Toronto Users Group - March 19 2014
DB2 Performance: Utilizing DB2 for i More Efficiently

Speaker Name Linda M Swan
lmswan@us.ibm.com

New Wiki for DB2 Enhancements via PTF

- **Regularly check (or Subscribe) to the DB2 for i Updates Wiki!**
  - Contains details on new PTFs that deliver new DB2 capabilities
  - Examples:
    - CANCEL_SQL system stored procedure
    - PROGRAM NAME keyword for controlling SQL Triggers Program Name
    - SQL Query Engine 6.1 support for Logical File on FROM clause
    - New RUNSQL CL command
  - Wiki URL:

    https://www.ibm.com/developerworks/systems/ibmi/techupdates/db2

- **The wiki is part of the IBM i zone in IBM developerWorks launched in April 2011**

  https://www.ibm.com/developerworks/systems/ibmi/

- **New DB2 for i Blog too! - http://db2fori.blogspot.com/**

Agenda

- **Why care about DB performance**
- **Causes of poor DB performance**
- **What tools are useful**
- **Things to look for**
- **DB performance tips**

Why care about DB performance

- **Key objective is to provide *Information***
  - Which means processing data
    - Sometimes very large quantities of data
  - In timely manner
  - Using resources efficiently
    - Both systems and personnel
  - To accommodate growth – SQL set at a time
    - At the Transactions
    - At the Data volume level
    - **Complexity** of **requests**

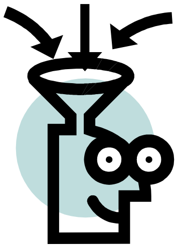
Typical causes of DB performance problems

1. **Accessing more data than needed**
2. **Physical disk operations are consecutive (versus concurrent)**
3. **Trying to consume more resource than is available, or more than your share.**
4. **Under utilizing hardware or not using it efficiently**
5. **High full opens or access plan rebuilds**
6. **Inefficient access plans – does the optimizer have what it needs?**

What tools would I use

- **Collection Services**
- **Job Watcher**
- **Disk Watcher**
- **Performance Explorer**
- **Plan Cache Analyzer**
  - Available through
    - IBM Navigator for i - Performance Data Investigator
    - iDoctor
- **iNavigator – SQL and DBA specific tools**
  - Plan Cache, Visual Explain, Index advisor….

What tools would I use …



- **Application knowledge**

- **Data knowledge**

Monitor, Analyze and Tune

- **Needs to be someone's Job**
  - – DB engineer position
  - – OR, cross team (Operations, Developers)

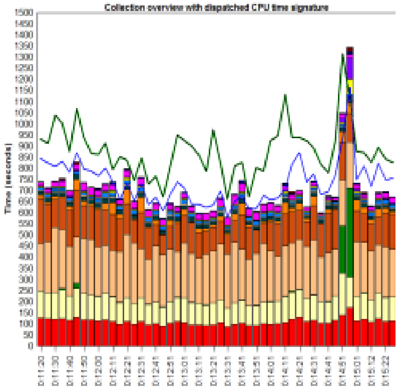
- **Get the right training**
  - **iDoctor workshop**
  - **DB2 for i SQL Performance Workshop**
    - http://www-03.ibm.com/systems/i/software/db2/db2performance.html

- ***Need to establish best practices.***
  - – *One being to capture performance data expectations* <u>*before*</u> *major changes*

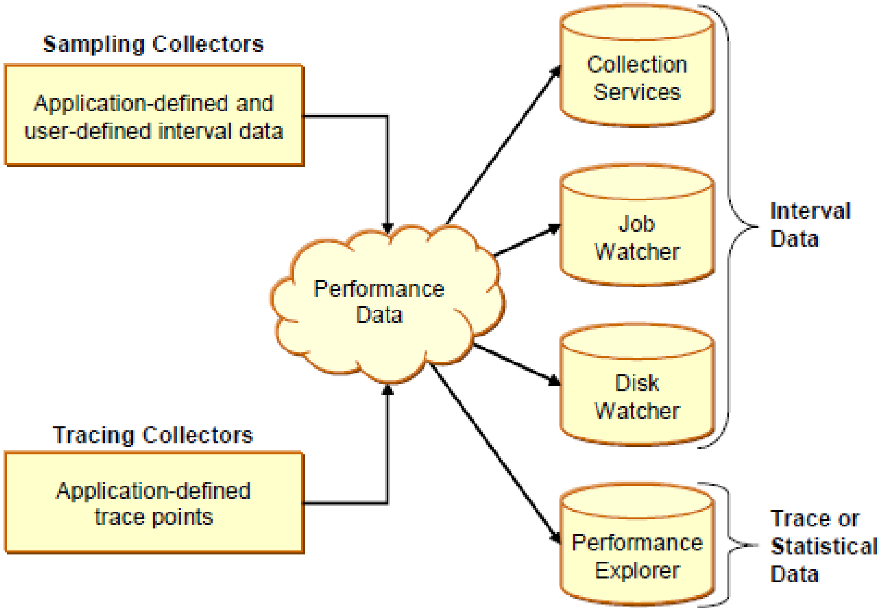
What to look for ?

Look at System Resources

- –Memory
  - WRKSYSSTS
- –I/O Disk Opts
  - WRKDSKSTS
- –High CPU
  - WRKSYSACT



Collect Based on What You Find/*Suspect*



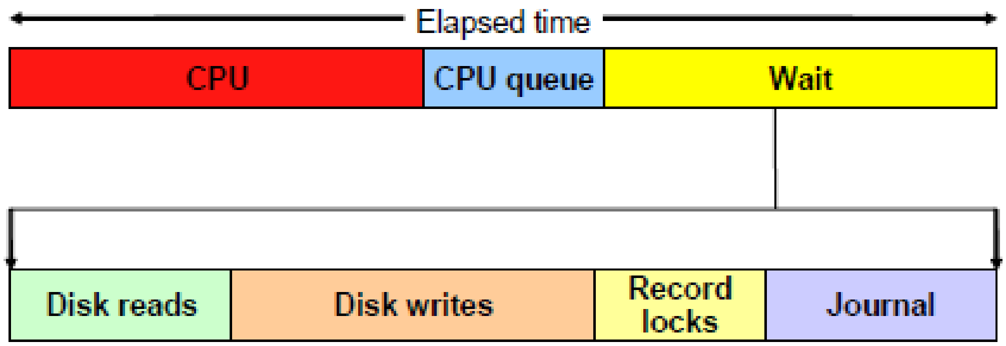
The collections at a glance

Collection Services	Disk Watcher	Job Watcher	Performance Explorer
• Sample data • Support for small intervals • No information concerning specific I/O operations	• Statistics as well as trace data • Focus on disk data	• Sample data • Support for very small intervals • Focus on job data • Call Stacks • SQL Statements • Wait buckets	• Trace data • Information collected for every I/O event • Collection and analysis complex

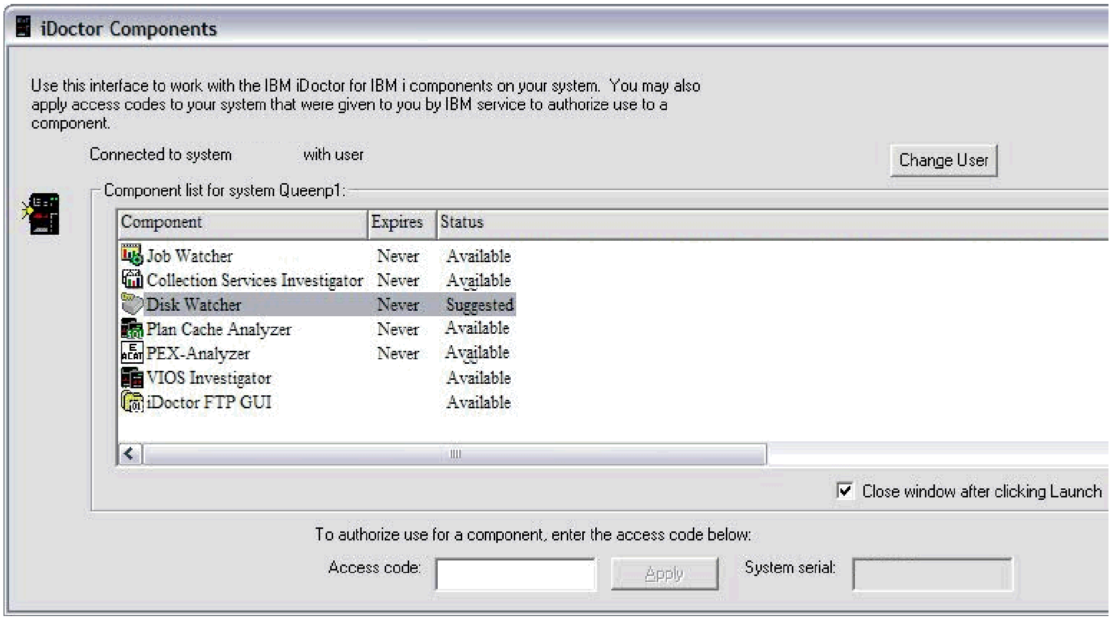
less detail ⟷ more detail

The collections at a glance

Example: Job are typically
- using CPU
- queued up waiting to use CPU
- in an idle wait
- waiting for something specific…JW wait buckets



iDoctor



iDoctor Components

Use this interface to work with the IBM iDoctor for IBM i components on your system. You may also apply access codes to your system that were given to you by IBM service to authorize use to a component.

Connected to system with user

Component list for system Queenp1:

Component	Expires	Status
Job Watcher	Never	Available
Collection Services Investigator	Never	Available
Disk Watcher	Never	Suggested
Plan Cache Analyzer	Never	Available
PEX-Analyzer	Never	Available
VIOS Investigator		Available
iDoctor FTP GUI		Available

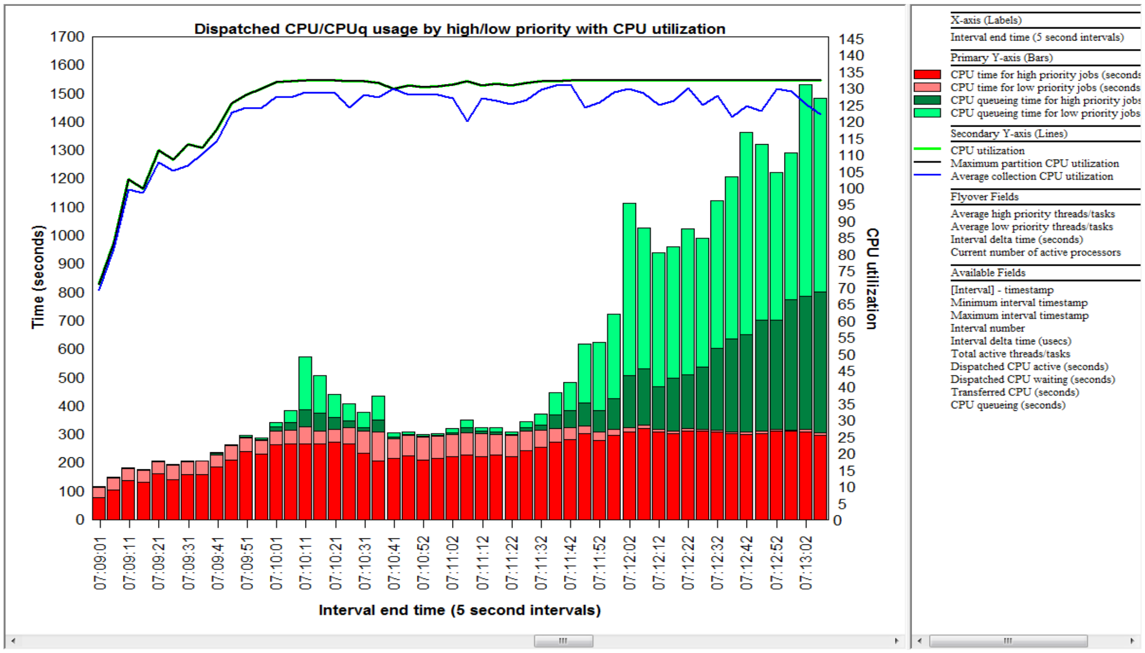
☑ Close window after clicking Launch

To authorize use for a component, enter the access code below:

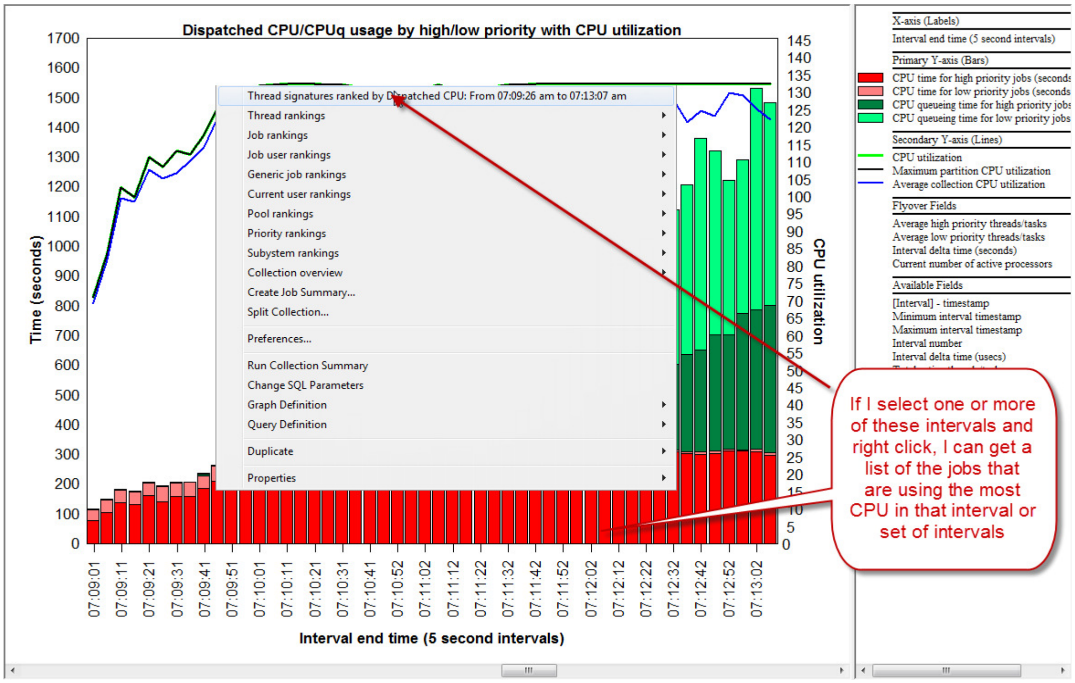
Access code:

System serial:

Symptom Example – High CPU
Job Watcher→ CPU Graphs→ dispatched CPU/CPUQ by high/Low Prior..

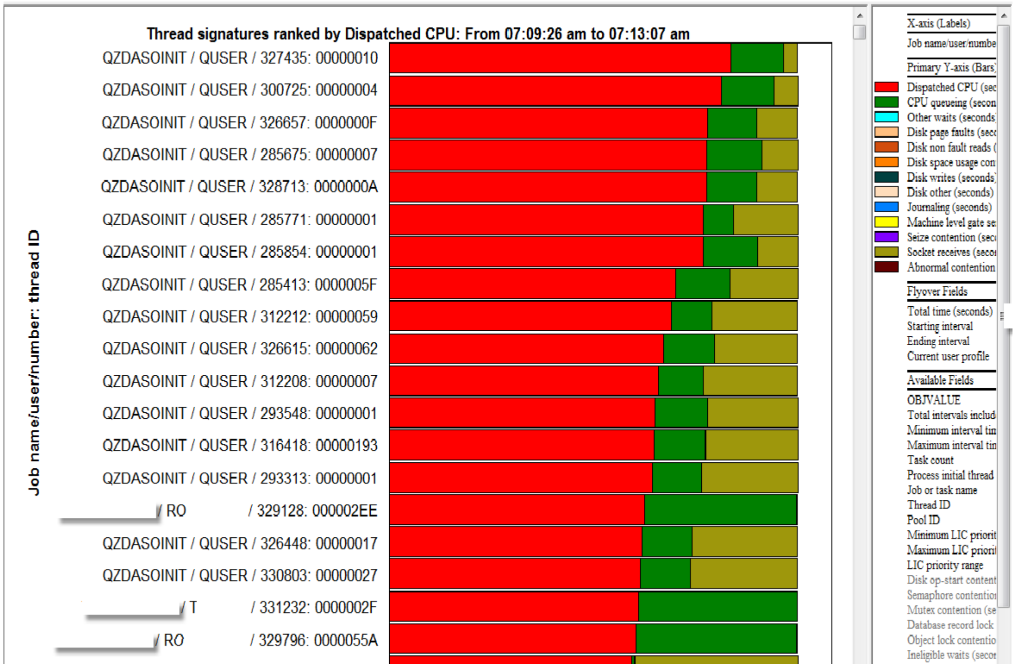


Symptom Example – High CPU
Drill into an interval or set of intervals

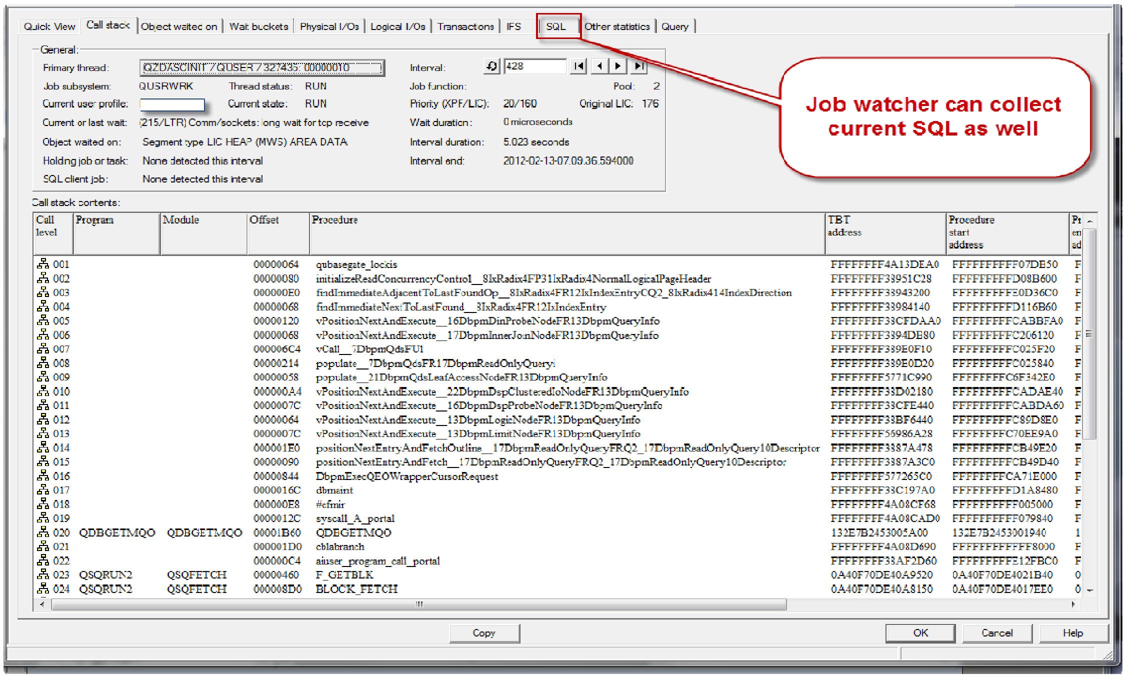


If I select one or more of these intervals and right click, I can get a list of the jobs that are using the most CPU in that interval or set of intervals

Symptom Example – High CPU
(JW) From list of jobs I can see call stacks to get an idea of What is running


Thread signatures ranked by Dispatched CPU: From 07:09:26 am to 07:13:07 am
Job name/user/number: thread ID
QZDASOINIT / QUSER / 327435: 00000010
QZDASOINIT / QUSER / 300725: 00000004
QZDASOINIT / QUSER / 326657: 0000000F
QZDASOINIT / QUSER / 285675: 00000007
QZDASOINIT / QUSER / 328713: 0000000A
QZDASOINIT / QUSER / 285771: 00000001
QZDASOINIT / QUSER / 285854: 00000001
QZDASOINIT / QUSER / 285413: 0000005F
QZDASOINIT / QUSER / 312212: 00000059
QZDASOINIT / QUSER / 326615: 00000062
QZDASOINIT / QUSER / 312208: 00000007
QZDASOINIT / QUSER / 293548: 00000001
QZDASOINIT / QUSER / 316418: 00000193
QZDASOINIT / QUSER / 293313: 00000001
[redacted] / RO / 329128: 000002EE
QZDASOINIT / QUSER / 326448: 00000017
QZDASOINIT / QUSER / 330803: 00000027
[redacted] / T / 331232: 0000002F
[redacted] / RO / 329796: 0000055A
X-axis (Labels)
Job name/user/numbe
Primary Y-axis (Bars)
🟥 Dispatched CPU (sec
🟩 CPU queueing (secon
🟦 Other waits (seconds
🟧 Disk page faults (sec
🟫 Disk non fault reads (
🟧 Disk space usage con
🟩 Disk writes (seconds)
🟧 Disk other (seconds)
🟦 Journaling (seconds)
🟨 Machine level gate se
🟪 Seize contention (sec
🟨 Socket receives (secon
🟫 Abnormal contention
Flyover Fields
Total time (seconds)
Starting interval
Ending interval
Current user profile
Available Fields
OBJVALUE
Total intervals includ
Minimum interval tin
Maximum interval tin
Task count
Process initial thread
Job or task name
Thread ID
Pool ID
Minimum LIC priorit
Maximum LIC priorit
LIC priority range
Disk op-start content
Semaphore contentio
Mutex contention (se
Database record lock
Object lock contentio
Ineligible waits (secon

Symptom Example – High CPU
I can have job watcher collect current SQL too



General:

Primary thread: QZDASCINIT / QUSER / 327435: 00000010
Interval: 428
Job subsystem: QUSRWRK
Thread status: RUN
Job function:
Pool: 2
Current user profile:
Current state: RUN
Priority (XPF/LIC): 20/160
Original LIC: 176
Current or last wait: (215/LTR) Comm/sockets: long wait for tcp receive
Wait duration: 0 microseconds
Object waited on: Segment type LIC HEAP (MWS) AREA DATA
Interval duration: 5.023 seconds
Holding job or task: None detected this interval
Interval end: 2012-02-13-07.09.36.594000
SQL client job: None detected this interval

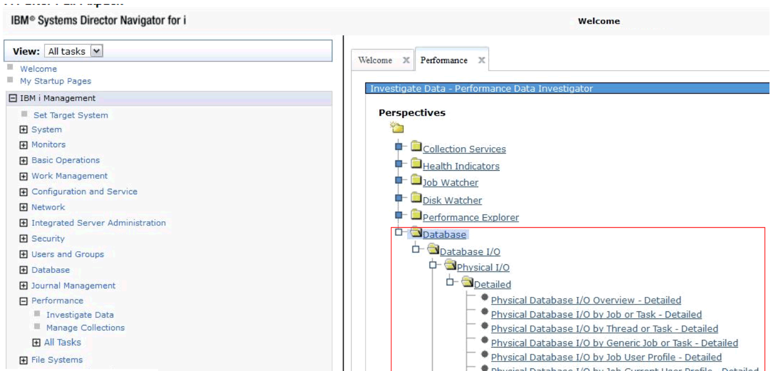
Call stack contents:

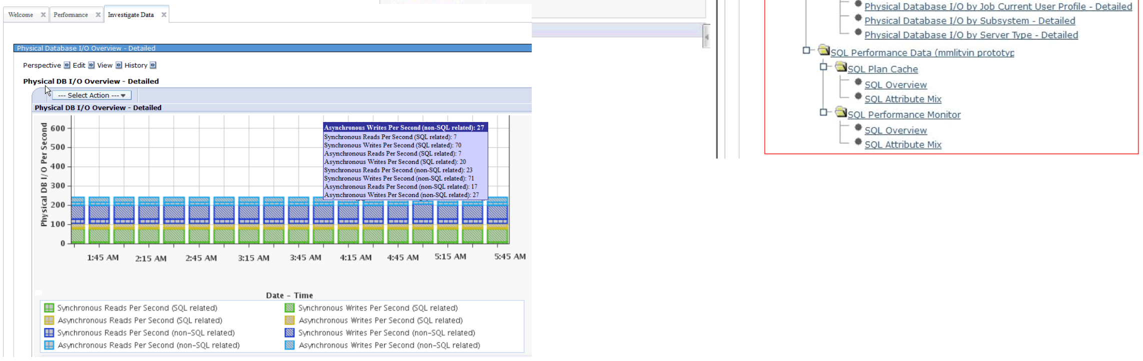
Call level	Program	Module	Offset	Procedure	TBT address	Procedure start address	Pr en ad
001			00000064	qubasegate_lockis	FFFFFFFF4A13DEA0	FFFFFFFFFF07DB50	F
002			00000080	initializeReadConcurrencyControl__8IxRadix4FP31IxRadix4NormalLogicalPageHeader	FFFFFFFF38951C28	FFFFFFFFFD08B600	F
003			000000E0	findImmediateAdjacentToLastFoundOp__8IxRadix4FR12IxIndexEntryCQ2_8IxRadix414IndexDirection	FFFFFFFF38943200	FFFFFFFFFE0D36C0	F
004			00000068	findImmediateNextToLastFound__8IxRadix4FR12IxIndexEntry	FFFFFFFF38984140	FFFFFFFFFD116B60	F
005			00000120	vPositionNextAndExecute__16DbpmDinProbeNodeFR13DbpmQueryInfo	FFFFFFFF38CFDAA0	FFFFFFFFFCABBFA0	F
006			00000068	vPositionNextAndExecute__17DbpmInnerJoinNodeFR13DbpmQueryInfo	FFFFFFFF3894DE80	FFFFFFFFFC206120	F
007			000006C4	vCall__7DbpmQdsFUl	FFFFFFFF389E0F10	FFFFFFFFFC025F20	F
008			00000214	populate__7DbpmQdsFR17DbpmReadOnlyQuery	FFFFFFFF389E0D20	FFFFFFFFFC025840	F
009			00000058	populate__21DbpmQdsLeafAccessNodeFR13DbpmQueryInfo	FFFFFFFF5771C990	FFFFFFFFFC6F342E0	F
010			000000A4	vPositionNextAndExecute__22DbpmDspClusteredIoNodeFR13DbpmQueryInfo	FFFFFFFF38D02180	FFFFFFFFFCADAE40	F
011			0000007C	vPositionNextAndExecute__16DbpmDspProbeNodeFR13DbpmQueryInfo	FFFFFFFF38CFE440	FFFFFFFFFCABDA60	F
012			00000064	vPositionNextAndExecute__13DbpmLogicNodeFR13DbpmQueryInfo	FFFFFFFF38BF6440	FFFFFFFFFC89D8E0	F
013			0000007C	vPositionNextAndExecute__13DbpmLimitNodeFR13DbpmQueryInfo	FFFFFFFF56986A28	FFFFFFFFFC70EE9A0	F
014			000001E0	positionNextEntryAndFetchOutline__17DbpmReadOnlyQueryFRQ2_17DbpmReadOnlyQuery10Descriptor	FFFFFFFF3887A478	FFFFFFFFFCB49E20	F
015			00000090	positionNextEntryAndFetch__17DbpmReadOnlyQueryFRQ2_17DbpmReadOnlyQuery10Descriptor	FFFFFFFF3887A3C0	FFFFFFFFFCB49D40	F
016			00000844	DbpmExecQEOWrapperCursorRequest	FFFFFFFF577265C0	FFFFFFFFCA71E000	F
017			0000016C	dbmaint	FFFFFFFF38C197A0	FFFFFFFFFD1A8480	F
018			000000E8	#cfmir	FFFFFFFF4A08CF68	FFFFFFFFFF005000	F
019			0000012C	syscall_A_portal	FFFFFFFF4A08CAD0	FFFFFFFFFF079840	F
020	QDBGETMQO	QDBGETMQO	00001B60	QDBGETMQO	132E7B2453005A00	132E7B2453001940	1
021			000001D0	cblabranch	FFFFFFFF4A08D690	FFFFFFFFFFFF8000	F
022			000000C4	aiuser_program_call_portal	FFFFFFFF38AF2D60	FFFFFFFFFE12FBC0	F
023	QSQRUN2	QSQFETCH	00000460	F_GETBLK	0A40F70DE40A9520	0A40F70DE4021B40	0
024	QSQRUN2	QSQFETCH	000008D0	BLOCK_FETCH	0A40F70DE40A8150	0A40F70DE4017EE0	0

Job watcher can collect current SQL as well

Performance Data Investigator – New Database Perspective

- Robust graphical interface enabled by Collection Services support.
- New breakdown makes it easier to notice “what changed”?


IBM® Systems Director Navigator for i
Welcome
View: All tasks
Welcome
My Startup Pages
IBM i Management
Set Target System
System
Monitors
Basic Operations
Work Management
Configuration and Service
Network
Integrated Server Administration
Security
Users and Groups
Database
Journal Management
Performance
Investigate Data
Manage Collections
All Tasks
File Systems
Welcome
Performance
Investigate Data - Performance Data Investigator
Perspectives
Collection Services
Health Indicators
Job Watcher
Disk Watcher
Performance Explorer
Database
Database I/O
Physical I/O
Detailed
Physical Database I/O Overview - Detailed
Physical Database I/O by Job or Task - Detailed
Physical Database I/O by Thread or Task - Detailed
Physical Database I/O by Generic Job or Task - Detailed
Physical Database I/O by Job User Profile - Detailed
Physical Database I/O by Job Current User Profile - Detailed
Physical Database I/O by Subsystem - Detailed
Physical Database I/O by Server Type - Detailed


SQL Performance Data (mmlitvin prototyp
SQL Plan Cache
SQL Overview
SQL Attribute Mix
SQL Performance Monitor
SQL Overview
SQL Attribute Mix
Welcome
Performance
Investigate Data
Physical Database I/O Overview - Detailed
Perspective
Edit
View
History
Physical DB I/O Overview - Detailed
--- Select Action ---
Physical DB I/O Overview - Detailed
Physical DB I/O Per Second
600
500
400
300
200
100
0
1:45 AM
2:15 AM
2:45 AM
3:15 AM
3:45 AM
4:15 AM
4:45 AM
5:15 AM
5:45 AM
Date - Time
Synchronous Reads Per Second (SQL related)
Asynchronous Reads Per Second (SQL related)
Synchronous Reads Per Second (non-SQL related)
Asynchronous Reads Per Second (non-SQL related)
Synchronous Writes Per Second (SQL related)
Asynchronous Writes Per Second (SQL related)
Synchronous Writes Per Second (non-SQL related)
Asynchronous Writes Per Second (non-SQL related)

Asynchronous Writes Per Second (non-SQL related): 27
Synchronous Reads Per Second (SQL related): 7
Synchronous Writes Per Second (SQL related): 70
Asynchronous Reads Per Second (SQL related): 7
Asynchronous Writes Per Second (SQL related): 20
Synchronous Reads Per Second (non-SQL related): 23
Synchronous Writes Per Second (non-SQL related): 71
Asynchronous Reads Per Second (non-SQL related): 17
Asynchronous Writes Per Second (non-SQL related): 27

Getting started with CS, JW and iDoctor

- **Get the redbook for**
  - Job Watcher and
  - End To End Performance Management on IBM i
- **Know about iDoctor links**
  - www.youtube.com/user/IBMiDoctor
- **Use the GUI interface, easier to drill and understand data/labels, contains some custom profiles for capturing the right data**
- **v7.1 PTFs more SQL metrics, instrumentation in Collection Services**
- **Experience makes the best teacher**

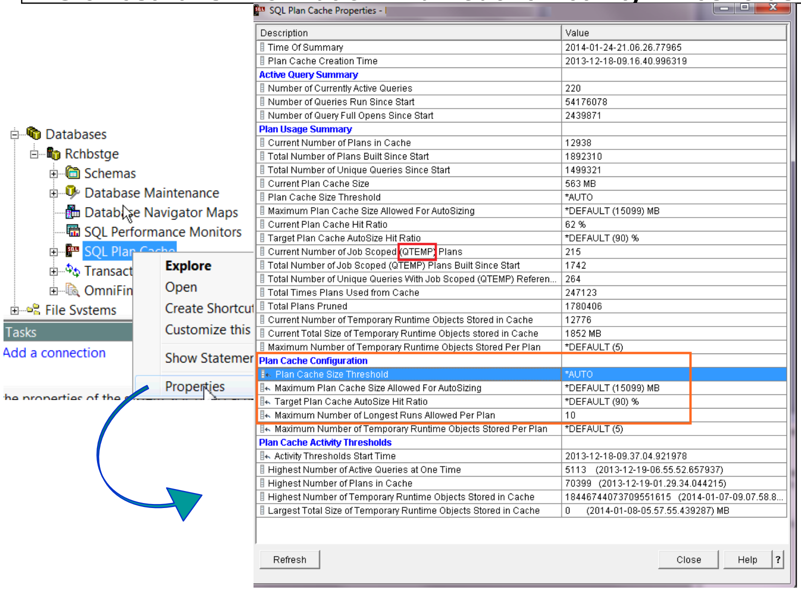
What to Look for - How to Analyze and Tune

- **Time Consuming SQL or queries -**
  - Where to start… SQE Plan Cache
    - Long running SQL
    - High usage (often extremely short running)
    - Those scoped to job only tables…QTEMP awareness
    - Temporary index use

Proactive vs Reactive

SQL Plan Cache properties – Greater insight & control

- **Autosizing enhancement**
  - The default hit ratio changes from 70%→90%
  - The maximum size allowed for autosizing, which was previously set to a hard coded value, is now calculated based on partition size.
- **Slowest runs information. Plan Cache Activity Thresholds**

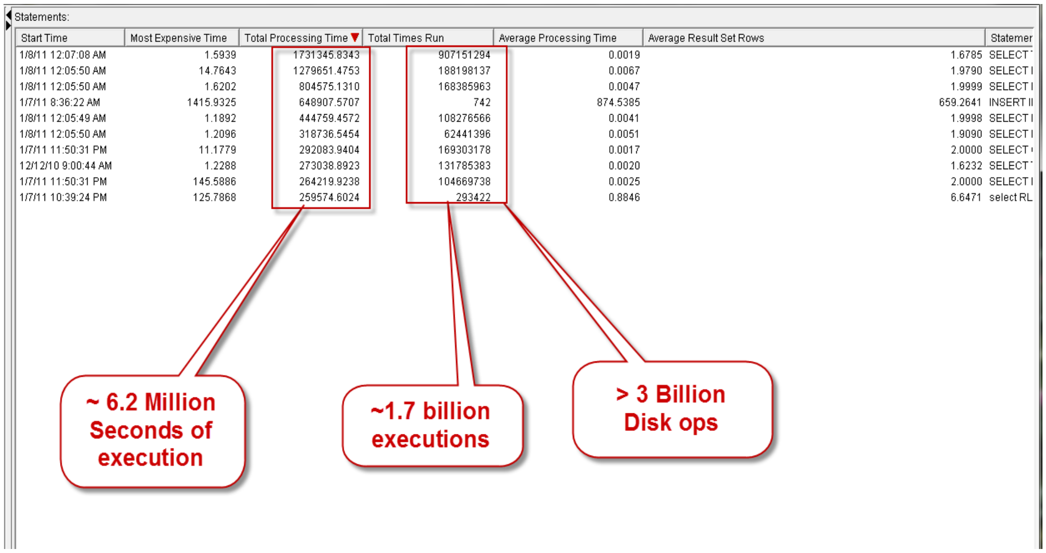


Description	Value
Time Of Summary	2014-01-24-21.06.26.77965
Plan Cache Creation Time	2013-12-18-09.16.40.996319
Active Query Summary	
Number of Currently Active Queries	220
Number of Queries Run Since Start	54176078
Number of Query Full Opens Since Start	2439871
Plan Usage Summary	
Current Number of Plans in Cache	12938
Total Number of Plans Built Since Start	1892310
Total Number of Unique Queries Since Start	1499321
Current Plan Cache Size	563 MB
Plan Cache Size Threshold	*AUTO
Maximum Plan Cache Size Allowed For AutoSizing	*DEFAULT (15099) MB
Current Plan Cache Hit Ratio	62 %
Target Plan Cache AutoSize Hit Ratio	*DEFAULT (90) %
Current Number of Job Scoped (QTEMP) Plans	215
Total Number of Job Scoped (QTEMP) Plans Built Since Start	1742
Total Number of Unique Queries With Job Scoped (QTEMP) Referen...	264
Total Times Plans Used from Cache	247123
Total Plans Pruned	1780406
Current Number of Temporary Runtime Objects Stored in Cache	12776
Current Total Size of Temporary Runtime Objects stored in Cache	1852 MB
Maximum Number of Temporary Runtime Objects Stored Per Plan	*DEFAULT (5)
Plan Cache Configuration	
Plan Cache Size Threshold	*AUTO
Maximum Plan Cache Size Allowed For AutoSizing	*DEFAULT (15099) MB
Target Plan Cache AutoSize Hit Ratio	*DEFAULT (90) %
Maximum Number of Longest Runs Allowed Per Plan	10
Maximum Number of Temporary Runtime Objects Stored Per Plan	*DEFAULT (5)
Plan Cache Activity Thresholds	
Activity Thresholds Start Time	2013-12-18-09.37.04.921978
Highest Number of Active Queries at One Time	5113 (2013-12-19-06.55.52.657937)
Highest Number of Plans in Cache	70399 (2013-12-19-01.29.34.044215)
Highest Number of Temporary Runtime Objects Stored in Cache	18446744073709551615 (2014-01-07-09.07.58.8...
Largest Total Size of Temporary Runtime Objects Stored in Cache	0 (2014-01-08-05.57.55.439287) MB

Customizable

- Maximum Autosize
- Target Hit Ratio
- # of longest runs pre plan to keep in the cache

Plan Cache Example – High use statements
Filter and ask for top N most time consuming statements



Start Time	Most Expensive Time	Total Processing Time ▼	Total Times Run	Average Processing Time	Average Result Set Rows	Statemer
1/8/11 12:07:08 AM	1.5939	1731345.8343	907151294	0.0019	1.6785	SELECT
1/8/11 12:05:50 AM	14.7643	1279651.4753	188198137	0.0067	1.9790	SELECT I
1/8/11 12:05:50 AM	1.6202	804575.1310	168385963	0.0047	1.9999	SELECT I
1/7/11 8:36:22 AM	1415.9325	648907.5707	742	874.5385	659.2641	INSERT II
1/8/11 12:05:49 AM	1.1892	444759.4572	108276566	0.0041	1.9998	SELECT I
1/8/11 12:05:50 AM	1.2096	318736.5454	62441396	0.0051	1.9090	SELECT I
1/7/11 11:50:31 PM	11.1779	292083.9404	169303178	0.0017	2.0000	SELECT
12/12/10 9:00:44 AM	1.2288	273038.8923	131785383	0.0020	1.6232	SELECT
1/7/11 11:50:31 PM	145.5886	264219.9238	104669738	0.0025	2.0000	SELECT I
1/7/11 10:39:24 PM	125.7868	259574.6024	293422	0.8846	6.6471	select RL

~ 6.2 Million Seconds of execution

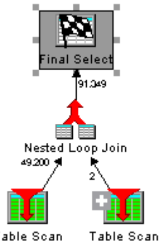
~1.7 billion executions

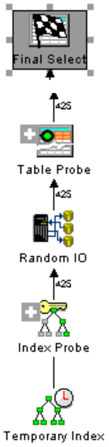
> 3 Billion Disk ops

Tuning these can have a major impact on the system !!

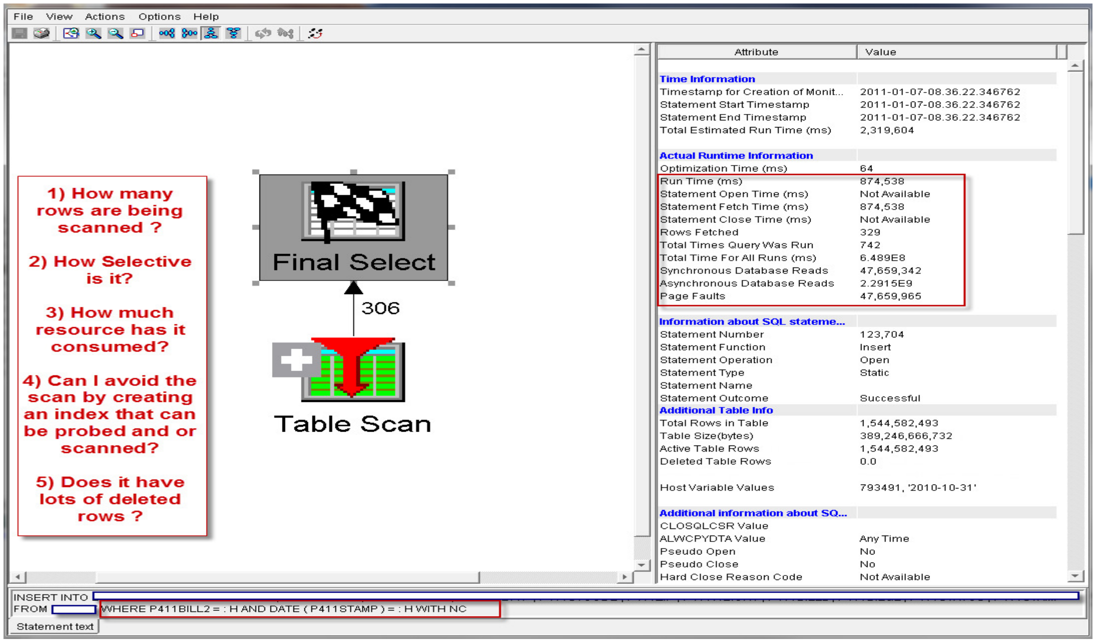
Long running SQL – Examine the access plans

- **What to look for**
  - Full table scans
  - Full index scans
  - Temp index builds
  - High use statements needing better indexes
    - Index Only Access





Look for expensive table scan



1) How many rows are being scanned ?

2) How Selective is it?

3) How much resource has it consumed?

4) Can I avoid the scan by creating an index that can be probed and or scanned?

5) Does it have lots of deleted rows ?

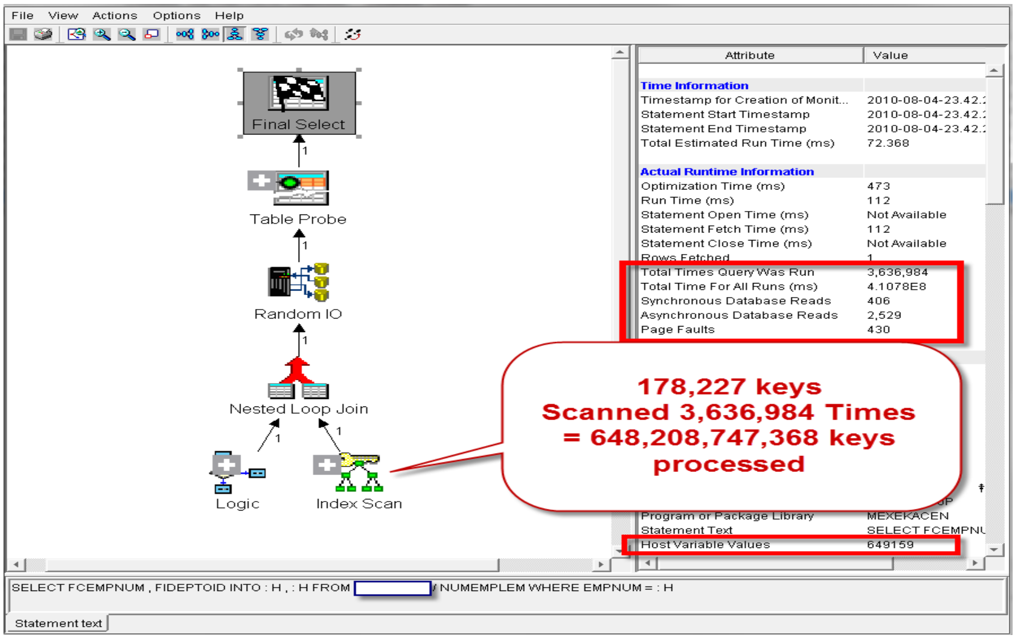
Attribute	Value
Time Information	
Timestamp for Creation of Monit...	2011-01-07-08.36.22.346762
Statement Start Timestamp	2011-01-07-08.36.22.346762
Statement End Timestamp	2011-01-07-08.36.22.346762
Total Estimated Run Time (ms)	2,319,604
Actual Runtime Information	
Optimization Time (ms)	64
Run Time (ms)	874,538
Statement Open Time (ms)	Not Available
Statement Fetch Time (ms)	874,538
Statement Close Time (ms)	Not Available
Rows Fetched	329
Total Times Query Was Run	742
Total Time For All Runs (ms)	6.489E8
Synchronous Database Reads	47,659,342
Asynchronous Database Reads	2.2915E9
Page Faults	47,659,965
Information about SQL stateme...	
Statement Number	123,704
Statement Function	Insert
Statement Operation	Open
Statement Type	Static
Statement Name	
Statement Outcome	Successful
Additional Table Info	
Total Rows in Table	1,544,582,493
Table Size(bytes)	389,246,666,732
Active Table Rows	1,544,582,493
Deleted Table Rows	0.0
Host Variable Values	793491, '2010-10-31'
Additional information about SQ...	
CLOSQLCSR Value	
ALWCPYDTA Value	Any Time
Pseudo Open	No
Pseudo Close	No
Hard Close Reason Code	Not Available

```
INSERT INTO [redacted]
FROM [redacted] WHERE P411BILL2 = : H AND DATE ( P411STAMP ) = : H WITH NC
```

Statement text

CREATE INDEX DTALIB.TABLENAME_IDX ON DTALIB.TABLENAME (P411BILL2 ASC, DATE(P411STAMP) as p411Date);

Replace a high use Index scan with a Index probe



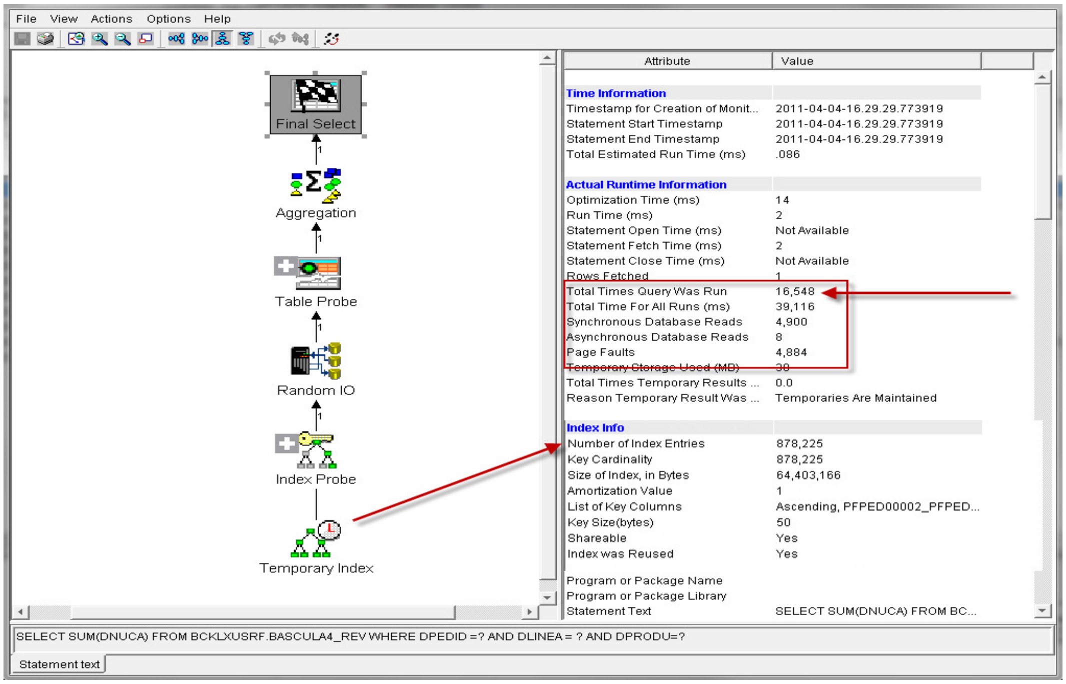
Attribute	Value
Time Information	
Timestamp for Creation of Monit...	2010-08-04-23.42.:
Statement Start Timestamp	2010-08-04-23.42.:
Statement End Timestamp	2010-08-04-23.42.:
Total Estimated Run Time (ms)	72.368
Actual Runtime Information	
Optimization Time (ms)	473
Run Time (ms)	112
Statement Open Time (ms)	Not Available
Statement Fetch Time (ms)	112
Statement Close Time (ms)	Not Available
Rows Fetched	1
Total Times Query Was Run	3,636,984
Total Time For All Runs (ms)	4.1078E8
Synchronous Database Reads	406
Asynchronous Database Reads	2,529
Page Faults	430
	UP
Program or Package Library	MEXEKACEN
Statement Text	SELECT FCEMPNU
Host Variable Values	649159

178,227 keys Scanned 3,636,984 Times = 648,208,747,368 keys processed

```
SELECT FCEMPNUM , FIDEPTOID INTO : H , : H FROM [redacted]/ NUMEMPLEM WHERE EMPNUM = : H
```

Pass matching Host variables, or alter the column to be an integer.

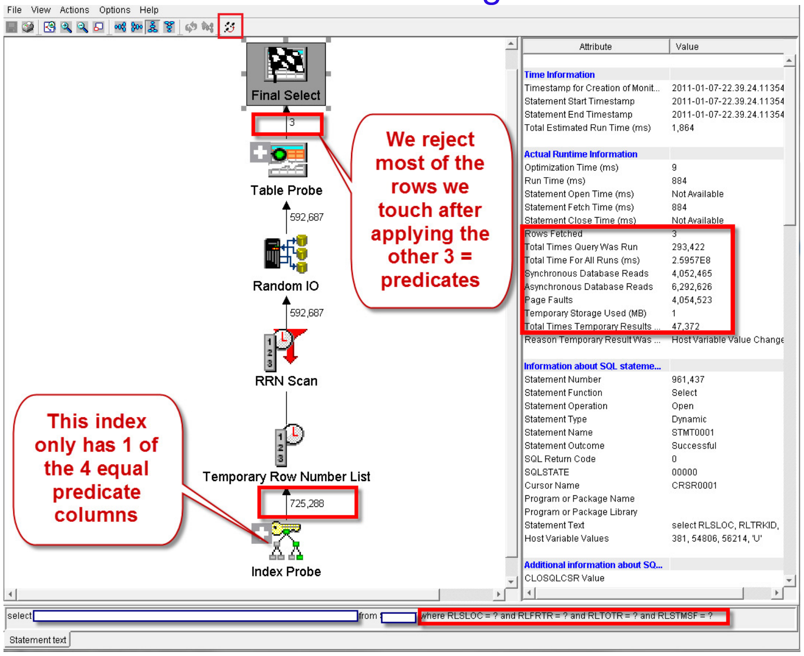
Replace high use temporary index with permanent **SQL INDEX**



Attribute	Value
Time Information	
Timestamp for Creation of Monit...	2011-04-04-16.29.29.773919
Statement Start Timestamp	2011-04-04-16.29.29.773919
Statement End Timestamp	2011-04-04-16.29.29.773919
Total Estimated Run Time (ms)	.086
Actual Runtime Information	
Optimization Time (ms)	14
Run Time (ms)	2
Statement Open Time (ms)	Not Available
Statement Fetch Time (ms)	2
Statement Close Time (ms)	Not Available
Rows Fetched	1
Total Times Query Was Run	16,548
Total Time For All Runs (ms)	39,116
Synchronous Database Reads	4,900
Asynchronous Database Reads	8
Page Faults	4,884
Temporary Storage Used (MB)	30
Total Times Temporary Results ...	0.0
Reason Temporary Result Was ...	Temporaries Are Maintained
Index Info	
Number of Index Entries	878,225
Key Cardinality	878,225
Size of Index, in Bytes	64,403,166
Amortization Value	1
List of Key Columns	Ascending, PFPED00002_PFPED...
Key Size(bytes)	50
Shareable	Yes
Index was Reused	Yes
Program or Package Name	
Program or Package Library	
Statement Text	SELECT SUM(DNUCA) FROM BC...

```
SELECT SUM(DNUCA) FROM BCKLXUSRF.BASCULA4_REV WHERE DPEDID =? AND DLINEA = ? AND DPRODU=?
```

Create a better index – Avoids touching the table



We reject most of the rows we touch after applying the other 3 = predicates

This index only has 1 of the 4 equal predicate columns

Attribute	Value
Time Information	
Timestamp for Creation of Monit...	2011-01-07-22.39.24.11354
Statement Start Timestamp	2011-01-07-22.39.24.11354
Statement End Timestamp	2011-01-07-22.39.24.11354
Total Estimated Run Time (ms)	1,864
Actual Runtime Information	
Optimization Time (ms)	9
Run Time (ms)	884
Statement Open Time (ms)	Not Available
Statement Fetch Time (ms)	884
Statement Close Time (ms)	Not Available
Rows Fetched	3
Total Times Query Was Run	293,422
Total Time For All Runs (ms)	2.5957E8
Synchronous Database Reads	4,052,465
Asynchronous Database Reads	6,292,626
Page Faults	4,054,523
Temporary Storage Used (MB)	1
Total Times Temporary Results ...	47,372
Reason Temporary Result Was ...	Host Variable Value Change
Information about SQL stateme...	
Statement Number	961,437
Statement Function	Select
Statement Operation	Open
Statement Type	Dynamic
Statement Name	STMT0001
Statement Outcome	Successful
SQL Return Code	0
SQLSTATE	00000
Cursor Name	CRSR0001
Program or Package Name	
Program or Package Library	
Statement Text	select RLSLOC, RLTRKID,
Host Variable Values	381, 54806, 56214, 'U'
Additional information about SQ...	
CLOSQLCSR Value	

```
select                    from       where RLSLOC = ? and RLFRTR = ? and RLTOTR = ? and RLSTMSF = ?
```

Statement text

CREATE INDEX DTALIB.TABLENAM_IDX ON DTALIB.TABLENAME (RLSTMSF ASC, RLSLOC ASC, RLFRTR ASC, RLTOTR ASC);

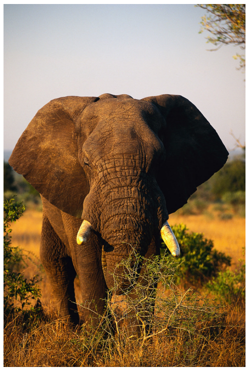
Identify SQL running in CQE

- **CQE Only Monitor**

  STRDBMON OUTFILE(LIB/MonFile) OUTMBR(*FIRST *ADD) JOB(*ALL) TYPE(*DETAIL) COMMENT('**WANT_CQE_ONLY**')

- **Summarize CQE Reasons**
  - See appendix for query to run against the DB monitor
- **Move them to SQE**
  - 6.1 PTF to support SQL that references simple LFs
  - Migrate to i 7.1 – All SQL goes to SQE
  - Replace non- SQL queries with SQL

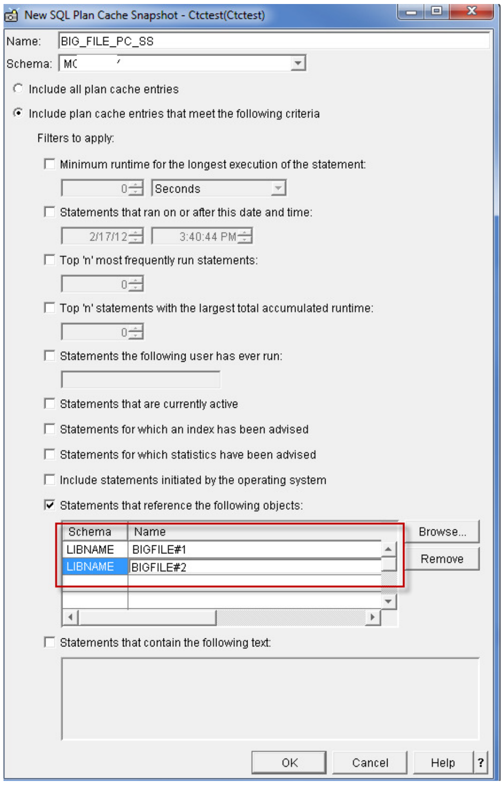
Profile the data – Look for BIG tables



- **How big are your tables ?**
  - The bigger they are the more important it may be to pay attention
  - Use Qsys2 views to understand your biggest files. Or Files with largest numbers of deleted rows

```
Select Table_schema, Table_name, Table_partition, Number_rows,
Number_Deleted_ROWS, DATA_SIZE,
Variable_length_size,  Number_Distinct_Indexes
FROM QSYS2.SYSPARTITIONSTAT
order by DATA_SIZE DESC
FETCH FIRST 100 rows only;
```

Look to see how the big files are used in SQE SQL Plan Cache



New SQL Plan Cache Snapshot - Ctctest(Ctctest)

Name: BIG_FILE_PC_SS

Schema: MC

○ Include all plan cache entries

● Include plan cache entries that meet the following criteria

Filters to apply:

□ Minimum runtime for the longest execution of the statement: 0 Seconds

□ Statements that ran on or after this date and time: 2/17/12 3:40:44 PM

□ Top 'n' most frequently run statements: 0

□ Top 'n' statements with the largest total accumulated runtime: 0

□ Statements the following user has ever run:

□ Statements that are currently active

□ Statements for which an index has been advised

□ Statements for which statistics have been advised

□ Include statements initiated by the operating system

☑ Statements that reference the following objects:

Schema	Name
LIBNAME	BIGFILE#1
LIBNAME	BIGFILE#2

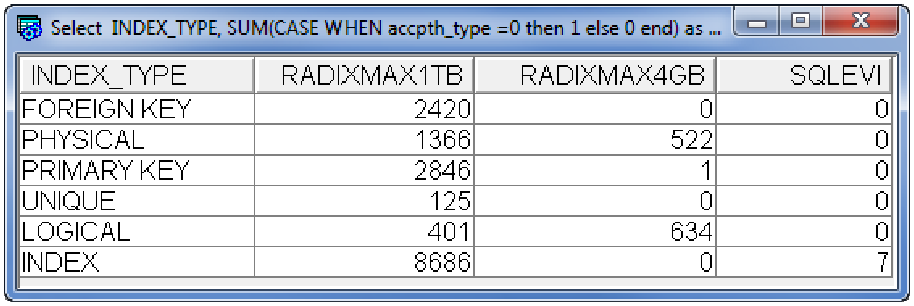
□ Statements that contain the following text:

Look for

1. **Table scans**
2. **Index scans**
3. **Index improvements**
4. **Temp index builds**

Profile the data – What kind of indexes do I have

- **What kinds of indexes do I have?**
  - DDS versus SQL index
  - Page Size or max index size

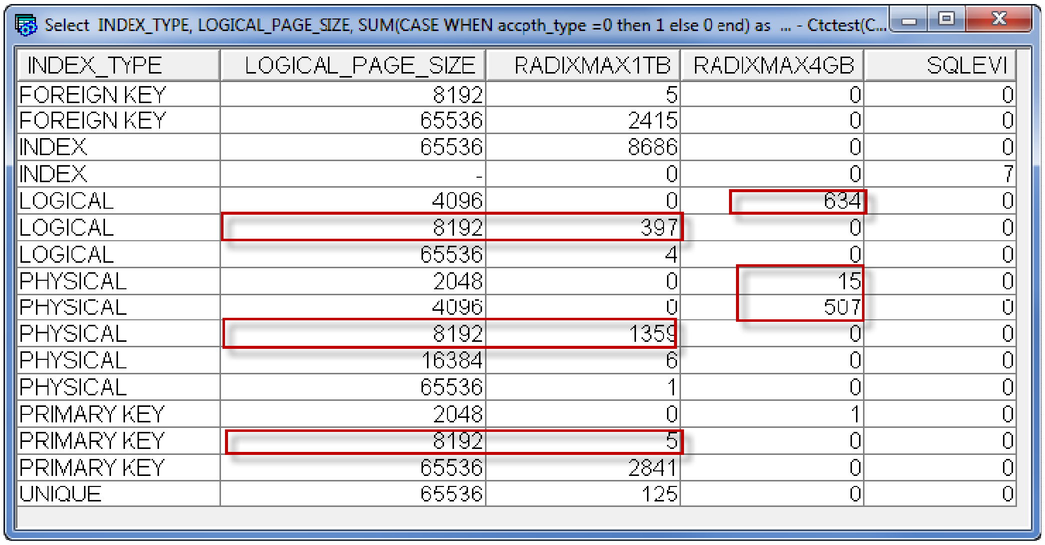


Select INDEX_TYPE, SUM(CASE WHEN accpth_type =0 then 1 else 0 end) as ...

INDEX_TYPE	RADIXMAX1TB	RADIXMAX4GB	SQLEVI
FOREIGN KEY	2420	0	0
PHYSICAL	1366	522	0
PRIMARY KEY	2846	1	0
UNIQUE	125	0	0
LOGICAL	401	634	0
INDEX	8686	0	7

* See Appendix for Index Profile queries

Profile the data – What kind of indexes do I have



Select INDEX_TYPE, LOGICAL_PAGE_SIZE, SUM(CASE WHEN accpth_type =0 then 1 else 0 end) as ... - Ctctest(C...

INDEX_TYPE	LOGICAL_PAGE_SIZE	RADIXMAX1TB	RADIXMAX4GB	SQLEVI
FOREIGN KEY	8192	5	0	0
FOREIGN KEY	65536	2415	0	0
INDEX	65536	8686	0	0
INDEX	-	0	0	7
LOGICAL	4096	0	634	0
LOGICAL	8192	397	0	0
LOGICAL	65536	4	0	0
PHYSICAL	2048	0	15	0
PHYSICAL	4096	0	507	0
PHYSICAL	8192	1359	0	0
PHYSICAL	16384	6	0	0
PHYSICAL	65536	1	0	0
PRIMARY KEY	2048	0	1	0
PRIMARY KEY	8192	5	0	0
PRIMARY KEY	65536	2841	0	0
UNIQUE	65536	125	0	0

* See Appendix for Index Profile queries

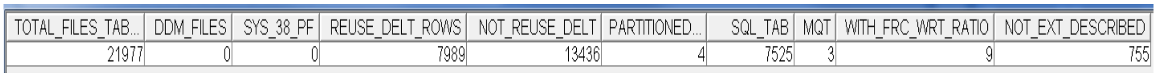
What tools should I use to look at the files

- **Use the following to get a file profile**

```
DSPFD FILE(*ALLUSR/*ALL) TYPE(*ATR) OUTPUT(*OUTFILE) FILEATR(*PF)
OUTFILE(<yourlib>/DSPFD_ATR)
```

```
Select count(*) as Total_Files_Tables, Sum( case WHEN PHFTYP='R' then 1 ELSE 0 end) as
DDM_files, SUM(CASE WHEN PHFATR ='PF38' then 1 else 0 end) as Sys_38_PF, -- Reuse
Setting SUM(CASE WHEN PHRUSE ='Y' then 1 else 0 end) as Reuse_Delt_Rows,
SUM(CASE WHEN PHRUSE ='N' then 1 else 0 end) as NOT_Reuse_Delt, -- Partitioned tables
sum(case when PHPRBY <>0 then 1 else 0 end) as Partitioned_tab, -- SQL Tables,  If not a T
or M, its a PF SUM(CASE WHEN PHSQLT ='T' then 1 else 0 end) as SQL_Tab,  -- MQTs
SUM(CASE WHEN PHSQLT ='M' then 1 else 0 end) as MQT, SUM(CASE when PHFRCR<>0
then 1 else 0 end) as With_Frc_WRT_RATIO, -- Program Described sum(case when
PHFLS='N' then 1 else 0 end) as NOT_Ext_Described

from <yourlib>.dspfd_ATR ;
```



TOTAL_FILES_TAB...	DDM_FILES	SYS_38_PF	REUSE_DELT_ROWS	NOT_REUSE_DELT	PARTITIONED...	SQL_TAB	MQT	WITH_FRC_WRT_RATIO	NOT_EXT_DESCRIBED
21977	0	0	7989	13436	4	7525	3	9	755

Look for

- **High number of deleted rows**
- **Max 4GB Indexes**
- **High use indexes with small page (8K) size**
- **Indexes that are not used**
  - How do I find these?
    - iNav show indexes OR
    - QUSRMBRD API,
      - **Data space index last query statistics use.** The date and time the last time this data space index was used for statistics when optimizing a query. This is in the YYYYMMDDHHMMSS format.
      - **Data space index last query use.** The date and time the last time this data space index was used in a query full open. This is in the YYYYMMDDHHMMSS format.

Interview the application developers

- **What data are you using?**
- **How do you access it?**
  - One row at a time?
  - Set based processing
  - Do you make copies of the data?
- **How much data do you process?**
  - What are the most expensive components
- **Have you done performance testing on production-like data?**
  - Size and content needs to be comparable
  - Environment is key as well
  - What analysis did you do on that test?

10 DB2 for i Performance Tips

What should I do about it – Tip #1 - **Improve indexing Strategy**

- **Minimize accessing data that you don't need to**
  - More Selection columns in the index
  - Utilize Index Only Access when possible
- **Utilize new indexing technology in 6.1 and 7.1**
  - **Derived indexes**
  - EVI Aggregates
  - Memory resident indexes
  - Indexes on SSDs (tables too)
- **Get rid of MAX 4G indexes**
- **High use indexes should have bigger page size**

Read the newly updated Indexing white paper

https://www-304.ibm.com/partnerworld/wps/servlet/ContentHandler/stg_ast_sys_wp_db2_i_indexing_methods_strategies

What should I do about it – Tip #2 - **Identify critical, inefficient jobs, applications and programs. Make those more efficient!!**

- **Move from Record at a time to Set based SQL processing**
  - Minimize the data you access in your programs
  - Rewrite to use SQL set at a time (don't do SQL record at a time)
    - Focus on the expensive, not everything!!
  - Potentially use ***RPG Open Access*** to intercept native I/O and do SQL set processing in the handler.
    - http://www.ibm.com/developerworks/ibmi/library/i-roaforsql/index.html?ca=drs-

What should I do about it – Tip #2 CONT **Native Record Level Access to SQL Set based processing**

- **Take more of a Data Centric Approach versus applications centric**
  - get the database management system to do more on your behalf

- **Drive as much work down into the DBMS as possible**

  <u>With traditional record level access:</u>

  You tell DB2 what you want, <u>and</u> how to do it.

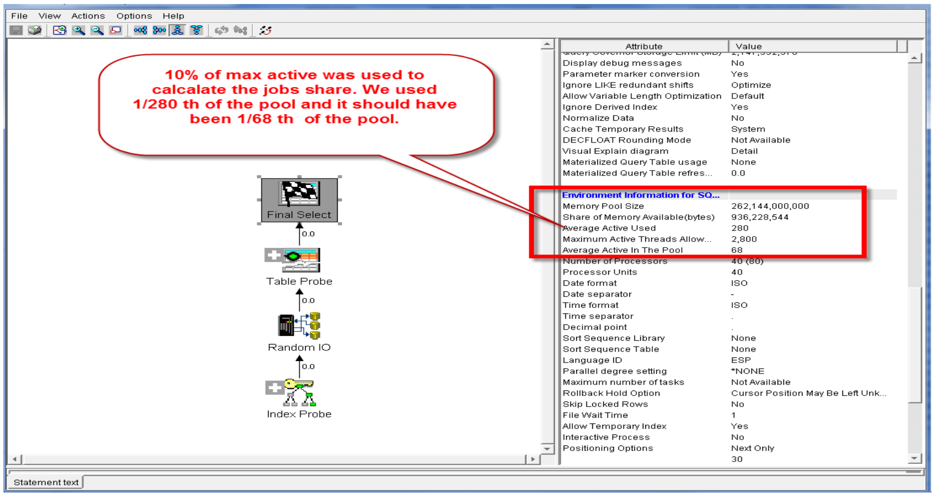
  <u>With SQL:</u>

  You tell DB2 what you want, <u>but</u> not how to do it.

- **Key attributes of Data centric Approach and set based SQL**
  - More sophisticated access
  - More Concurrent I/O
  - More Easily multithreaded
  - Centralized data integrity

What should I do about it – Tip #3 **Ensure that Jobs Memory share is not too Small**

- Optimizer calculates Jobs fair share and considers Max Active job setting in the pool. Smallest active number of jobs that will be used to calculate the Share is 10% of max
- Avoid setting Max active higher than it needs to be



10% of max active was used to calcalate the jobs share. We used 1/280 th of the pool and it should have been 1/68 th of the pool.

Attribute	Value
Query Governor Storage Limit (MB)	2,147,352,578
Display debug messages	No
Parameter marker conversion	Yes
Ignore LIKE redundant shifts	Optimize
Allow Variable Length Optimization	Default
Ignore Derived Index	Yes
Normalize Data	No
Cache Temporary Results	System
DECFLOAT Rounding Mode	Not Available
Visual Explain diagram	Detail
Materialized Query Table usage	None
Materialized Query Table refres...	0.0
Environment Information for SQ...	
Memory Pool Size	262,144,000,000
Share of Memory Available(bytes)	936,228,544
Average Active Used	280
Maximum Active Threads Allow...	2,800
Average Active In The Pool	68
Number of Processors	40 (80)
Processor Units	40
Date format	ISO
Date separator	-
Time format	ISO
Time separator	.
Decimal point	.
Sort Sequence Library	None
Sort Sequence Table	None
Language ID	ESP
Parallel degree setting	*NONE
Maximum number of tasks	Not Available
Rollback Hold Option	Cursor Position May Be Left Unk...
Skip Locked Rows	No
File Wait Time	1
Allow Temporary Index	Yes
Interactive Process	No
Positioning Options	Next Only
	30

What should I do about it – Tip #4 **Set Optimization Goal to match behavior**

- Set via optional SQL statement clause
  - –OPTIMIZE FOR n ROWS
  - –OPTIMIZE FOR ALL ROWS
- Set via QAQQINI options file -or- ODBC/JDBC connection attributes
  - –*FIRSTIO
  - –*ALLIO
  - –(chgqrya qryoptlib(<ini lib>))
- Default for dynamic interfaces is **First I/O**
  - –ODBC, JDBC, STRSQL, run SQL scripts (iNav), dynamic SQL in programs
  - –**CQE - 3% of expected result set**
  - –**SQE - 30 rows**
- Otherwise default is **ALL I/O**
  - –Extended dynamic, RUNSQLSTM, INSERT + subSELECT, CLI, static SQL in programs
  - –**All of expected excepted result set**
- Optimization goal will affect the optimizer's decisions
  - –Use of indexes, SMP, temporary intermediate results like hash tables
  - –**Tell the optimizer as much information as possible**
  - –**If the application fetches the entire result set, use *ALLIO**

What should I do about it – Tip #5 **Reorganize high use files with lots of deleted rows**

- **We can reduce disk space, main store foot print, CPU and disk ops by compressing out deleted rows**
  - – Rule of thumb says > 10% should be reorganized ??
- **Look for the really bad ones first**

-- Top 25 tables that Have > 10,000,000 deleted rows

Select Table_schema, Table_name, Table_partition, Number_rows, Number_Deleted_ROWS, DATA_SIZE, Variable_length_size, Number_Distinct_Indexes

FROM QSYS2.SYSPARTITIONSTAT WHERE NUMBER_DELETED_ROWS > 10000000 order by Number_Deleted_ROWS DESC FETCH FIRST 25 rows only;

- Use Reorganize to compress out deleted rows

RGZPFM FILE(FLGHT400/ORDERS) MBR(ORDERS) KEYFILE(*RPLDLTRCD) ALWCANCEL(*YES) LOCK(*SHRUPD) ; – This example allows concurrent updates and replaces deleted records at the start of the file with valid records from the end of the file.

What should I do about it – Tip #6 **Control adhoc queries**

- **Control significant resource usage**
  - Ad hoc queries
    - Limit access to certain interfaces
    - Use Query Governor (see the QAQQINI file)
  - Look for users that are doing expensive queries
    - Should they be doing that off peak?
    - Should they be doing it at all??

What should I do about it – Tip #7 **Don’t ask for more data that is needed**

- **Limit result set to something reasonable**
  - Ask yourself, will a end user really scroll through millions of rows in a report?
  - Use FETCH N rows
  - Search for SQL Pagination
    - http://www.itjungle.com/fhg/fhg111109-story01.html

What should I do about it – Tip #8 **Avoid making copies of data**

- **Replace creation of work files or temp table with a virtual table**
  - Optimizer is often given <u>no implementation tools</u> with these temp tables AND the plans cannot be shared.
  - Create a view that defines that data set and access the view instead of the work file (if reuse of view is likely in other queries and no hostvar requirements)
  - Otherwise, Use **<u>Common Table Expression</u>** or derived table
    - See CTE examples in the appendix

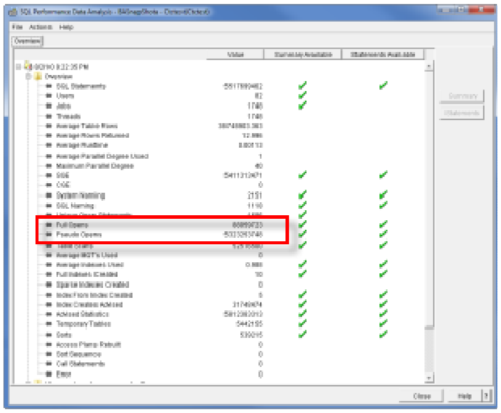
What should I do about it – Tip #9 **Use separate environment for Query and reporting**

- **Mixing core applications with analytical type queries/reports can cause problems**
  - Query and reporting may be resource intensive
  - Needs different set of indexes than OLTP apps
  - May need different DB design (OLAP vs OLTP)
- **Move the query and reporting to separate system or LPAR**
  - Maybe the HA target
  - Separate box with an Operational Data Store
    - Appliance like i for BI Optimized Solution
      - http://www-03.ibm.com/systems/resources/systems_i_software_db2webquery_whyibmiforbi.pdf

What should I do about it – Tip # 10 Reduce Full opens

- **First determine if it's a problem**
  - Analyze plan cache snapshot or DB monitor
  - Use System Tools
- **Common Causes**
  - Frequently getting new connection from remote app
  - In RPG setting LR on exit
  - Programs compiled incorrectly

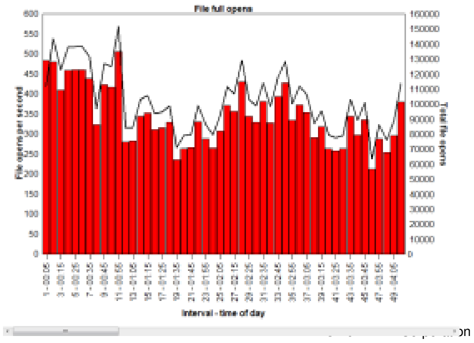
See Appendix for more info



8/9/2010 8:32:35 PM

Overview

	Value	Summary Available	Statements Available
SQL Statements	5517890482	✓	✓
Users	82	✓	
Jobs	1748	✓	
Threads	1748		
Average Table Rows	387488803.383		
Average Rows Returned	12.886		
Average Runtime	0.00113		
Average Parallel Degree Used	1		
Maximum Parallel Degree	40		
SQE	5411312471	✓	✓
CQE	0		
System Naming	2151	✓	✓
SQL Naming	1110	✓	✓
Unique Query Statements	[illegible]	✓	
Full Opens	668890723	✓	✓
Pseudo Opens	5033283748	✓	✓
Table Scans	[illegible]	✓	✓
Average MQT's Used	0		
Average Indexes Used	0.888	✓	✓
Full Indexes Created	10	✓	✓
Sparse Indexes Created	0		
Index From Index Created	0	✓	✓
Index Created Advised	31749474	✓	✓
Advised Statistics	5812382012	✓	✓
Temporary Tables	5442155	✓	✓
Sorts	539015	✓	✓
Access Plans Rebuilt	0		
Sort Sequence	0		
Call Statements	0		
Errors	0		



Miscellaneous Tips

- **For Varchar columns set the allocated length so it covers most occurrences**
- **Utilize the DB2 for i Grouping sets support**
  - Allows grouping at many levels/dimensions within one query
- **Avoid use of Force write ratio**
- **Rebuild EVIs with high over flow count**
- **Utilize Tuned Views – DB engineer creates, tests and tunes before making them available**
- **Improve DB design (3rd normal form recommended)**
- **Use Referential integrity – Avoid data integrity checking**
- **Ensure you have a balanced configuration**
  - Avoid bottleneck on CPU, Memory Disk (do analysis and tuning before adding HDW)
- **You can't cheat the SQE Stats collector – leave it on!**
- **Use Performance Explorer to identify** expensive programs **and focus attention on those – histogram of the expensive calls**
- **Utilize SMP to take advantage of CPU power**
  - Do proper tuning first
  - Careful about SMP scope (do NOT set it system wide unless its a dedicated query box)
  - Utilize % capability in the QAQQINI file (*OPTIMIZE NN – Set degree to NN % of what the optimizer calculates) to throttle back SMP

Miscellaneous Tips –continued

- **Be aware of need (or NOT) for Live Data Mode or Allow Copy Data *NO**
  - This restricts optimizer choices
- **There are three possible options for cursor sensitivity. The default on a create cursor is ASENSITIVE (the most flexible) but be aware while debugging or developing an application that different environments may have this set to other values.**
  - ASENSITIVE - Specifies that the cursor may behave as SENSITIVE or INSENSITIVE depending on how the select-statement is optimized (optimizers choice!)
  - INSENSITIVE -Specifies that once the cursor is opened, it does not have sensitivity to inserts, updates, or deletes performed by this or any other activation group (tends towards plan with more cached results in the query)
  - SENSITIVE – Specifies that changes made to the database after the cursor is opened are visible in the result table (results in a 'live data' mode access plan, focuses on indexes and direct table access methods)
- **Beware of the DEFAULTS for Creating UDFs (and UDTFs)!**
  Although defaults on UDF are the 'safe' **choice, they are NOT the most performance friendly!**
  - NOT DETERMINISTIC - function might not return the same result each time that the function is invoked with the same input arguments (prevents caching)
  - EXTERNAL ACTION - function can take an action that changes the state of an object that the database manager does not manage (prevents caching)
  - FENCED – Function runs in a separate thread (additional overhead for thread)
  - READS SQL DATA – function can execute SQL but not statements that modify SQL
  - **IF** the nature of the UDF permits, the following options can result in a more optimal performing access plan: DETERMINISTIC, NO EXTERNAL ACTION, UNFENCED, CONTAINS SQL
  - Ensure that UDFs are NOT FENCED unless they need to run on a different thread

Miscellaneous tips -continued

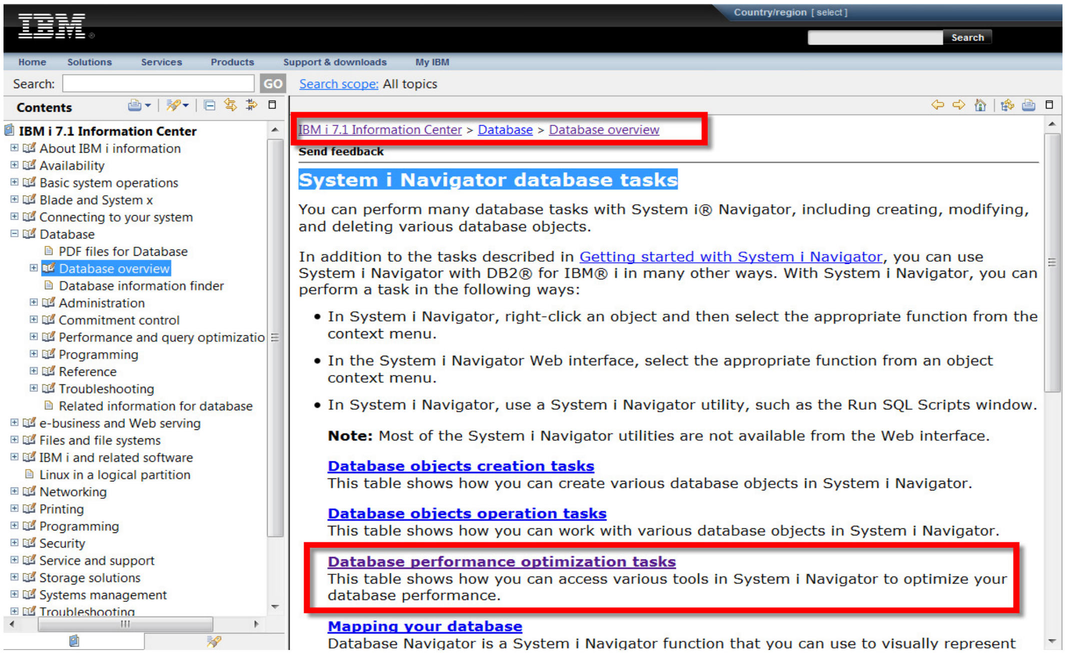
- **Inline UDFs (available in PTF)**
  - In cases of very simple SQL functions, instead of invoking the function as part of a query, the expression in the RETURN statement of the function will be in-lined to the query. To take advantage an existing UDF needs to be altered or recreated with enablement PTFs.
    - A function is an inline function IF:
      - It is deterministic
      - SQL-routine-body contains only a RETURN statement.
      - RETURN statement does not contain a scalar subselect or fullselect.
      - SQL-routine-body does not contain BEGIN ATOMIC.

      - Example Tip. If you do an if/then/else with a RETURN on each leg, it cannot be inlined…but …if you return a CASE statement .. it can …

Additional Education

- IBM i Database Performance and Query Optimization Guide (infocenter pub)
- SQL Performance tuning workshop
  - http://www-03.ibm.com/systems/i/software/db2/db2performance.html
- DB2 for IBM i Website
  - http://www-03.ibm.com/systems/i/software/db2/
- DB2 Performance tuning Redbook (old)
  - http://www.redbooks.ibm.com/abstracts/sg247326.html?Open
- iDoctor
  - https://www-912.ibm.com/I_dir/idoctor.nsf
- Navigator for i (browser)
  - http://www-03.ibm.com/systems/i/software/navigator/directornavigator.html

Additional Education - IBM i Navigator database tasks



IBM i 7.1 Information Center > Database > Database overview

Send feedback

System i Navigator database tasks

You can perform many database tasks with System i® Navigator, including creating, modifying, and deleting various database objects.

In addition to the tasks described in Getting started with System i Navigator, you can use System i Navigator with DB2® for IBM® i in many other ways. With System i Navigator, you can perform a task in the following ways:

- In System i Navigator, right-click an object and then select the appropriate function from the context menu.
- In the System i Navigator Web interface, select the appropriate function from an object context menu.
- In System i Navigator, use a System i Navigator utility, such as the Run SQL Scripts window.

Note: Most of the System i Navigator utilities are not available from the Web interface.

Database objects creation tasks
This table shows how you can create various database objects in System i Navigator.

Database objects operation tasks
This table shows how you can work with various database objects in System i Navigator.

Database performance optimization tasks
This table shows how you can access various tools in System i Navigator to optimize your database performance.

Mapping your database
Database Navigator is a System i Navigator function that you can use to visually represent

Summary

- **The more data being processed the more important efficiency becomes.**
  - Not just lots of data per request. But large number of small requests as well.
- **Learn to use the tools**
  - Collection Services, Job Watcher and iNavigator
  - Experience is the best teacher
- **Focus tuning effort on what is expensive**
  - low hanging fruit
- **Think Data Centric, not Application Centric**
  - Learn to utilize the Power of SQL - set at a time processing!

OnDemand Performance Center – User Authority Simplification

*JOBCTL (Job Control Authority)

- Whatever worked with *JOBCTL in IBM i 6.1 will continue to work

QIBM_DB_SQLADM – Database Administrator

- This is a database specific alternative to *JOBCTL. It is a superset of the function authorized by QIBM_DB_SYSMON.
- Examples:
  - Change parallel degree for DB2 SMP feature
  - Work with Plan Cache
  - Work with OmniFind Text Search Server

QIBM_DB_SYSMON – Database Information

- This allows a user to view some system level details, but not specifics about operations or anything related to changing or controlling the database.
- Examples:
  - QUSRJOBI for SQL information
  - Show SQL Information for Jobs

User Authorization Commands:

```
CHGFCNUSG FCNID(QIBM_DB_SQLADM)
    USER(userid) USAGE(*ALLOWED)

CHGFCNUSG FCNID(QIBM_DB_SYSMON)
    USER(userid) USAGE(*ALLOWED)
```

No Special Authority required when using OnDemand Performance Center with own job

- Starting/ending SQL Performance Monitors on your own job
- Analysis of SQL Monitor/Plan cache data
- Visual Explain in Run SQL Scripts

Questions ?

Appendix

Index Profile queries

```
Select  INDEX_TYPE, SUM(CASE WHEN accpth_type =0 then 1 else 0 end) as
   RADIXMAX1TB,
         SUM(CASE WHEN accpth_type = 1 then 1 else 0 end) as RADIXMAX4GB,
   SUM(CASE WHEN ACCPTH_TYPE=2 then 1 else 0 end) as SQLEVI

   FROM qsys2.syspartitionindexstat
   where table_NAME not like 'Q%' and table_SCHEMA NOT LIKE 'Q%'
   group by index_TYPE;

Select  INDEX_TYPE, LOGICAL_PAGE_SIZE, SUM(CASE WHEN accpth_type =0
   then 1 else 0 end) as RADIXMAX1TB,
         SUM(CASE WHEN accpth_type = 1 then 1 else 0 end) as RADIXMAX4GB,
   SUM(CASE WHEN ACCPTH_TYPE=2 then 1 else 0 end) as SQLEVI

   FROM qsys2.syspartitionindexstat
   where table_NAME not like 'Q%' and table_SCHEMA NOT LIKE 'Q%'
   group by index_TYPE, LOGICAL_PAGE_SIZE
   order by index_type, logical_page_size;
```

CTE EXAMPLE – Logical Step-by-Step

- **Break query into logical steps when query requires multiple SQL statements**
  - Improved readability
  - Removes SQL view management, host variables also supported
  - Can be used to avoid **temp** files

```
WITH staff (deptno, empcount) AS
(SELECT deptno, COUNT(*) FROM employee
 WHERE division = :div_var GROUP BY deptno)
SELECT deptno, empcount FROM staff
WHERE empcount =
        (SELECT MAX(empcount) FROM staff)
```

CTE EXAMPLE – Logical Step-by-Step & Work Tables

- **CTE logical step processing can <u>eliminate the need</u> for temporary work tables**

```
DECLARE GLOBAL TEMPORARY TABLE t1 AS
(SELECT shipdate, customer, phone, orderkey, linenumber
  FROM item_fact i, cust_dim c
  WHERE c.custkey=i.custkey AND discount=0.08) WITH DATA;

DECLARE GLOBAL TEMPORARY TABLE t1 AS
(SELECT customer, phone, orderkey, linenumber, year, quarter
  FROM t1, star1g.time_dim t
  WHERE t.datekey=shipdate ) WITH DATA;

...
```

CTE EXAMPLE – Logical Step-by-Step & Work Tables (Continued)

- **CTE provides the same step-by-step approach without the overhead of populating physical tables**

```
WITH t1 AS
 (SELECT shipdate, customer, phone, orderkey, linenumber
  FROM item_fact i, cust_dim c
  WHERE c.custkey = i.custkey AND discount=0.08),
     t2 AS
  (SELECT customer, phone, orderkey, linenumber, year, quarter
   FROM t1, star1g.time_dim t
   WHERE t.datekey = shipdate)
SELECT * FROM t2;
```

Finding EVIs with large over flow

```
Select Index_schema, Index_name, Column_names, Number_keys,
   Last_query_use, Query_use_count, Query_Statistics_count, index_size,
   Overflow_values, EVI_CODE_SIZE
   FROM QSYS2.syspartitionindexstat
   where ACCPTH_TYPE=2 and overflow_values > 0 and table_NAME not like
   'Q%' and table_SCHEMA NOT LIKE 'Q%'
   order by Overflow_values desc
```

Refresh the symbol table using CHGLF

```
CHGLF FILE(LIBNAME/INDEXNAME) FRCRBDAP(*YES)
```

More Info on Tip #1 – Find and replace max 4GB indexes

Replace MAX 4G indexes with MAX 1TB

```
Select *
from qsys2.syspartitionindexstat
where (table_schema not like 'Q%' or table_schema = 'QUSRSYS') and
ACCPTH_TYPE=1
Order by INDEX_SIZE DESC;
```

- **For LFs**
  - Option 1 Change these to MAX 1TB **USE THIS OPTION**

    ```
    CHGLF FILE(DTALIB/PF1) ACCPTHSIZ(*MAX1TB)
    ```

  - Option 2 – Recreate Simple Keyed LFs using SQL CREATE INDEX and new 6.1 syntax. You can access these as LFs from Native I/O Commands

- **For PFs**

  ```
  CHGPF FILE(DTALIB/PF1) ACCPTHSIZ(*MAX1TB)
  ```

Tip #2 – Monitor, Analyze and Tune strategy

- **Monitor**
  - Keep archive of historical performance data
    - For a typical Day or week
      - Collection services
      - Plan cache snapshot
      - Health center file information
      - DB monitor sample
    - For Peak periods (end of month, peak season…)
    - Batch window as well (maybe a separate collection of data)
  - If problem occurs capture data and save to allow analysis of what just happened
  - Automate collection process (i.e. schedule job to monitor or collect data)
  - Here is article showing how to automate a Plan cache snapshot collection
    - http://www.mcpressonline.com/tips-techniques/system-administration/techtip-automatically-generate-an-sqe-plan-cache-snapshot.html

Tip #2 – Monitor, Analyze and Tune strategy…..

- **Analyze**
  - Periodically analyze top N most time consuming SQL from Plan cache (Save those to a new snapshot and compare to old one to see if anything is slower)
    - Look for Table scans, Index scans, Temp index builds, queries that access more data than they need to. Apply the tips (i.e. copies of data, large result sets…)
    - Are there statements that can be written more efficiently
  - Look for expensive executions (sort descending by longest run time)
    - Should these even be run?
    - Look for Users that are doing expensive queries and investigate
  - Assess the system
    - Are any resources a bottleneck. Don't just look at system CPU %
  - Look for Indexes that are advised a large number of times
    - Find the statement(s) that are causing that advice and assess those

Tip #2 – Monitor, Analyze and Tune strategy…..

- **Tune**
  - Proactive analysis for new or changed applications
    - Part of testing needs to include data collection and analysis
    - **A good indexing strategy is KEY**!!
  - Apply the tuning Tips given earlier
  - Use Visual Explain on the statements that are generating index advise for those IXs that have a high advise count.

Reducing Full Opens

- **Use DBMON analyze program summary to see program with high SQL full Opens**
  - CLOSQLCSR setting needs to be explored for those programs that show all full opens
- **Use iDoctor Pex full open trace to identify programs with high native full opens**
- **In addition the Information given, the SQL Query and performance optimization guide should be used. The following link can be used to access that manual:**
  - http://publib.boulder.ibm.com/infocenter/iseries/v7r1m0/topic/rzajq/rzajq.pdf

Reducing Full Opens

- **Reducing Full opens in native applications**
  - we typically see high rate of full opens and closes is that programs issue SETON LR and then Return or end the program. This has a consequence of closing the open files. In some cases the SETON LR (or *INLR = *ON) can be removed avoiding the close and subsequent open on the next call to the program. You would want to use RETURN (without SETON LR) if you are calling this program multiple times from the same calling program, in order to reduce overhead of opening and closing the files each time. For more info on use of SETON LR See the following link:
    - IBM i 7.1 Information Center > Programming > Programming languages > RPG

Reducing Full Opens

- **Reducing Full Opens in JDBC Access**
  - The key to reducing full opens in the JDBC access is to use **connection pooling**. Based on the data we see, you are not using connection pooling successfully. You need to avoid getting a new connection for each external request. Java has classes that will support the creation and reuse of connections. IBM Info Center has information on JDBC Object pooling, which also includes ConnectionPoolDataSource properties. Go to IBM InfoCenter to see additional info
    - IBM i 7.1 Information Center > Programming > Java > IBM Developer Kit for Java > Database access from Java programs > Accessing your IBM i database with the Java JDBC driver
  - You may also build your own connection pooling. See that topic at the following info center location:
    - IBM i 7.1 Information Center > Programming > Java > IBM Developer Kit for Java > Database access from Java programs > Accessing your IBM i database with the Java JDBC driver > JDBC object pooling

OnDemand Performance Center & Sensitive Data – SECURE columns

- **Prevents sensitive data values from being displayed in DB2 performance tools – Database Monitor & Plan Cache (5.4 & 6.1 PTFs)**
  - Only security officer will be able to see sensitive values, '*SECURE' value presented to *normal* users (… WHERE cardnumber=:hostvar1)
  - User must register sensitive columns with DB2 tooling

- **Registration interface is system stored procedure: SET_COLUMN_ATTRIBUTE**
  - Procedure parameter descriptions

Table_Schema	- System name of a table's schema
Table_Name	- System name of a table
Column_Name	- System column name being secured.
Attribute	- Secure attribute setting for column

    - SECURE NO
    - SECURE YES
  - Example:

```
CALL SYSPROC.SET_COLUMN_ATTRIBUTE
  ('MYLIB1', 'ORDERS', 'CCNBR', 'SECURE YES');
```

Trademarks and Disclaimers

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Cell Broadband Engine and Cell/B.E. are trademarks of Sony Computer Entertainment, Inc., in the United States, other countries, or both and are used under license therefrom.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

The customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM's current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Prices are suggested U.S. list prices and are subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.