



iSeries

SQL Performance Monitoring Tools for DB2

Doug Mack – mackd@us.ibm.com

Agenda

- The impact of SQL/Query workloads
- Analyzing SQL/Query jobs with DB Monitor(s)
- Centerfield Technology's Analysis Tools: Overview
- Q & A

ibm.com/series/db2

- Your portal to everything DB2!
- General Information
- See it Action Demos
- Education Roadmaps
- Technical White Papers

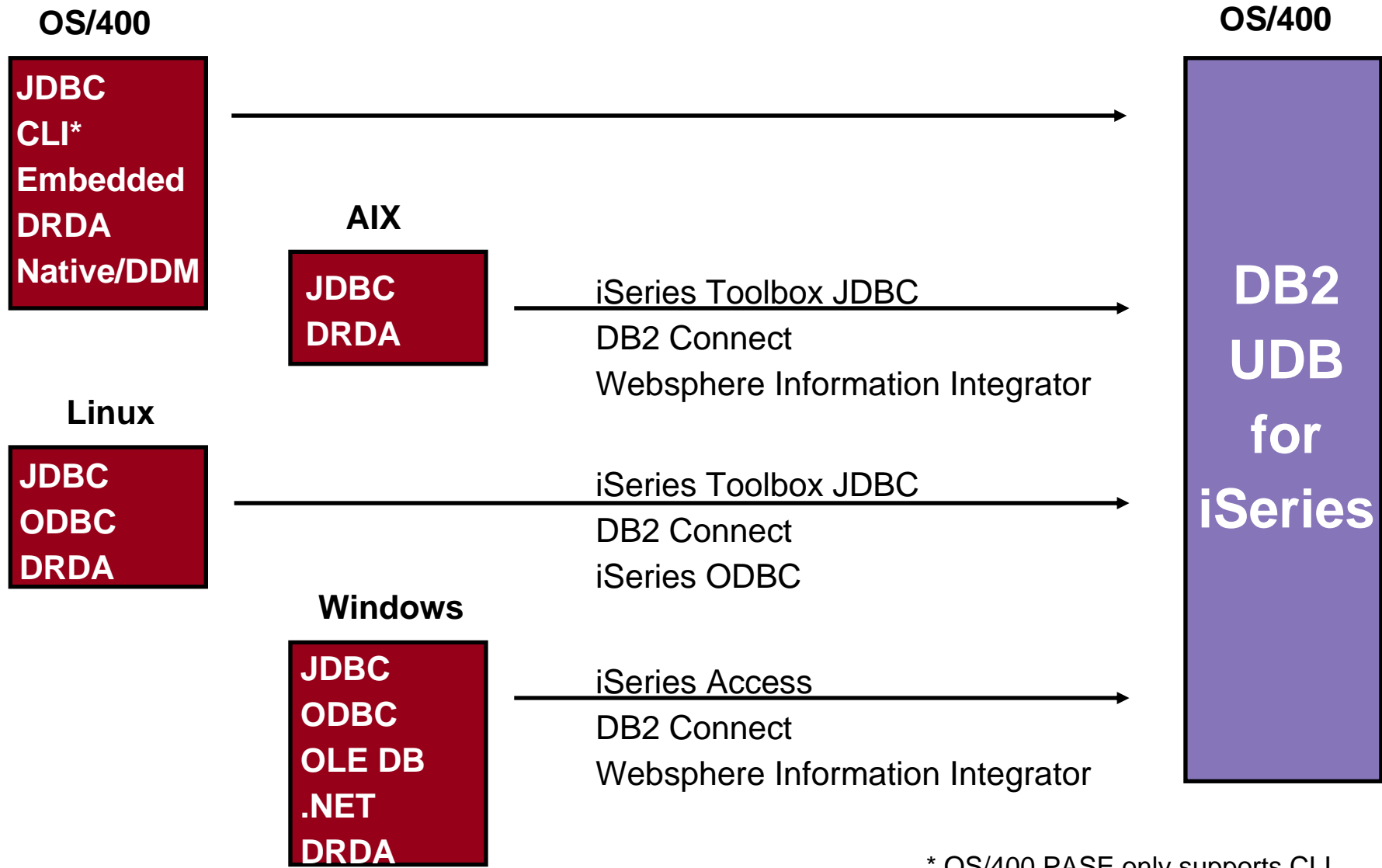
- WHAT'S NEW
 - V5R3 Trifold
 - Technical Papers
 - "Instead Of" Triggers
 - Materialized Query Tables
 - DB2 .NET Plug Ins
 - Redbooks
 - Integrating DB2 Universal Database for iSeries with Microsoft ADO .NET: SG24-6440-00
 - OnLine Education
 - DB Monitor !!

Why SQL?

- **Portability of code & skills**
- **Strategic database interface for industry & OS/400**
 - **Faster performance delivered by SQE only available to SQL-based interfaces**
 - **SQL required for certain functions & middleware**
 - **Data types: BLOB, CLOB, Datalink, ...**
 - **Auto-Incrementing Constructs: Sequence & Identity column attribute**
 - **Column-level Triggers**
 - **Encryption & Decryption functions**
 - **Encoded Vector Indices, ...**
- **Enables better positioning of iSeries as a Database Server**
- **SQL as a programming language can reduce Total LOC**
- **DB2 SMP - parallel database processing**



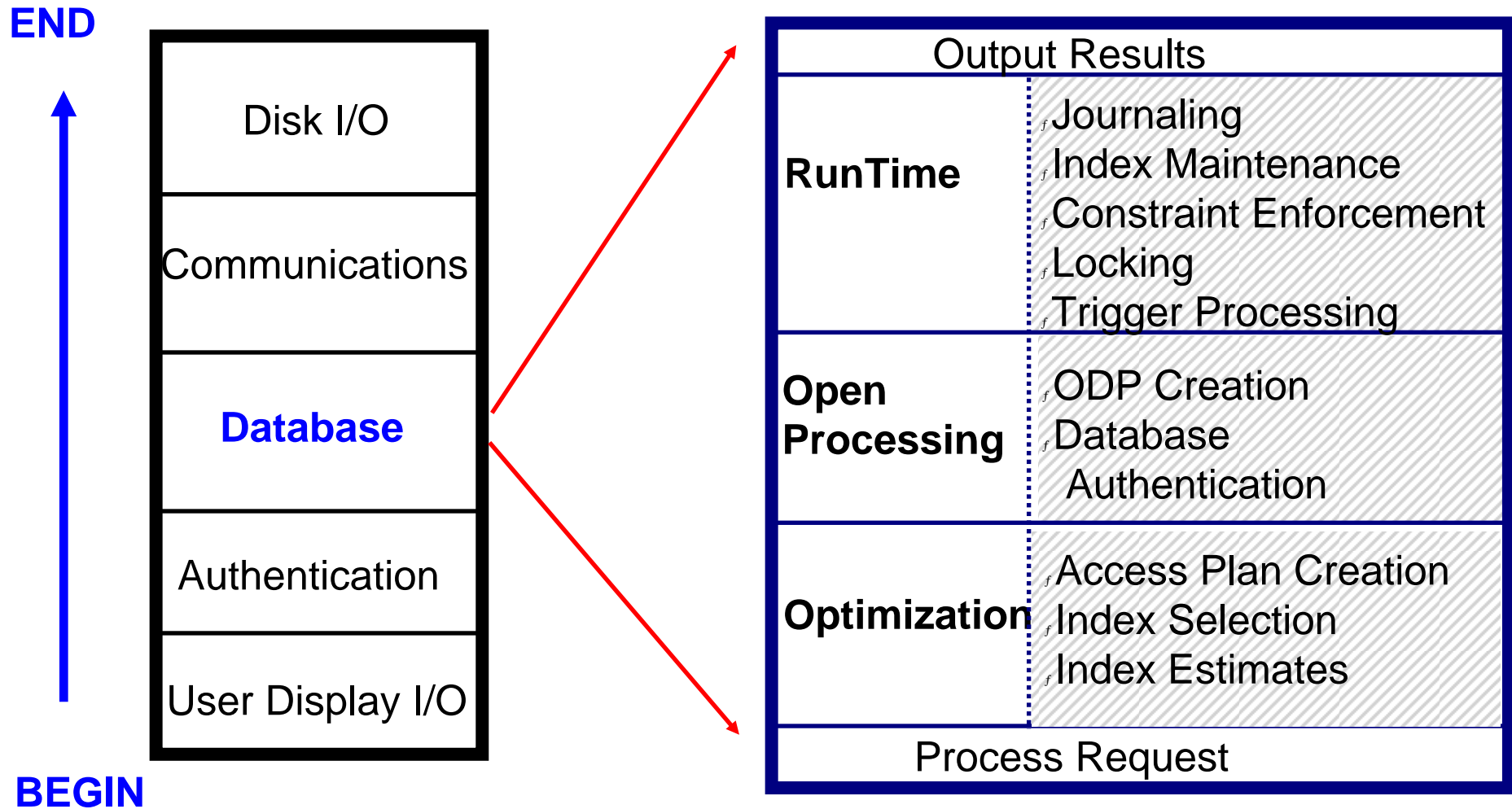
OS/400 + DB2 UDB for iSeries as the "Server"



* OS/400 PASE only supports CLI

Analyzing SQL/Query Performance

- New methodology and tools needed
 - Traditional performance tools do not adequately capture the DB2 “context” of a performance issue
 - Need access to the details of the actual SQL/Query request to analyze if incomplete “Indexing Strategy” is causing DB2 inefficiencies
 - Distributed (SOA?) solutions make it hard to know where SQL/Query requests are running
 - Need ability to look at DB2 performance issues & possible solutions from a system-wide usage perspective
 - Some tools don’t actually describe actions of SQE



SQL Performance Tuning Considerations

- Database server tuning
 - CPU capacity & I/O bandwidth (disk arms!)
 - Expert cache (*CALC memory paging option)
 - Memory pool size & max activity level
 - Number of pre-start jobs
- Client tuning
 - Isolation Level
 - Extended Dynamic (SQL packages)
- SQL Tuning
 - Reusable Open Data Paths (ODPs)
 - SQL Procedural Language (PSM) – minimize nested calls
 - Indexing Strategy (ibm.com/servers/enable/site/education/ibo/record.html?indxng)
 - SQE “Warmup” effect with Automatic Stats Collection
 - Variable Length Columns



SQL Performance Tuning Tools

What are the DB2 Performance Monitors?

- Integrated tools used to gather database performance related statistics for SQL-based requests running on the iSeries
 - Applicable to both batch jobs and interactive work
 - Only collect information on the DB2 engine processing - does not include/time data that it took for the requests to get to the DB2 engine (eg, JDBC driver processing)
- Monitor data dumped into table(s) where it can be queried to help identify and tune performance problem areas
 - Collected data can be analyzed to look at SQL performance **at a global system level, job-level or, specific SQL statement level**
- Monitor data most useful given basic understand of DB2 UDB engine and query optimization techniques

Database Monitor Information

- Provides all information from STRDBG or PRTSQLINF plus additional info such as:
 - System and job name
 - **SQL statement text**
 - Start and end timestamp
 - Estimated processing time
 - Total rows in table queried
 - Number of rows selected
 - Estimated number of rows selected
 - Estimated number of joined rows
 - **Key fields for advised index**
 - Total optimization time
 - **ODP implementation**
 - **QAQQINI settings**
 - Others

Database Monitor Information

- Collected information and stats can be analyzed later to analyze things such as:
 - Which queries are the most time consuming?
 - How many temporary indexes have been created over a particular table?
 - Which user is running the longest running queries?
 - Which queries were implemented using reusable ODPs?
 - How many queries rebuild access plans?
 - Has the implementation of a particular query changed with the application of a PTF or a new release? (Before/After comparison of the monitor data)

What are the DB2 Performance Monitors?

- Two types:
 - Database Performance Monitor (Detailed)
 - "Trace" of SQL requests & non-SQL queries (eg, OPNQRYF)
 - Detailed set of information on optimization & runtime behavior
 - Monitor data dumped to a single output table
 - System performance impact (can be Disk & CPU Intensive)
 - Trace for shorter durations
 - Focused analysis
 - Memory-based Database Performance Monitor (Summary)
 - "Monitoring" of SQL requests
 - Summary of information on optimization and runtime behavior
 - Monitor data dumped to a set of normalized tables
 - Smaller system performance impact (info collected in memory)
 - Monitoring for longer durations
 - Broad analysis (no Visual Explain support)

Monitor Interfaces

- Detailed Monitor
 - STRDBMON & ENDDDBMON CL commands
 - Note: STRDBMON TYPE parameter does **not** control monitor type
 - Info & stats written into output table as soon as DB2 collects the info
- Summary Monitor
 - API-based interface:
 - Start SQL Database Monitor (QQQSSDBM)
 - Clear SQL Database Monitor Statistics (QQQCSDBM)
 - Dump SQL Database Monitor (QQQDSDBM)
 - End SQL Database Monitor (QQQESDBM)
 - Query SQL Database Monitor (QQQQSDBM)
 - Info & stats collected in memory and then later written to output tables
- iSeries Navigator interface via SQL Performance Monitors

Detailed Monitor Row Types

- Row types most often used in analysis (QQRID value)
 - 1000 - SQL summary record
 - 3000 - Arrival sequence
 - 3001 - Using existing index
 - 3002 - Index created
 - 3003 - Query sort
 - 3004 - Temporary results file
 - 3006 - Access plan rebuild info
 - 3007 - Index Evaluation data
 - 3010 - Host variable and ODP implementation
 - 3014 - General Environment info (including QAQQINI settings)
 - 3015 - SQE Statistics Advised info
- Documented in Info Center: [DB2 Database Query and Performance Guide](#)

Global DB Monitor Columns

- Columns common to all row types
 - QQRID: Row Type ID (1000, 3000, 3001, etc)
 - **QQUCNT: Query/Request Identifier - surrogate key**
 - QQJOB: Job name
 - QQUSER: Job user name
 - QVC102: CURRENT job user name **(new with V5R3)**
 - QQJNUM: Job number
 - Job number very useful when multiple jobs collected in one DB monitor table
 - QQTIME: Time that the row was created
 - Can be useful when trying to find out what queries were running in a given time period
 - QQJFLD: Join column, combo of QQJOB,QQUSER,QQJNUM,QQUCNT
 - QQI9: Thread Identifier (may be useful for multi-threaded apps)

Global DB Monitor Columns

select qqjob,qqjnum,qqrid,qqc21,qqucnt,qqi5, qq1000,qqtime from kmtest.daledbmon where qqucnt=1...

QQJOB	QQJNUM	QQRID	QQC21	QQUCNT	QQI5	QQ1000
QZDASOINIT	063224	3000		1118	-	
QZDASOINIT	063224	3014	-	1118	817	
QZDASOINIT	063224	3015	-	1118	-	MERCHANTNUMBER
QZDASOINIT	063224	3010	-	1118	0	10001
QZDASOINIT	063224	1000	OP	1118	0	SELECT * FROM SCN2WCS.ETACCOUNTCFG WHE
QZDASOINIT	063224	1000	DE	0	0	SELECT * FROM SCN2WCS.ETACCOUNTCFG WHE
QZDASOINIT	063224	1000	FE	1118	0	

Linking together rows in the monitor

Column used for different data based on row type

1000 Row - SQL Statement Summary

- Row generated for each SQL operation (Update, Commit, etc)
- Field/Column definition:
 - QQ1000: Prepared text of SQL statement
 - QQC21: Type of SQL operation (OP, FE, CL, UP, IN, DL, ...)
 - 'MT' in this field indicates continuation record for SQL statements that exceed 1000 characters
 - 'FE' is a Fetch summary record, NOT the actual number of fetch operations
 - ODP-related operation types:
 - SI, SK, OP, IN, UP, DL
 - QQI2: Number of rows updated/inserted/deleted
 - QQI3: Number of rows fetched (only on FE rows)
 - Actual number of rows fetched, not fetch attempts
 - QQI6: Elapsed time for this operation in microseconds
 - Time to fetch all rows may not be included in with Open & Select operations, need to look at time on Fetch operation rows

1000 Row - SQL Statement Summary

- Access Plan Information
 - QQC103 & QQC104: Package/Program Name & Library
 - QVC18: Dynamic SQL Statement Type
 - 'E' - Extended Dynamic // 'S' - SystemWideStatement Cache // 'L' - Prepared Stmt
 - QQC22 & QVC22: Access Plan Rebuild Code & Subcode
 - Subcode useful for IBM debug purposes
 - QVC24: Access Plan Save Status
 - 'Ax' values mean access plan could not be saved
 - Blank & 'Bx' values mean access plan was saved successfully
- ODP Information
 - QQI5: Query Instance counter, 0 value means full open occurred
 - QQC15: Hard Close Reason Code (for 'HC' operation type)
 - QVC12: Pseudo Open Indicator
 - QVC13: Pseudo Close Indicator
 - QQC181 & QQC182: Cursor & statement name

Sample Queries

- ibm.com/servers/eserver/series/db2/dbmonqry.htm

Is a single job causing a problem?

- Identify the most time consuming jobs:

```
SELECT SUM(qqi6) "Total Time", COUNT(*) "Total SQL Requests",
       qqjnum,qqjob,qquser FROM ??File Name
WHERE qqrid=1000 AND qqc21 <> 'MT'
GROUP BY qqjob,qquser,qqjnum ORDER BY 1 DESC
```

- Analyze results to determine if a specific job should be targeted for analysis. Use QQJNUM to narrow future monitor data analysis

Total Time	Total SQL Requests	QQJNUM	QQJOB	QQUSER
8206552	9505	063229	QZDASOINIT	QUSER
7687984	10695	063227	QZDASOINIT	QUSER
6342424	11578	063218	QZDASOINIT	QUSER
5901256	10243	063220	QZDASOINIT	QUSER

Maybe a certain SQL Statement is painful ?

- Which SQL statements account for the most run time:

```

SELECT SUM(qqi6) "Total Time" , COUNT(*) "Nbr Times Run", qq1000
FROM ??File Name
WHERE qqjnum=??'xxxxxx' AND qqrid=1000 AND qqucnt<>0
AND qqc21<>'MT'
GROUP BY qq1000 ORDER BY 1 DESC
    
```

Total Time	Nbr Times Run	QQ1000
917728	5	SELECT T1.FIELD1, T1.FLAGS, T1.MINIMUMC
646936	4	SELECT T1.FIELD1, T1.FLAGS, T1.MINIMUMC
322752	2	SELECT T1.FIELD1, T1.FLAGS, T1.MINIMUMC
300840	5	select distinct CATENTRY.CATENTRY_ID from
290184	2	SELECT T1.FIELD1, T1.FLAGS, T1.MINIMUMC
256880	4	select distinct CATENTRY.CATENTRY_ID from
155048	4	SELECT T1.TOTALTAX, T1.TOTALSHIPPING,
152760	15	SELECT T1.PREPAREFLAGS, T1.LASTCREA

- QQUCNT<>0 eliminates non-ODP operations (Prepare, Commit, etc)

- Row types 30xx are referred to as Optimization rows
 - Contain information to construct the access plan for an SQL request
 - Presence of Optimization rows indicate that a Full Open occurred
 - Data Access Method columns contain details on the optimization process
 - QQEPT: Optimizer's estimated processing time
 - QQREST: Estimated number of rows selected
 - QQAJN: Estimated number of rows joined
 - QVPARD: Parallel degree requested
 - QVPARU: Parallel degree used
 - QVP154: Memory Pool Size
 - QVP155: Memory Pool ID

QQRID	QQC21	QQUCNT	QQEPT	QQREST	QQ1000
3000		1118	1	1	-
3014	-	1118	1	1	-
3015	-	1118	-	-	MERCHANTNUMBER
3010	-	1118	-	-	10001
1000	OP	1118	-	-	SELECT * FROM SCN2
1000	DE	0	-	-	SELECT * FROM SCN2
1000	FE	1118	-	-	

3002 Row - Temporary Index Creation

- Index & Table Information
 - QVQTBL & QVQLIB: Table name for which index is built
 - QQRCOD - Reason the index build was done
 - I2 - ordering or grouping
 - I3 - selection and ordering/grouping
 - I4 - nested loop join
 - QQTOTR: Number of rows in table
 - QQRIDX: Number of entries in temporary index
 - QQSTIM: Timestamp for start of index build
 - QQETIM: Timestamp for end of index build
 - QQ1000: Name of columns used for index keys
 - Column names are the "short" column names

3002 Row - Temporary Index Analysis

- Which index builds are done the most often?

```

SELECT qqucnt, ((HOUR(qqetim-qqstim)*3600) +
  (MINUTE(qqetim-qqstim)*60) + SECOND(qqetim-qqstim) +
  (MICROSECOND(qqetim-qqstim)*.000001)) "Index Build Time",
  qqtbl "Table Name", qqtotr "Rows in Table",
  qqridx "Entries in Index", SUBSTR(qq1000, 1, 100) "Key Fields"
FROM ??File Name
WHERE qqrid=3002 AND qqjnum=??'xxxxxx' ORDER BY 5
  
```

QQUCNT	Index Build Time	Table...	Rows in Table	Entries in Index	Key Fields
3075	0.064040	OFFER	6316	6316	CATEN00001 ASCEND, TRADE00001 ASCEND
2164	0.067158	OFFER	6316	6316	CATEN00001 ASCEND, TRADE00001 ASCEND
4192	0.072063	OFFER	6316	6316	CATEN00001 ASCEND, TRADE00001 ASCEND

f Look for repeated index builds or long index builds first

f SUBSTR for QQ1000, but will need to increase them if 100 bytes is not enough

f Need to add (DAY(qqetim-qqstim)*24*3600) if index build spans days

Database: Rchasrnd INDEXES FOR QSYS2.SYSROUTINES

SQL Name	Type	LAST QUERY USE	LAST QUERY STATISTICS USE	QUERY USE COUNT	QUERY STATISTICS USE
QASQRESL	LOGICAL FILE	2004-09-10 11:31:54	2004-09-10 11:31:54	586	746
Q_QSYS2_...	Primary Key Constraint	2004-09-09 15:51:19	2004-09-09 15:51:19	649	649

1 - 2 of 2 objects

Required PTFs:

- V5R3 usage requires iSeries Access V5R3 Fixpak #2 and the following V5R3 Server PTFs: SI12938 & SI12873
- V5R2 usage requires iSeries Access V5R3 Fixpak #3 & V5R2 Server PTF SI16313

3015 Row - SQE Statistics Advised

- Generated by SQE when it determines that a column statistic needs to be collected or refreshed
- Commonly used columns
 - QVQTBL: Table name
 - QVQLIB: Table library name
 - QQC11: Statistics request type
 - 'N' - No Statistic Existed for Column
 - 'S' - Column Statistic was Stale
 - QQ1000: Name of column that statistic advised for
- QQC16 column in 3014 row will contain a 'Y' when SQE is used to process the SQL statement (CQE='N')
**SELECT qqc16, COUNT(*) FROM ??File Name
WHERE qqrid=3014 GROUP BY qqc16**

Analyzing Full Opens

- Which SQL requests are significantly affected by Full Opens:
SELECT SUM(qqi6) "Total Time" , COUNT(*) "Nbr Full Opens", qq1000
FROM ??File Name
WHERE qqjnum=??'xxxxxx' AND qqrid=1000 AND qqi5=0
AND qqc21 IN ('OP', 'DL', 'IN', 'UP')
GROUP BY qq1000 ORDER BY 1 DESC

Total Time	Nbr Full Opens	QQ1000
28536	87	SELECT T1.NOMINALQUANTITY, T1.LENGTH, T1.QUANT
27552	2	SELECT ORDERS.ORDERS_ID, STOREENT_ID FROM C
27432	1	SELECT * FROM PAYSYNCH WHERE PAYSRFNBR=? FC

Identifying and Tuning Problem Areas

- Best to first concentrate on repetitious non-reusable ODPs, table scans and long index builds
 - Also look for repetitious short-running queries that are not optimized well
 - Joins and sorts can be more difficult to analyze, but if they are accounting for a significant portion of run time, they need to be addressed as well
 - Fine tuning smaller problems should be done after large problems addressed
- Base set of queries designed to help produce useful results in most situations
 - Using these queries will help you understand the data and construct your own queries

Minimizing impact of running Detail Monitor

- Large monitor tables can slow analysis
 - Try collecting only the job you want, if possible
 - If not, subset the data down to a job/connection:
CREATE TABLE small AS (SELECT ... WHERE QQJNUM=xyz) WITH DATA
 - Only collect Monitor Data for long running SQL statements - based on Optimizer's Estimated Runtime
 - QAQQINI Option: DATABASE_MONITOR_THRESHOLD
 - Only those queries whose estimated runtime exceeds this threshold value (in secs) will be included in the database monitor collection
 - Default Value: All Statements (ie, 2147483647 seconds)
 - Table Filter - only collect statements referencing certain tables
 - STRDBMON OUTFILE(o) COMMENT('TABLEFILTER(lib1/tab1,lib2/tab3)')
 - V5R2 PTFs: SI15035, SI15139, SI15140, SI15142, SI15143, SI15154, SI15155, SI15157
 - V5R3 PTFs: SI15955, SI16331, SI16332, SI16333

- **iSeries Navigator**

- SQL Performance Monitors
- Summary->Memory-based Monitor collection
- Detailed -> STRDBMON collection
 - Includes the ability Start, Pause & End collections
 - Some beginner "canned" reports to help with analysis of monitor data
 - Drill-down with Visual Explain
- URL: <http://www.iseries.ibm.com/access>

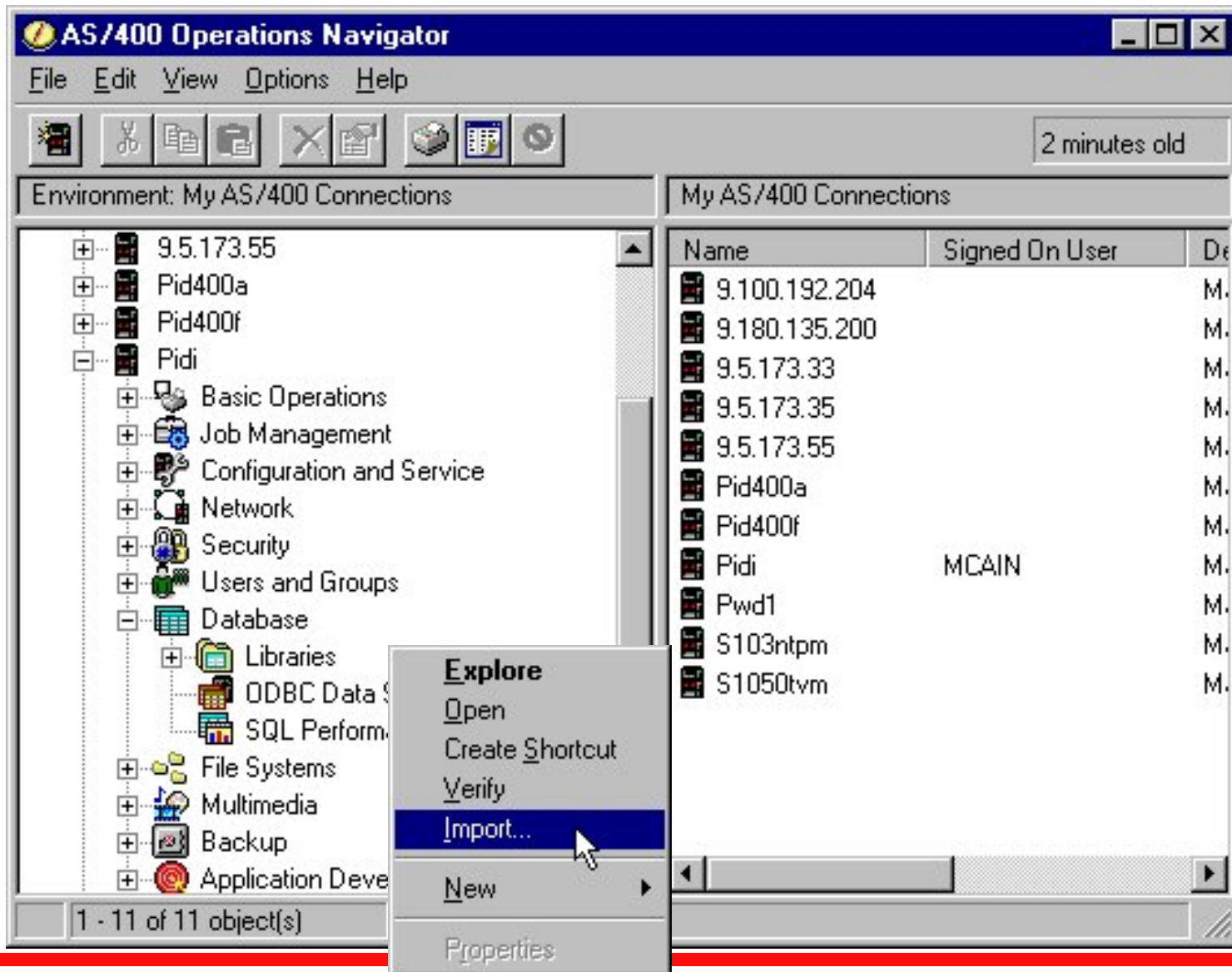
- **insure/SQL Toolset**

- Index Advisor based on Monitor Data
- Monitor Data Analysis services
- URL: <http://insureSQL.com>

- **Info & Education:**

- DB2 UDB for iSeries SQL Performance Workshop
ibm.com/servers/eserver/series/service/igs/db2performance.html
- White Paper: ibm.com/servers/enable/site/education/ibo/view.html?wp#db2
- Redbook – due out late June/early July
- Sample queries & reports: ibm.com/series/db2/dbmonqrys.htm

iSeries Navigator SQL Performance Monitors



Database Monitor Data

The screenshot shows the AS/400 Operations Navigator interface. The main window displays a tree view of system components, including 'Database' and 'SQL Performance Monitors'. An 'Import SQL Performance Monitor Files - Pidi' dialog box is open in the foreground. The dialog contains the following fields and options:

- Monitor name: [Empty text box]
- File: [Empty text box]
- Library: [Dropdown menu]
- Type of monitor:
 - Summary (Circled in red, with an arrow pointing to 'Memory Monitor')
 - Detailed (Circled in red, with an arrow pointing to 'STRDBMON')
- Buttons: OK, Cancel, Help

Collecting New Monitor Data

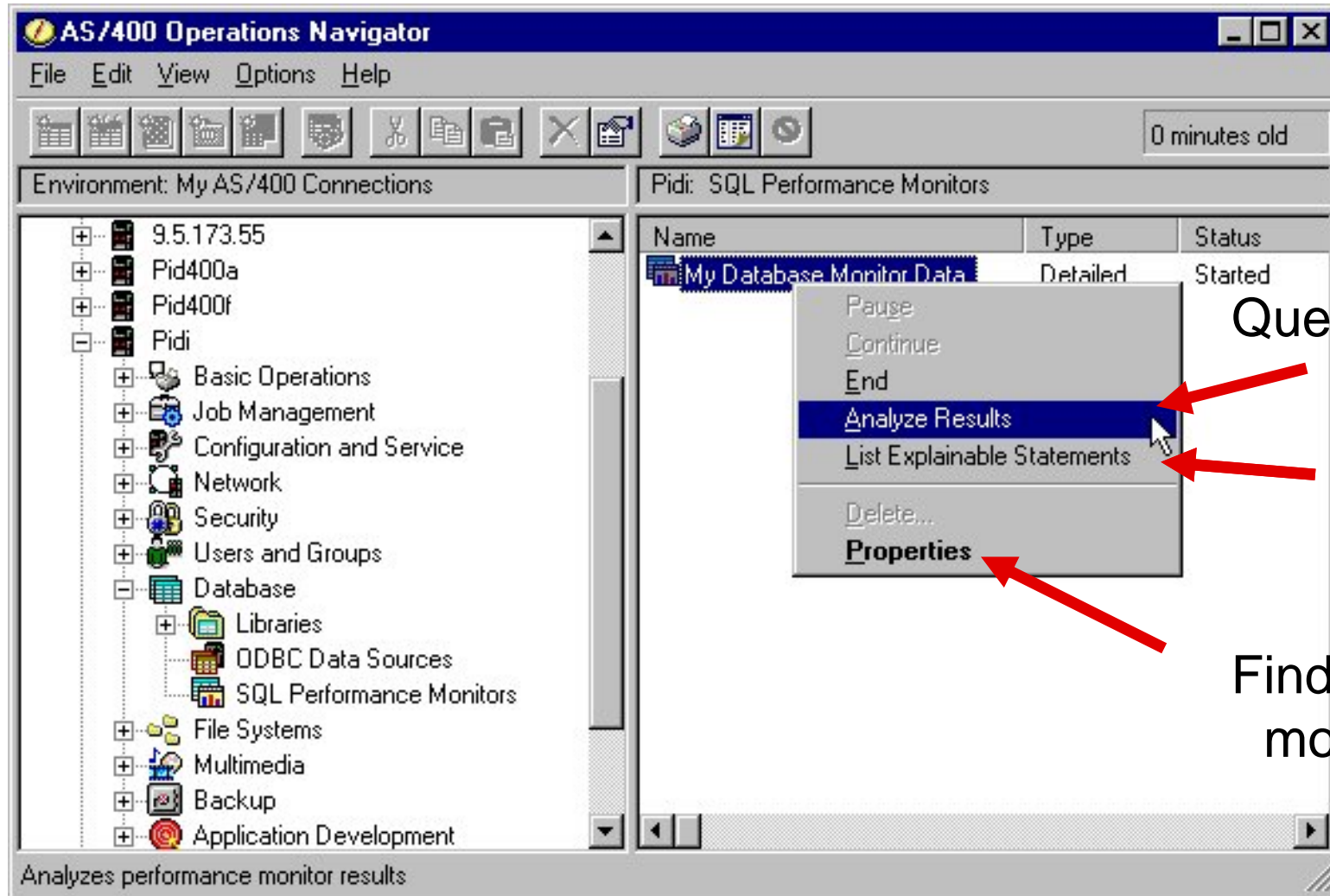
The screenshot shows the AS/400 Operations Navigator interface. The title bar reads "AS/400 Operations Navigator". The menu bar includes "File", "Edit", "View", "Options", and "Help". The toolbar contains various icons for navigation and actions. The status bar indicates "1 minutes old".

The main window is divided into two panes. The left pane, titled "Environment: My AS/400 Connections", shows a tree view of system components. The right pane, titled "Pidi: SQL Performance Monitors", displays a table of monitors.

Name	Type	Status
My Database Monitor Data	Detailed	Ended

A context menu is open over the "SQL Performance Monitors" entry in the tree view. The menu options are: Explore, Open, Create Shortcut, Verify, Import..., New, Summary, Detailed, and Properties. The "New" option is selected, and its sub-menu is visible, showing "Summary" and "Detailed" options.

Working with Monitor Data



Query monitor data

Visual Explain

Find location of monitor data

Query & Analyze Monitor Data

My Database Monitor Data Results - Pidi

Summary Results | Detailed Results | Extended Detailed Results

Collection period: From: 02/02/01 10:21:44 AM AM To: 02/02/01 10:25:43 AM

Select summary queries:

- General summary
- Job summary
- Operation summary
- Program summary
- SQL attributes summary
- Isolation level summary
- Statement use summary
- Open summary
- Data access summary
- Statement type summary
- Parallel processing summary
- Optimizer summary

Choose a report → Select All

Change the query/report → Modify Selected Queries

Run the query/report → View Results

OK Cancel Help

Query & Analyze - Summary Reports

- Summary Report: all reports summarized at a database monitor collection level
 - Good for getting a high-level perspective of what was happening during the period when DB monitor data was collected
 - Analyzing the type of SQL operations
 - Identifying the jobs that are consuming DB2 resources the most
 - Reports do not pinpoint the cause of performance problems
 - Operation Summary & Job Summary good reports to determine a place to start

Operation	Total Statements	Total Runtime (sec)	Maximum Runtime	Average Runtime	Maximum Open Time
OPEN	23768	45.934	.694	.001	.694
CLOSE	23536	24.436	.021	.001	0
COMMIT	635	4.153	.107	.006	0

Query & Analyze - Extended Detailed Reports

- **Extended Detailed Report: all reports show each SQL request**
 - Gives you a DB2 trace at an SQL statement level
 - Statements are not sorted or grouped by job, so the analyst needs to determine how to address this issue
 - Copy all the data for a single job into it's own collection
 - Modify reports???
 - Most useful reports are:
 - Open Information
 - Index Created Information

Start Time	End Time	Total Runtime (sec)	Operation	Statement Text
2003-06-08 21:20:06.136557	2003-06-08 21:20:06.830742	.694184	OPEN	SELECT T1.FIELD1, T1.FLAGS,
2003-06-08 21:20:29.563669	2003-06-08 21:20:29.781493	.217824	OPEN	SELECT T1.FIELD1, T1.FLAGS,
2003-06-08 21:20:18.363505	2003-06-08 21:20:18.574435	.210928	OPEN	SELECT T1.FIELD1, T1.FLAGS,

Visual Explain & DB Monitors

Explainable Statements For SQL Performance Monitor My Database Monitor Data

SQL statements monitored: ****All columns sortable with V5R3**

Date	Time	Processing Time	SQL Text
02/02/01	10:24:12 A...	121 ms	select * from starlib/time_dim where datekey > ?
02/02/01	10:24:23 A...	258 ms	select * from starlib/time_dim where datekey > ?
02/02/01	10:25:28 A...	998 ms	select * from starlib/cust_dim c, starlib/item_fact i where c.custkey = i.custkey

↑
QQSTIM

SQL statement selected:

```
select * from starlib/time_dim where datekey > ?
```

Refresh

Run Visual Explain

Close Help

Visual Explain

Visual Explain for SELECT A.YEAR, A.MONTH, A.RETURNFLAG, B.PART, B.MFGR, A.QUANTITY, A.REVENUE_WO_TAX ...

File View Actions Help

Attribute	Value
Estimated Rows Selected	87
Join Position	1
Table Data Space Number	1
Number of Tables Joined	2
I/O or CPU Bound	CPU BOUND
Information about the index scan ...	
Index Scan - Key Positioning	Y
Number of Columns Using Key P...	3
Estimated Entries Using Key Posi...	87
Index Scan - Key Selection	N
Index Only Access	N
Index Fits in Memory	Y
Memory Pool Size	12519688
Memory Pool ID	2
Type of Index	ENCODED VECTOR INDEX
INDEX USAGE	TERTIARY INDEX
Number of Index Entries	100029
Number of Unique Index Values	69
Percent Overflow for EVI	0
Vector Size for EVI	1
Size of Index, in Bytes	245760
Page Size of Index, in Bytes	4096
Reason Code	I5
Index is a Constraint	N
Information about the table scan p...	
Data Space Selection	N
Derived Selection Performed	N