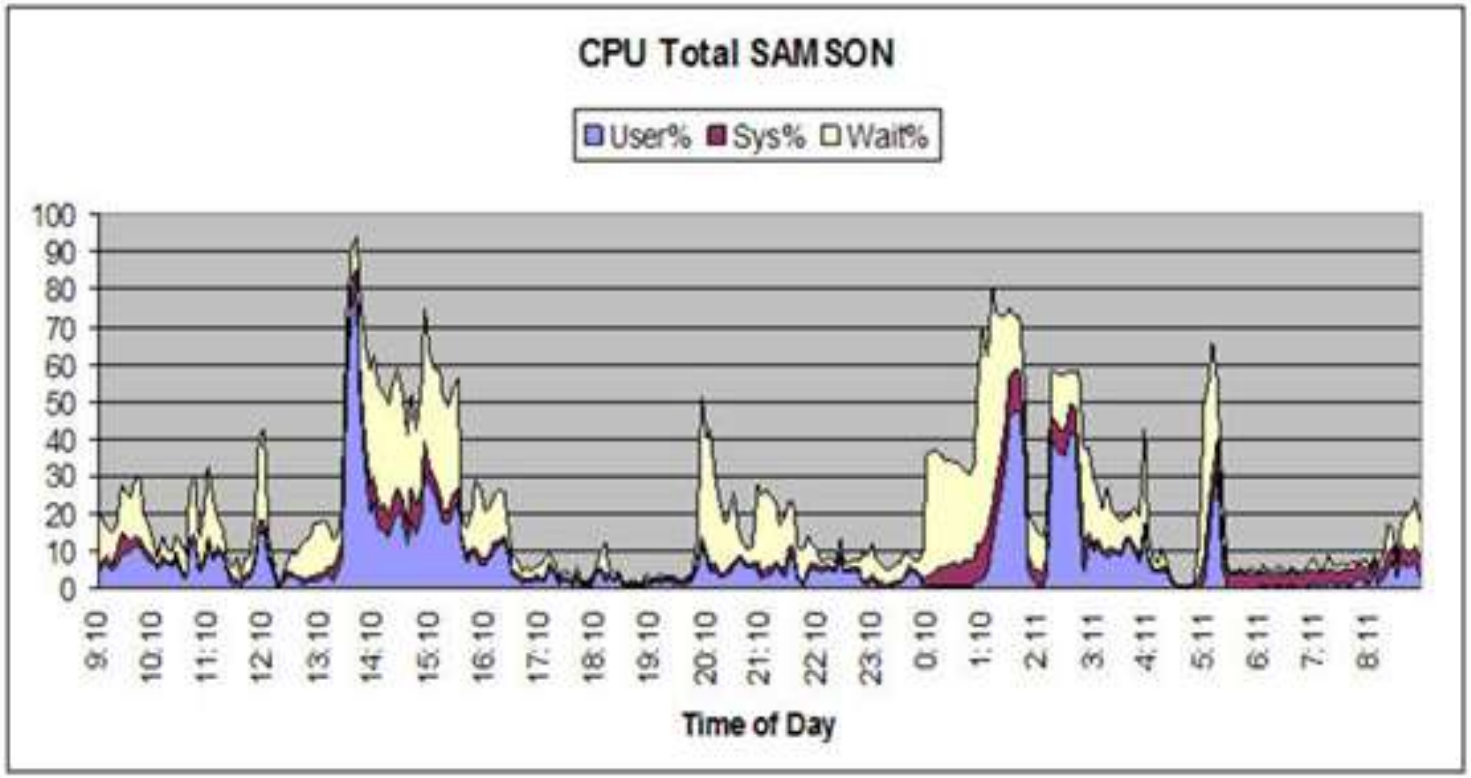
Isolate the Problem with a Quick 5-minute Fire Fighting Drill

- First step – Check the Graphs
- Next, quickly take an application snapshot and database snapshot for later analysis
  - This will capture state of database and all applications executing
  - If it is a DB2 problem it will be associated with an EXECUTING application
- Then immediately review last entry in db2diag.log
  - VI, cat or tail, db2diag command or notepad
- Quickly Review OS related metrics
  - TOP, TOPAS, NMON, Windows Task Manager
  - Review CPU usage of db2sysc process
- Identify top process or application
  - Is it DB2?

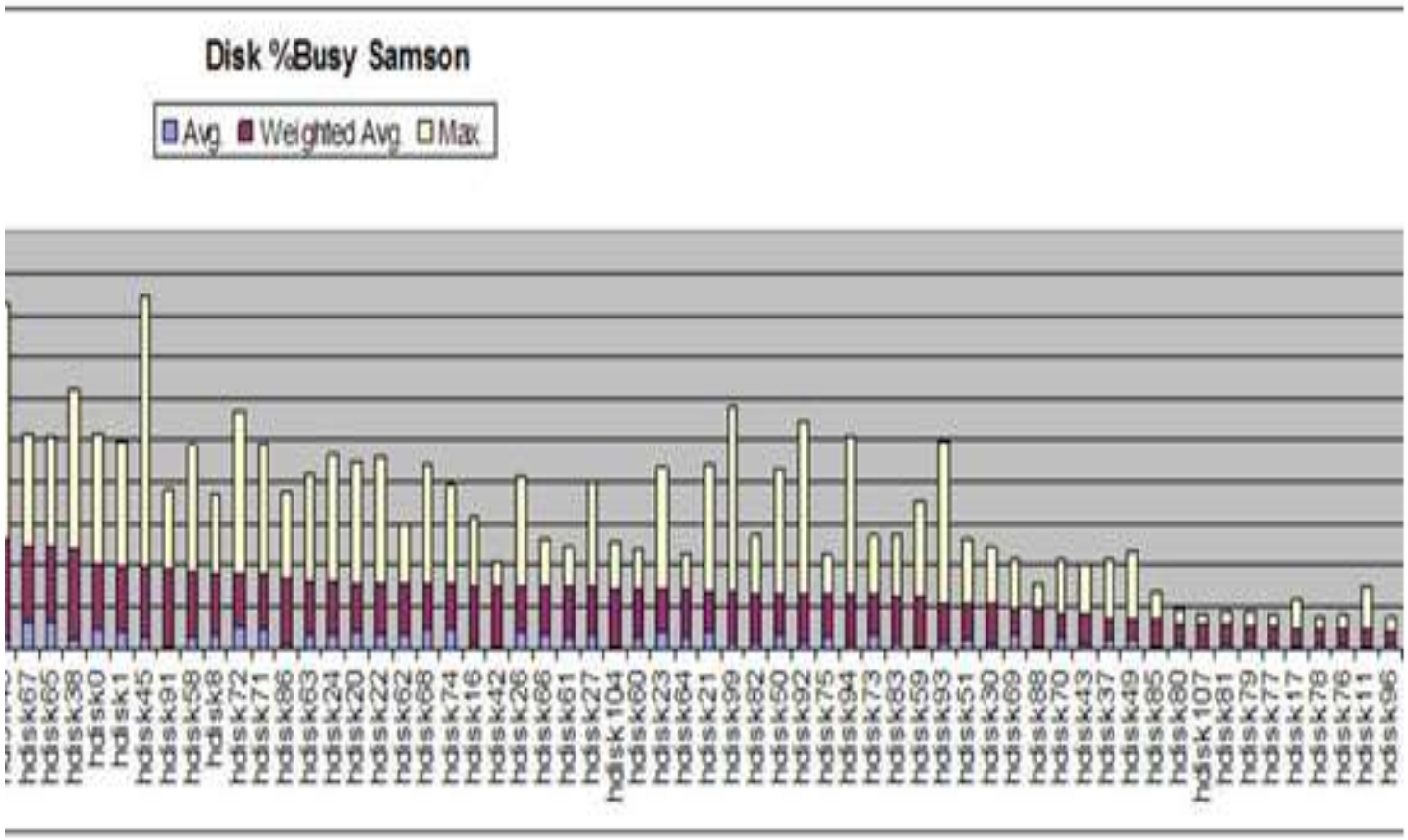
Quick Check of Key OS Resources



NMON Example - AIX



NMON Example, CONT.



Quick Check of Key DB2 Potential Problem Areas

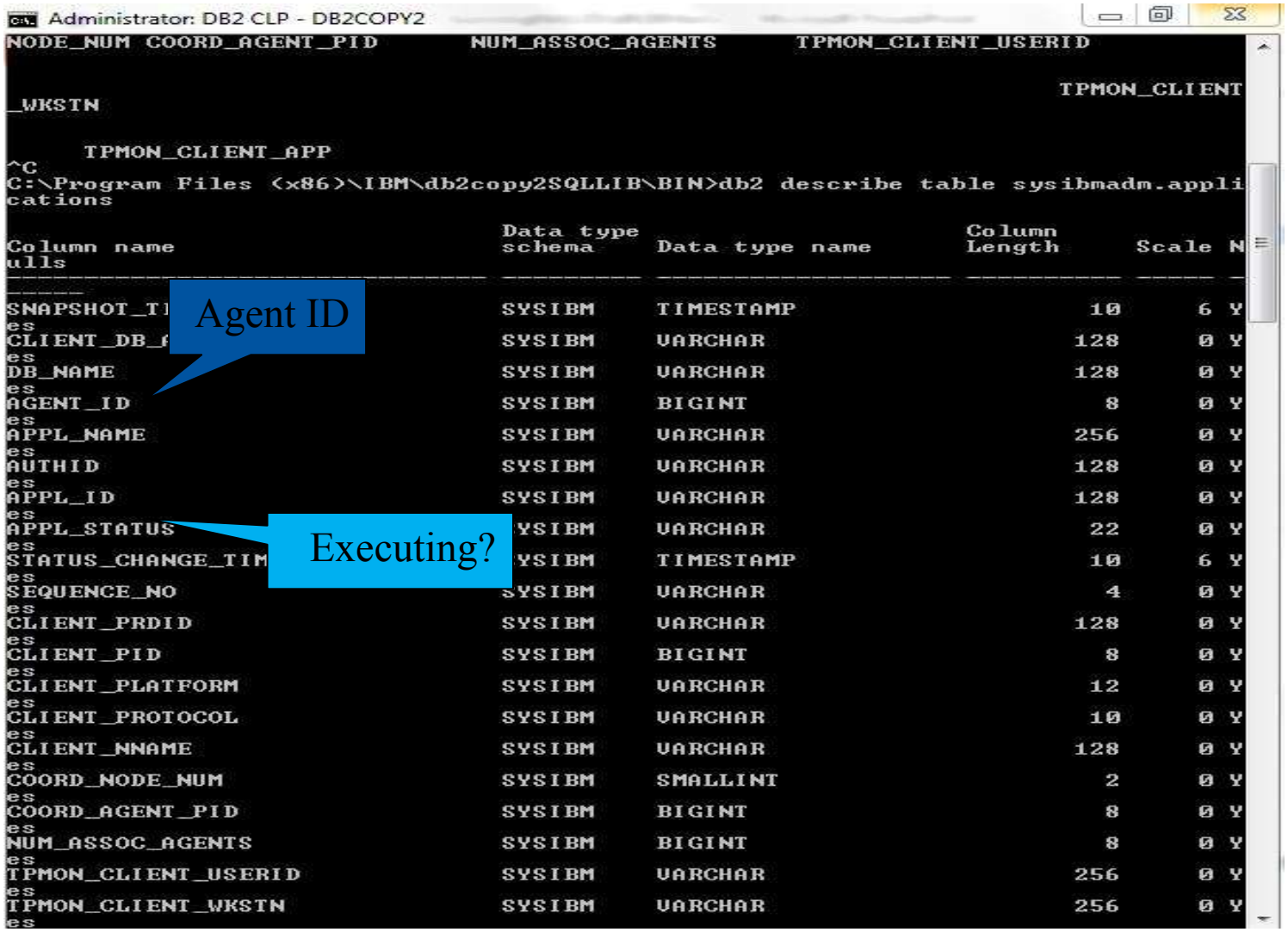
- What will cause DB2 to hang or stop processing
  - Archive Log filesystem full or problems with archive logging?
    - Check the db2diag log for archive log failure
      - DF command on UNIX or Linux
      - Windows – Disk full?
- Suboptimal query or queries doing scans in memory
  - High number of logical index or table reads
- SAN or Disk subsystem problems
  - Controller issues, disks become unmapped, unmounted
- Network Problem
  - Ping the DB2 server and save timings
  - Graph network performance

db2pd -d <dbname> -applications


```
0x0780000001749620 24051    [000-24051] 1        58700       UOW-Waiting         0           0           0           0           *LOCAL.fnprdi.120329234143
                      1             234384
0x0780000001150080 347      [000-00347] 1        5318        UOW-Waiting         0           0           239         220         10.254.71.175.34291.12032304153
                      1             7
0x07800000014A4EA0 26393    [000-26393] 1        52199       UOW-Executing       114         20210       168         14112       *LOCAL.fnprdi.120330002044
                      1             235885
0x0780000000F88080 393      [000-00393] 1        13541       UOW-Waiting         0           0           223         247         10.254.71.175.39155.12032304153
                      1             8
0x07800000011106C0 360      [000-00360] 1        14055       UOW-Waiting         0           0           239         220         10.254.71.175.35315.12032304153
                      1             26
0x0780000000F83A40 406      [000-00406] 1        3519        UOW-Waiting         0           0           239         220         10.254.71.175.40691.12032304153
                      1             5
0x07800000015F0080 24827    [000-24827] 1        49861       UOW-Waiting         0           0           20          224         10.254.71.176.25661.12032817444
                      1             234384
0x078000000141C800 498      [000-00498] 1        45235       UOW-Waiting         0           0           63          15477       *LOCAL.fnprdi.120323051416
                      1             110
0x0780000000F35320 25794    [000-25794] 1        11742       UOW-Executing       221         10409       221         10409       10.254.71.178.25779.12032817565
                      1             235516
0x0780000001080080 419      [000-00419] 1        22876       UOW-Waiting         0           0           73          43          10.254.71.175.43507.12032304153
                      1             74
0x0780000001746440 25715    [000-25715] 1        14826       UOW-Waiting         0           0           20          224         10.254.71.176.16023.12032817555
                      1             235457
0x07800000015FC800 23340    [000-23340] 1        52970       UOW-Waiting         0           0           168         13882       10.254.18.142.22803.12032817250
                      1             233924
0x0780000000EC0780 20965    [000-20965] 1        47034       UOW-Waiting         0           0           20          224         10.254.71.175.22840.12032311030
                      1             13033
0x0780000001117800 25774    [000-25774] 1        57879       UOW-Executing       228         20141       242         15615       *LOCAL.fnprdi.120330001028
                      1             235495
0x0780000000EE3260 399      [000-00399] 1        20223       UOW-Waiting         0           0           160         1           10.254.71.178.22759.12032304153
                      1             48
0x07800000011B1500 366      [000-00366] 1        4547        UOW-Waiting         0           0           233         7           10.254.71.175.36339.12032304153
                      1             27
0x0780000000F5B120 58991    [000-58991] 1        53741       UOW-Waiting         0           0           233         7           10.254.71.175.22709.12032613051
                      1             88038
0x0780000001520080 20991    [000-20991] 1        48062       UOW-Waiting         0           0           20          224         10.254.71.175.23608.12032311032
                      1             13050
0x07800000015B3260 34208    [000-34208] 1        43436       UOW-Waiting         0           0           253         1           10.254.71.175.42430.12032804151
                      1             209967
0x07800000014F3260 8004     [000-08004] 1        44721       UOW-Waiting         0           0           95          13051       10.254.14.254.58372.12032814264
                      1             225316
0x0780000000F31460 392      [000-00392] 1        20224       UOW-Waiting         0           0           223         247         10.254.71.175.38899.12032304153
                      1             65
0x078000000138C020 20938    [000-20938] 1        25189       UOW-Waiting         0           0           20          224         10.254.71.175.33710.12032311014
Standard input
```

Tying db2pd –applications to Application Snapshot

Application Information via Application SQL Administrative View in DB2 10



SQL Snapshot Table Functions

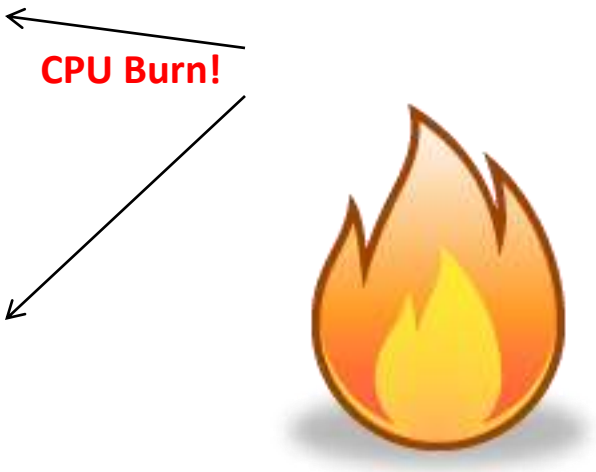
```
#!/bin/ksh
db2 connect to dsdm;
db2 "SELECT INTEGER(applsnap.agent_id) AS agent_id,
CAST(LEFT(applinfo.appl_name,10) AS CHAR(10)) AS appl_name,
CAST(left(client_nname,35) AS CHAR(35)) AS nname,
 INTEGER(locks_held) AS locks,applsnap.rows_read as rr, applsnap.rows_written as rw,applsnap.total_sorts as sorts,
applsnap.sort_overflows as oflows, applsnap.lock_timeouts as touts, applsnap.total_hash_loops as loops, applsnap.agent_usr_cpu_time_s as usersecs,
applsnap.agent_sys_cpu_time_s as syscpu, applsnap.locks_waiting as lkwait,
 SUBSTR(APPL_STATUS,1,10) AS APPL_STATUS, SUBSTR(stmt_snap.STMT_TEXT, 1, 999) AS STMT_TEXT
FROM TABLE( sysproc.snap_get_appl('',-1)) AS applsnap,
TABLE( sysproc.snap_get_appl_info('',-1)) as applinfo,
TABLE (sysproc.snap_get_stmt('',-1)) as stmt_snap
 WHERE applinfo.agent_id = applsnap.agent_id
and applinfo.agent_id = stmt_snap.agent_id
and appl_status in ('UOWEXEC','LOCKWAIT')
 ORDER BY appl_status";
db2 connect reset;
```

Steps Taken

- Step 1 – Determine if problem in DB2, if not, EXIT!
- Step 2 – If in DB2, take database manager and database snapshot, application snapshot, (maybe lock snapshot) and use db2diag command or tail db2diag log
- Step 3 – If db2diag.log does not contain errors then proceed to quick review of Instance and DB snapshots to see if thresholds breached
- Step 4 – review applications in ***Executing state*** *and determine which application is causing problem*
  - db2pd, application snapshot, SQL Administrative View, snapshot table functions, MONREPORT reporting module, db2top or other monitor

Essential Application Elements to Examine

- Look at applications in Executing and Lock-Wait status, one of these will be the cause of the problem
- For applications in Executing status, look for the following:
  - ✓ Total sorts = 35782
  - ✓ Total sort time (ms) = 7097
  - ✓ Total sort overflows = 218
  - ✓ Buffer pool data logical reads = 1102578477
  - ✓ Buffer pool data physical reads = 87171
  - ✓ Buffer pool temporary data logical reads = 55264
  - ✓ Buffer pool temporary data physical reads = 0
  - ✓ Buffer pool data writes = 579
  - ✓ Buffer pool index logical reads = 325915793
  - ✓ Buffer pool index physical reads = 124802
  - ✓ Buffer pool temporary index logical reads = 0
  - ✓ Buffer pool temporary index physical reads = 0



Essential Application Elements to Examine, cont.

- For applications in Executing status, look for the following:
  - ✓ Rows deleted = 57991
  - ✓ Rows inserted = 350298
  - ✓ Rows updated = 1185248
  - ✓ Rows selected = 366993
  - ✓ Rows read = 1106728657
  - ✓ Rows written = 5009851

✓ This application had to read 1 Billion rows to select 366,000! Indication of suboptimal SQL!

Essential Application Elements to Examine, cont.

- ✓ Total User CPU Time used by agent (s) = 8402.923420
- ✓ Total System CPU Time used by agent (s) = 35.695327
- ✓ Host execution elapsed time = 8979.020210
- ✓ Number of hash joins = 258
- ✓ Number of hash loops = 0
- ✓ Number of hash join overflows = 0
- ✓ Number of small hash join overflows = 0

Essential Application Elements to Examine, cont.

- ✓ Statement start timestamp = 03/23/2012 18:38:30.903272
- ✓ Statement stop timestamp =
- ✓ Elapsed time of last completed stmt(sec.ms)= 0.000145
- ✓ Total Statement user CPU time = 0.008349
- ✓ Total Statement system CPU time = 0.000088
- ✓ SQL compiler cost estimate in timerons = **16658**

Essential Application Elements to Examine, cont.

- ✓ Dynamic SQL statement text:

```
SELECT SUM(MONETARY_AMOUNT) , SUM(STATISTIC_AMOUNT) , SUM(MONETARY_AMOUNT) ,
SUM(STATISTIC_AMOUNT) FROM PS_BP_ACT_TAO13 WHERE KK_TRAN_ID = '0003867472' AND
KK_TRAN_DT =
'2012-03-15' AND BUSINESS_UNIT= 'SDPBC' AND LEDGER_GROUP= 'DETAIL' AND ACCOUNT=
'516000' AND DEPTID= '1991' AND BASE_CURRENCY ='USD' AND STATISTICS_CODE =' ' AND
BALANCING_LINE
 = 'N' AND KK_SKIP_EDITS <> 'Y' AND LIQ_FLG = 'N' AND AFFECT_SPEND_OPTN <> 'N' AND
OPERATING_UNIT = 'BD01' AND PRODUCT = '000' AND FUND_CODE = '1000' AND CLASS_FLD =
'7902' AND PROGRAM_CODE = '0000' AND BUDGET_REF = ' ' AND AFFILIATE = ' ' AND
AFFILIATE_INTRA1 = ' ' AND AFFILIATE_INTRA2 = ' ' AND CHARTFIELD1 = ' ' AND CHARTFIELD2 = ' '
AND CHARTFIELD3 = ' ' AND BUSINESS_UNIT_PC = ' ' AND PROJECT_ID = ' ' AND ACTIVITY_ID = ' '
AND RESOURCE_TYPE = ' ' AND BUDGET_PERIOD = '2012' AND PROCESS_INSTANCE = 6227207
```

db2exfmt explain tool

```
Connecting to the Database.
******************** EXPLAIN INSTANCE ********************
Original Statement:
------------------
UPDATE PS_BP_PST1_TAO13 SET KK_PROC_INSTANCE = 6211340+ 1000000000
WHERE PROCESS_INSTANCE=? AND NOT EXISTS (
  SELECT 'X'
  FROM PS_LEDGER_KK
  WHERE PS_LEDGER_KK.BUSINESS_UNIT = PS_BP_PST1_TAO13.BUSINESS_UNIT
AND
        PS_LEDGER_KK.LEDGER = PS_BP_PST1_TAO13.LEDGER AND
        PS_LEDGER_KK.ACCOUNT = PS_BP_PST1_TAO13.ACCOUNT AND
        PS_LEDGER_KK.DEPTID = PS_BP_PST1_TAO13.DEPTID AND
        PS_LEDGER_KK.OPERATING_UNIT = PS_BP_PST1_TAO13.OPERATING_UNIT
AND
        PS_LEDGER_KK.PRODUCT = PS_BP_PST1_TAO13.PRODUCT AND
        PS_LEDGER_KK.FUND_CODE = PS_BP_PST1_TAO13.FUND_CODE AND
        PS_LEDGER_KK.CLASS_FLD = PS_BP_PST1_TAO13.CLASS_FLD AND
        PS_LEDGER_KK.PROGRAM_CODE = PS_BP_PST1_TAO13.PROGRAM_CODE AND
        PS_LEDGER_KK.BUDGET_REF = PS_BP_PST1_TAO13.BUDGET_REF AND
        PS_LEDGER_KK.AFFILIATE = PS_BP_PST1_TAO13.AFFILIATE AND
        PS_LEDGER_KK.AFFILIATE_INTRA1 =


PS_BP_PST1_TAO13.AFFILIATE_INTRA1
```

```
Access Plan:
-----------
    Total Cost:         72510.8
    Query Degree:       1

                    Rows
                   RETURN
                   ( 1)
                    Cost
                     I/O
                     |
                  0.000713898
                   UPDATE
                   ( 2)
                   72510.8
                   5762.12
                  /-----+-----\
            0.000713898      35019
            x^NLJOIN    TABLE: ACCESSFN
            ( 3)     PS_BP_PST1_TAO13
             72510.8        Q1
             5762.12
           /----------+----------\
        1400.76                0
        FETCH                FETCH
        ( 4)               ( 6)
        486.816              51.445
        159.08                4
       /----+----\          /---+----\
    1400.76      35019      0      6.3082e+06
    IXSCAN   TABLE: ACCESSFN  IXSCAN   TABLE: ACCESSFN
    ( 5)  PS_BP_PST1_TAO13 ( 7)   PS_LEDGER_KK
    228.46       Q3       51.4446      Q2
     65.68                  4
      |                     |
     35019               6.3082e+06
  INDEX: ACCESSFN         INDEX: ACCESSFN
 PSABP_PST1_TAO13          PSBLEDGER_KK
     Q3                     Q2
```

So, what do we have so far?

- ✓ High number of logical data page reads
- ✓ High number of index logical page reads
- ✓ Complaint from user that application is SLOW
- ✓ High USER and SYSTEM and CPU usage
- ✓ Could it be suboptimal SQL
- ✓ Could correct indexes help?
- Next step in a fire fighting drill
  - Explain
  - Design Advisor

Firefighting Drill led to index solution

- ✓ db2advis –d dbname –i hicost.sql –q schema

```
found [1] SQL statements from the input file
Recommending indexes...
total disk space needed for initial set [  15.450] MB
total disk space constrained to         [54032.880] MB
Trying variations of the solution set.
Optimization finished.
  1  indexes in current solution
[16636.0000] timerons  (without recommendations)
[ 39.0000] timerons  (with current solution)
[99.77%] improvement
-- LIST OF RECOMMENDED INDEXES
-- ===========================
-- index[1],   15.450MB
   CREATE INDEX "FNPRDI  "."IDX203242250540000" ON "ACCESSFN"."PS_BP_ACT_TAO13"
   ("DEPTID" ASC, "PROGRAM_CODE" ASC, "OPERATING_UNIT"
   ASC, "CLASS_FLD" ASC, "FUND_CODE" ASC, "ACCOUNT" ASC,
   "BUDGET_REF" ASC, "PRODUCT" ASC, "LEDGER_GROUP" ASC,
   "AFFILIATE_INTRA2" ASC, "AFFILIATE_INTRA1" ASC, "AFFILIATE"
   ASC, "PROCESS_INSTANCE" ASC, "BUDGET_PERIOD" ASC,
   "RESOURCE_TYPE" ASC, "ACTIVITY_ID" ASC, "PROJECT_ID"
   ASC, "BUSINESS_UNIT_PC" ASC, "LIQ_FLG" ASC, "BALANCING_LINE"
   ASC, "STATISTICS_CODE" ASC, "BASE_CURRENCY" ASC, "CHARTFIELD3"
   ASC, "CHARTFIELD2" ASC, "CHARTFIELD1" ASC, "BUSINESS_UNIT"
   ASC, "KK_TRAN_DT" ASC, "KK_TRAN_ID" ASC, "AFFECT_SPEND_OPTN"
   ASC, "KK_SKIP_EDITS" ASC) ALLOW REVERSE SCANS COLLECT SAMPLED DETAILED STATISTICS;
   COMMIT WORK ;
```

Solution

- SQL Rewrite not possible in this case as it is PeopleSoft and business rules prevent rewrite
- Applied new index in DEV, TEST, and QA and ran entire application to ensure benefit of index realized and no impact to other SQL/processes

✓ Reduced part of a 28 hour job by 3 hours

✓ Entire analysis from time of reported problem to recommended solution using previous steps was 5 minutes

Other Methods

- MONREPORT Reporting Module
  - DB2 9.7, DB2 10
- Use one of the 29 SQL Administrative Views or Snapshot Table Functions provided with DB2
  - Returns monitoring data
- Use one of the 13 SQL Administrative Convenience Views and SQL Table Snapshot Functions provided by DB2
  - Returns monitoring data and computed (Convenient!) values

SQL Snapshot Table Functions

```
#!/bin/ksh
db2 connect to dsdm;
db2 "SELECT INTEGER(applsnap.agent_id) AS agent_id,
CAST(LEFT(applinfo.appl_name,10) AS CHAR(10)) AS appl_name,
CAST(left(client_nname,35) AS CHAR(35)) AS nname,
 INTEGER(locks_held) AS locks,applsnap.rows_read as rr, applsnap.rows_written as rw,applsnap.total_sorts as sorts,
applsnap.sort_overflows as oflows, applsnap.lock_timeouts as touts, applsnap.total_hash_loops as loops, applsnap.agent_usr_cpu_time_s as usersecs,
applsnap.agent_sys_cpu_time_s as syscpu, applsnap.locks_waiting as lkwait,
 SUBSTR(APPL_STATUS,1,10) AS APPL_STATUS, SUBSTR(stmt_snap.STMT_TEXT, 1, 999) AS STMT_TEXT
FROM TABLE( sysproc.snap_get_appl('',-1)) AS applsnap,
TABLE( sysproc.snap_get_appl_info('',-1)) as applinfo,
TABLE (sysproc.snap_get_stmt('',-1)) as stmt_snap
 WHERE applinfo.agent_id = applsnap.agent_id
and applinfo.agent_id = stmt_snap.agent_id
and appl_status in ('UOWEXEC','LOCKWAIT')
 ORDER BY appl_status";
db2 connect reset;
```

NOTE: Replace with MON_CURRENT_SQL and MON_CURRENT_UOW Administrative views

Resolving Lock Contention with db2pd

Database Partition 0 -- Database SAMPLE -- Active -- Up 0 days 16:39:33

db2pd –db SAMPLE –locks –file /tmp/lockc.txt

Locks:

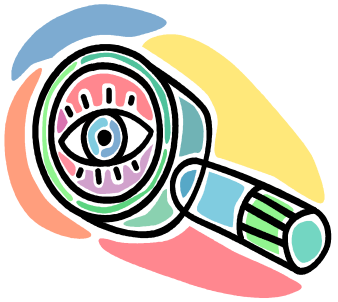
Address	TranHdl	Lockname	Type	Mode	Sts	Owner	Dur	HldCnt	Att	ReleaseFlg
0x0459C510	2	53514C4332453036BD4A32C841	Internal P	..	S	G	2	1	0	0x0000	0x40000000
0x0459CA10	3	53514C4332453036BD4A32C841	Internal P	..	S	G	3	1	0	0x0000	0x40000000
0x0459CA60	3	01000000010000000001007B0056	Internal V	.	S	G	3	1	0	0x0000	0x40000000
0x0459C9E8	3	53514C4445464C5428DD630641	Internal P	.	S	G	3	1	0	0x0000	0x40000000
0x0459EF90	2	02000300270000000000000052	Row	.	X	G	2	1	0	0x0008	0x40000002
0x0459CAB0	3	02000300270000000000000052	Row	.N	S	W	2	1	0	0x0000	0x00000001
0x0459C8F8	2	02000300000000000000000054	Table	.I	X	G	2	1	0	0x0000	0x40000002
0x0459CA88	3	02000300000000000000000054	Table	.I	S	G	3	1	0	0x0000	0x00000001

TranHdl 2 has an X lock on this row

Type of lock

Lock mode

TranHdl 3 is waiting on a lock held by TranHdl 2



-locks showlocks option

Locks:

Address TranHdl Lockname Type Mode Sts Owner Dur HldCnt Att ReleaseFlg

0x0459C510 2 53514C4332453036BD4A32C841 Internal P ..S G 2 1 0 0x0000 0x40000000
Pkg UniqueID 434c5153 36304532 Name c8324abd Loading = 0

0x0459CA10 3 53514C4332453036BD4A32C841 Internal P ..S G 3 1 0 0x0000 0x40000000
Pkg UniqueID 434c5153 36304532 Name c8324abd Loading = 0

0x0459CA60 3 0100000001000000001007B0056 Internal V ..S G 3 1 0 0x0000 0x40000000
Anchor 123 Stmt 1 Env 1 Var 1 Loading 0

0x0459C9E8 3 53514C4445464C5428DD630641 Internal P ..S G 3 1 0 0x0000 0x40000000
Pkg UniqueID 444c5153 544c4645 Name 0663dd28 Loading = 0

0x0459EF90 **2** 020003002700000000000000052 Row **..X G 2** 1 0 0x0008 0x40000002
TbspaceID 2 TableID 3 RecordID 0x27

0x0459CAB0 **3** 020003002700000000000000052 Row **.NS W 2** 1 0 0x0000 0x00000001
TbspaceID 2 TableID 3 RecordID 0x27

0x0459C8F8 2 020003000000000000000000054 Table .IX G 2 1 0 0x0000 0x40000002
TbspaceID 2 TableID 3

0x0459CA88 3 020003000000000000000000054 Table .IS G 3 1 0 0x0000 0x00000001
TbspaceID 2 TableID 3

SNAPLOCKWAIT Administrative View

- db2 connect to dsdm;

db2 " select agent_id, lock_mode, lock_object_type, agent_id_holding_lk,

lock_wait_Start_time, lock_mode_requested from sysibmadm.snaplockwait";

- db2 connect reset;

Replace with new MON_LOCKWAITS administrative view which includes holders, waiters and holder SQL

MONREPORT.LOCKWAIT Stored Procedure

- Part of MONREPORT reporting module introduced in DB2 9.7 FP1
- "DB2 CALL MONREPORT.LOCKWAIT (*monitoring_interval, application_handle*"
  - Default reports on 10 second interval
  - Reports on current lock wait events, holders, waiters and characteristic of locks held
  - No historic data -- use new LOCK event monitor for details
- Output similar to lock snapshot except lock holder and lock waiter SQL is provided

DB2DETAILDEADLOCK Event Monitor Deprecated

- Replaced with new LOCKING event monitor in DB2 9.7- DB2 10
- Create new LOCKING event monitor and DROP the DB2DETAILDEADLOCK event monitor
- DB2 9.7 FP writes to unformatted event monitor
  - Must configure formatting tool
- DB2 10 LOCK event monitor now supports WRITE TO TABLE (regular relational table) event monitor
- Rich set of locking events collected
- Can be collected at the Database level or Workload (service class) level

Long Running SQL Adminstrative View

db2 connect to dsdm;

db2 "SELECT agent_id, authid, elapsed_time_min, appl_status, SUBSTR(STMT_TEXT, 1, 550) AS STMT_TEXT

FROM SYSIBMADM.LONG_RUNNING_SQL where APPL_STATUS in ('UOWEXEC','LOCKWAIT') ORDER BY elapsed_time_min desc";

db2 connect reset;

- The problem here is it is “relative” to what is currently running

New DB2 10 - MONREPORT Stored Procedure Reports

- Monreport.currentapps: (UOW states: Executing, Lock Wait,etc)
- Monreport.connection: (similar to application snapshot)
- Monreport.lockwait: (Lock waiters and holders)
- Monreport.currentsql: (Top 10 SQL currently running with entire SQL)
- Monreport.pkgcache: (Top partial SQL from package cache, per stmt and per execution)

Identify and Tune Top 10 SQL Statements

```
with t (snap_ts, rows_read, num_exec, sys_time, usr_time, exec_time, n_rr, n_ne, n_st,
n_ut, n_te, stmt_text) as (
select snapshot_timestamp, rows_read, num_executions, total_sys_cpu_time,
total_usr_cpu_time, total_exec_time
  , row_number() over (order by rows_read desc)
  , row_number() over (order by num_executions desc)
  , row_number() over (order by total_sys_cpu_time desc)
  , row_number() over (order by total_usr_cpu_time desc)
  , row_number() over (order by total_exec_time desc)
  , substr(stmt_text,1,300)
from sysibmadm.snapdyn_sql as t2
)
select * from t
where n_rr < 11 or n_ne < 11 or n_st < 11 or n_ut < 11 or n_te < 11
;
```

Top 10 SQL Output - Example

SNAP_TS	ROWS_READ	NUM_EXEC	SYS_TIME	USR_TIME	EXEC_TIME	N_RR
						N_NE
			N_ST	N_UT	N_TE	STMT_TEXT
2008-04-08-11.42.50.109894	88422919	4	2	1103	1207	1
2724	4	1	1 SELECT HRS_JOB_OPENING_ID FROM PS_HRS_JO_ALL_I WHERE HRS_JOB_OPENING_ID = ? AND (MANAGER_ID = ? OR RECRUITER_ID =? OR HRS_JOB_OPENING_ID IN (SELECT HRS_JOB_OPENING_ID FROM PS_HRS_JO_TEAM WHERE EMPLID = ?) OR 'HALLL' IN (SELECT OPRID FROM PSOPRDEFN WHERE ROWSECCLASS IN (SELECT ROWSECCLASS FROM PS_						
2008-04-08-11.42.50.109894	76654367	2116	1	501	572	2
97	6	2	3 SELECT FILL.HRS_JOB_OPENING_ID,FILL.OPRID,FILL.EMPLID FROM PS_HRS_JO_SEC_VW FILL WHERE HRS_JOB_OPENING_ID = ? AND OPRID = ?				
2008-04-08-11.42.50.109894	13976336	176	0	40	44	3
498	12	8	15 SELECT T.TYPE, SUM(CASE WHEN TC.ENFORCED='Y' THEN 1 ELSE 0 END) AS CHILDREN, SUM(CASE WHEN TC.ENFORCED='Y' AND R.TABNAME=T.TABNAME AND R.TABSCHEMA=T.TABSCHEMA THEN 1 ELSE 0 END) AS SELFREFS FROM TABLE(SYSPROC.BASE_TABLE('ACCESSHR','PS_TL_IPT15')) B, SYSCAT.TABLES T LEFT OUTER JOIN SYSCAT.REFERENCES				

Tuning the #1 Ranked SQL

```
SELECT HRS_JOB_OPENING_ID FROM ACCESSHR.PS_HRS_JO_ALL_I WHERE HRS_JOB_OPENING_ID = ?
  AND (MANAGER_ID = ? OR RECRUITER_ID =? OR HRS_JOB_OPENING_ID IN ( SELECT
  HRS_JOB_OPENING_ID FROM ACCESSHR.PS_HRS_JO_TEAM WHERE EMPLID = ?) OR 'HALL' IN (
  SELECT OPRID FROM ACCESSHR.PSOPRDEFN WHERE ROWSECCLASS IN ( SELECT ROWSECCLASS FROM
  ACCESSHR.PS_HRS_SEC_TBL WHERE HRS_SEC_SU = 'Y')));

execution started at timestamp 2008-01-28-18.39.32.251421
found [1] SQL statements from the input file
Recommending indexes...
total disk space needed for initial set [  15.091] MB
total disk space constrained to         [22356.627] MB
Trying variations of the solution set.
Optimization finished.
  11  indexes in current solution
```

[2505588.0000] timerons (without recommendations)
[7507.0000] timerons (with current solution)
[99.70%] improvement--

```
--

-- LIST OF RECOMMENDED INDEXES
-- ===========================
-- index[1],    0.743MB
   CREATE INDEX "HRPRDI  "."IDX801282342230000" ON "ACCESSHR"."PS_HRS_JO_TEAM" ("EMPLID" ASC, "HRS_JOB_OPENING_ID" DESC) ALLOW REVERSE SCANS ;
   COMMIT WORK ;
   RUNSTATS ON TABLE "ACCESSHR"."PS_HRS_JO_TEAM" FOR INDEX "HRPRDI  "."IDX801282342230000" ;
   COMMIT WORK ;
-- index[2],    3.056MB
   CREATE UNIQUE INDEX "HRPRDI  "."IDX801282341000000" ON "ACCESSHR"."PS_SJT_OPR_CLS" ("OPRID" ASC, "CLASSID" ASC) ALLOW REVERSE SCANS ;
   COMMIT WORK ;
   RUNSTATS ON TABLE "ACCESSHR"."PS_SJT_OPR_CLS" FOR INDEX "HRPRDI  "."IDX801282341000000" ;
   COMMIT WORK ;
-- index[3],    0.079MB
   CREATE INDEX "HRPRDI  "."IDX801282341560000" ON "ACCESSHR"."PS_SJT_CLASS_ALL" ("SCRTY_SET_CD" ASC, "CLASSID" ASC) ALLOW REVERSE SCANS ;
   COMMIT WORK ;
   RUNSTATS ON TABLE "ACCESSHR"."PS_SJT_CLASS_ALL" FOR INDEX "HRPRDI  "."IDX801282341560000" ;
   COMMIT WORK ;
-- index[4],    8.157MB
   CREATE INDEX "HRPRDI  "."IDX801282341580000" ON "ACCESSHR"."PS_HRS_SJT_JO" ("SCRTY_KEY2" ASC, "SCRTY_KEY1" ASC, "SCRTY_TYPE_CD" ASC, "EMPLID" ASC,
"SCRTY_KEY3" ASC) ALLOW REVERSE SCANS ;
   COMMIT WORK ;
   RUNSTATS ON TABLE "ACCESSHR"."PS_HRS_SJT_JO" FOR INDEX "HRPRDI  "."IDX801282341580000" ;
   COMMIT WORK ;
```

Top 10 SQL Summary

- Use my Top 10 SQL query or MONREPORT.CURRENTSQL report to identify the Top 10 SQL
  - Tune the #1 SQL
- Or, use the SYSIBMADM.TOP_DYNAMIC_SQL Administrative view to identify and tune Top SQL
- TOP 10 SQL tuning process is an iterative process
- Keep tuning until you have done all the Top 10
- New SQL will show-up over time and you will have a new TOP 10 list

Use of Dynamic SQL Snapshot or Administrative View

- "Farm" the Dynamic SQL snapshot or Administrative View for resource intensive queries
- **In 9.7 and DB2 10 Replace snapshot with new MONREPORT.PKGCACHE Report (ranked by num exec, lock wait, I/O wait, rows read, rows modified cumulative and per execution and MON_GET_PKG_CACHE_STMT table function))**

"select num_executions as num_exec, num_compilations as num_comp, prep_time_worst as worst_prep, prep_time_best as best_prep, **rows_read** as rr,

rows_written as rw,**stmt_sorts** as sorts, **sort_overflows** as sort_oflows, total_exec_time as tot_time, total_exec_time_ms as tot_timems, total_usr_cpu_time

as totusertime, total_usr_cpu_time_ms as totusrcpums,

total_sys_cpu_time as sys, total_sys_cpu_time_ms as sysms, total_sys_cpu_time as syscpu, total_sys_cpu_time_ms as syscpums , **substr(stmt_text,1,5999)** as stmt_text from

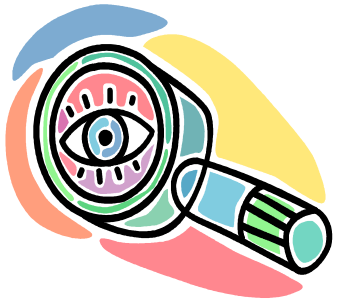
sysibmadm.snapdyn_sql where total_sys_cpu_time > 1 or total_usr_cpu_time > 1 order by total_usr_cpu_time, total_sys_cpu_time,num_compilations, prep_time_worst"

New DB2 9.7 and DB2 10 - MONREPORT Module Stored Procedure Reports

- Monreport.currentsql: (Top 10 SQL currently running with entire SQL)
- Monreport.pkgcache: (Top SQL from package cache, per stmt and per execution, partial SQL)

db2pd –tcbstats

- Used the –tcbstats option to identify tables being scanned, page overflows, highly active tables, index splits, unused indexes, indexes scanned, indexes used for index-only access, index include column usage and types of table activity (Inserts, Deletes, Updates)

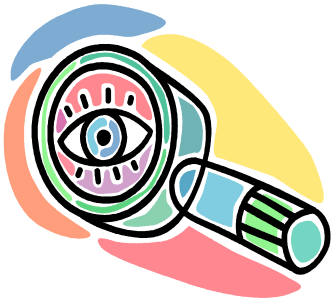


db2pd –db GTS1 -tcbstats Example

TbspaceID	TableID	TableName	Scans	UDI	DataSize	IndexSize	PgReorgs	NoChgUpd	Reads	FscrUpdat	Inserts	Updates	Deletes	OvFlReads	OvFlCrtes	LfSize	LobSize
0	1	SYSBOOT	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	2	SYSTABLI	54	499	51	24	0	0	3819	2	17	15	0	0	0	0	576
0	3	SYSCOLU	0	6820	358	176	0	0	20	24	247	1	0	0	0	0	0
3	2	HMON_AT	196	2507	15	5	56	7225	3802	135	156	2507	0	21581	100	0	0
0	4	SYSINDEX	0	206	20	12	0	0	0	1	4	0	0	0	0	0	1
3	3	POLICY	0	5	1	3	0	0	4	0	5	0	0	0	0	0	64
0	5	SYSCOLP	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
3	4	HMON_CC	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0
0	6	SYSINDEX	0	560	8	12	0	0	0	1	10	0	0	0	0	0	0
3	5	STMG_DB	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0
0	8	SYSVIEW	0	794	21	34	0	0	0	0	0	0	0	0	0	0	0
0	9	SYSPLAN	0	77	9	5	0	0	154	2	1	0	0	0	0	0	320
0	10	SYSPLAN	0	117	5	8	1	0	0	36	16	0	0	0	0	0	64
0	11	SYSSECT	0	302	11	10	0	0	1448	2	32	0	0	0	0	0	2944
0	12	SYSSTMT	0	323	11	10	0	0	0	2	32	0	0	0	0	0	128
0	13	SYSDBAU	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0
0	14	SYSPLAN	0	110	3	8	0	0	0	1	1	0	0	0	0	0	0
0	15	SYSTABA	0	372	10	25	0	0	0	2	17	0	0	0	0	0	0

db2pd -db <dbname> -tcbstats index option

- Command: **db2pd –db GTS1 –tcbstats index**



Address	TableName	IID	PartI	EmpPgDel	RootSplits	BndrySplt	PseuEmpt	EmPgMkd	Scans	IxOnlyScn	KeyUpdat	InclUpdat	NonBndSp	PgAllocs	Merges	PseuDels	DelClean	IntNodSpl
0x000007F	TRANSACTION	11	n/a	0	0	0	0	0	29648	29647	2	0	5	5	0	2	0	0
0x000007F	TRANSACTION	10	n/a	0	0	0	0	0	4696	0	3839	0	33	35	0	3839	3750	2
0x000007F	TRANSACTION	9	n/a	0	0	0	0	0	20886	20858	2466	0	69	70	0	2466	2474	1
0x000007F	TRANSACTION	8	n/a	0	0	0	0	0	202	202	2466	0	26	27	0	2466	2494	1
0x000007F	TRANSACTION	7	n/a	0	0	0	0	0	84	84	3748	0	5	5	0	3748	3796	0
0x000007F	TRANSACTION	6	n/a	0	0	0	0	0	1460	1460	0	0	7	7	0	0	0	0
0x000007F	TRANSACTION	5	n/a	0	0	0	0	0	0	0	25	0	29	30	0	25	25	1
0x000007F	TRANSACTION	4	n/a	0	0	0	0	0	46	0	1100	0	24	25	0	1100	1052	1
0x000007F	TRANSACTION	3	n/a	0	0	0	0	0	34937	34937	1246	0	15	15	0	1246	1188	0
0x000007F	TRANSACTION	2	n/a	0	0	0	0	0	443	306	2466	0	24	24	0	2466	2470	0
0x000007F	TRANSACTION	1	n/a	0	0	16	0	0	11081	0	0	0	0	16	0	0	0	0

Identify Unused Indexes using SYSCAT.INDEXES view

- "db2 describe table syscat.indexes"
- "select lastused,indname, tabname from syscat.indexes where lastused > '2012-01-01'" (note: Available in DB2 9.7 and above)
- Great feature for identifying unused indexes for large applications like PeopleSoft and SAP
- Review unused indexes with application developers and known weekly, monthly or yearly processes to prevent accidental drop of used index
- But, by all means, get rid of unused indexes!

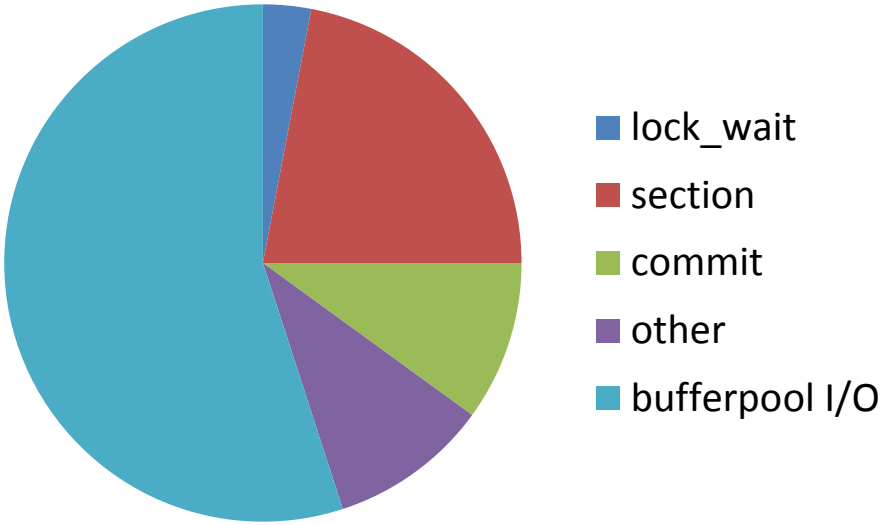
LASTUSED Column of SYSCAT.INDEXES 9.7 FP3a and below

- Column does not reflect last used data correctly if indexes created in a different table space than table
- Fix is to apply fix pack 4
  - https://www-304.ibm.com/support/docview.wss?uid=swg1IC70265

DB2 9.7 New Time-spent Monitoring

- New monitoring infrastructure and DB CFG parameters provide database-wide monitoring control
- New relational monitoring functions are lightweight and SQL accessible
- Information about work performed by applications is collected and reported through table function interfaces at three levels
  - System level
    - Details about worked performed on the system
    - Service subclass, workload definition, uow and connection
  - Activity level
    - Details about a subset of work being performed on the system
  - Data object level
    - Details of work within specific objects
      - Indexes, tables, bufferpools, tablespaces and containers

Where is the time being spent?



Monitor Collection DB CFG Parameters

- Mon_act_metrics – controls collection of activity level monitor elements on the entire database (DEFAULT – BASE)
  - MON_GET_ACTIVITY_DETAILS
  - MON_GET_PKG_CACHE_STMT
  - Activity event monitor (DETAILS_XML monitor element in the event_activity logical data groups)
- Mon_deadlock – controls generation of deadlock events on the entire database (DEFAULT-WITHOUT_HIST)
- Mon_locktimeout – controls generation of lock timeout events on the entire database (DEFAULT – NONE)
- Mon_lockwait – controls generation of lock wait events for the lock event monitor (DEFAULT – NONE)
- Mon_lw_thresh – the amount of time spent in lock wait before an event for mon_lockwait is generated (DEFAULT - 5000000)
- Mon_obj_metrics – controls collection of data object monitor elements on the entire database (DEFAULT- BASE)
  - MON_GET_BUFFERPOOL
  - MON_GET_TABLESPACE
  - MON_GET_CONTAINER

MON_GET_ACTIVITY_DETAILS

- Use this table function to get similar data as that obtained from an application snapshot, plus much more detailed information not available in past releases
  - Log_buffer_wait_times
  - Num_log_buffer_full
  - Log_disk_wait_time
  - Log_disk_wait_time_total
  - Lock_escals
  - Lock_timeouts
- In 9.7, activity metrics were stored in the DETAILS_XML column and had to be converted to a relational format by the XMLTABLE function
- As of 9.7 FP4, activity metrics can now be collected in a table and queried with SQL directly

Monitor Collection DB CFG Parameters

- Mon_req_metrics – controls the collection of request monitor elements on the entire database (DEFAULT – BASE)
  - MON_GET_UNIT_OF_WORK
  - MON_GET_UNIT_OF_WORK_DETAILS
  - MON_GET_CONNECTION
  - MON_GET_CONNECTION_DETAILS
  - MON_GET_SERVICE_SUBCLASS
  - MON_GET_SERVICE_SUBCLASS_DETAILS
  - MON_GET_WORKLOAD
  - MON_GET_WORKLOAD_DETAILS
  - Statistics event monitor (DETAILS_XML monitor element in the event_wlstats and event_scstats logical data groups)
  - Unit of work event monitor
- Mon_uow_data – controls the generation of UOW events at the database level for the UOW event monitor (DEFAULT – NONE)

MON_GET_ACTIVITY_DETAILS Usage

- Get the application handle, activity ID and UOW ID using the table function: wlm_get_workload_occurrence_activities_v97

```
 "select application_handle, activity_id, uow_id, local_Start_time from
table(wlm_get_workload_occurrence_activities_v97(Cast (null as bigint), -1) ) as t

APPLICATION_HANDLE   ACTIVITY_ID UOW_ID      LOCAL_START_TIME
-------------------- ----------- ----------- --------------------------
             63595           1          28 2012-04-12-13.01.47.400679

  1 record(s) selected.
```

MON_GET_ACTIVITY_DETAILS cont.

```
SELECT actmetrics.application_handle,
  actmetrics.activity_id,
  actmetrics.uow_id,
  varchar(actmetrics.stmt_text, 400) as stmt_text,
  actmetrics.total_act_time,
  actmetrics.total_act_wait_time,
  CASE WHEN actmetrics.total_act_time > 0
    THEN DEC((
      FLOAT(actmetrics.total_act_wait_time) /
      FLOAT(actmetrics.total_act_time)) * 100, 5, 2)
    ELSE NULL
  END AS PERCENTAGE_WAIT_TIME
FROM TABLE(MON_GET_ACTIVITY_DETAILS(63595, 28, 1, -2)) AS ACTDETAILS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
  '$actmetrics/db2_activity_details'
  PASSING XMLPARSE(DOCUMENT ACTDETAILS.DETAILS) as "actmetrics"
  COLUMNS "APPLICATION_HANDLE" INTEGER PATH 'application_handle',
    "ACTIVITY_ID" INTEGER PATH 'activity_id',
    "UOW_ID" INTEGER PATH 'uow_id',
    "STMT_TEXT" VARCHAR(1024) PATH 'stmt_text',
    "TOTAL_ACT_TIME" INTEGER PATH 'activity_metrics/total_act_time',
    "TOTAL_ACT_WAIT_TIME" INTEGER PATH 'activity_metrics/total_act_wait_time'
  ) AS ACTMETRICS;
```

DB2 10 Event Monitor Enhancements

- All event monitors support write-to-table format
- Can be altered to capture additional logical data groups
- Can be upgraded from previous releases
  - EVMON_UPGRADE_TABLES stored procedure
- New Change History event monitor
  - Tracks DDL, Configuration, Registry and Utilities
- Pruning of data from Unformatted Event Monitor tables
  - Use PRUNE_UE_TABLES option of the EVMON_FORMAT_UE_TO_TABLES stored procedure
- New DB2 10 Usage List object

Evaluate my session online:
www.idug.org/na2013/eval

Philip K. Gunning

Gunning Technology Solutions, LLC

pgunning@gts1consulting.com

Session C2

Title DB2 LUW Monitoring Essentials