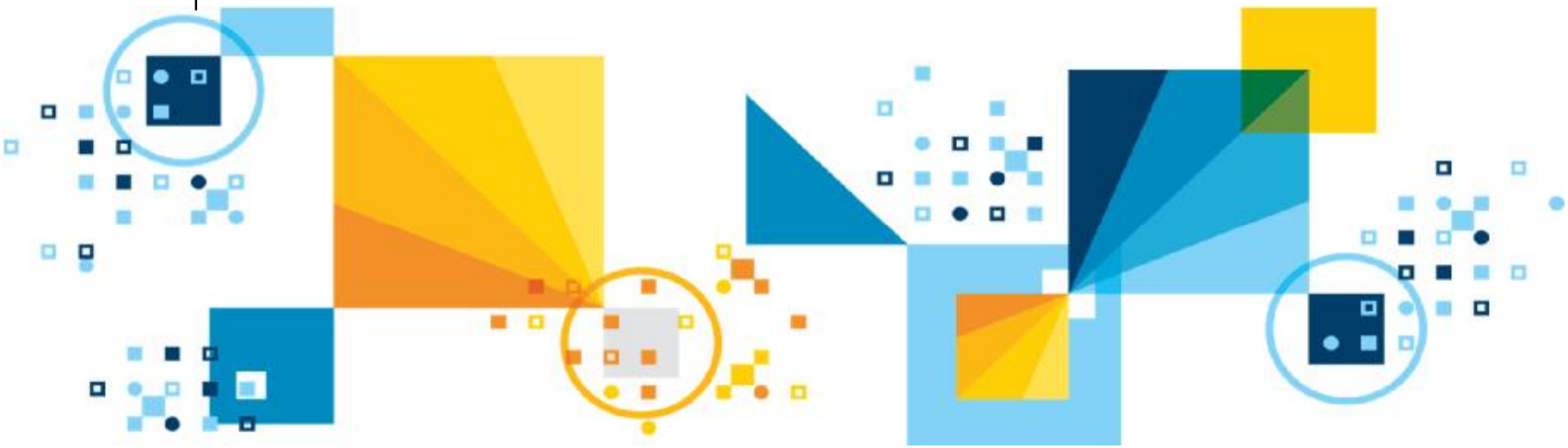
DB2 Performance Monitoring Essentials II

Module ID	10302
Length	1 hour + 1.5 hour hands on lab



For questions about this presentation contact askdata@ca.ibm.com

February 9, 2015

PRESENTATION NOTES FOR PAGE 1

Disclaimer

PRESENTATION NOTES FOR PAGE 2

Module Information

- You should have completed or acquired the necessary knowledge for the following modules in order to complete this module:
  - Module DB2 Performance Monitoring Essentials

- After completing this module, you should be able to:
  - Describe the major functions of the db2 monitoring interfaces
  - Able to perform the tasks to collect db2 performance monitoring data

PRESENTATION NOTES FOR PAGE 3

Agenda

- Introduction to Monitoring Essentials
- Monitoring Interfaces Available with DB2
- **Essential Monitoring Targets**
- Performance Monitoring Tools and Methodology
- Summary

PRESENTATION NOTES FOR PAGE 4

Performance Monitoring Methodology

- **Operational monitoring strategy**
  - Needs to be very light weight
  - Analysis and comparison of monitoring data
  - Do not limit yourself to just metrics that the DB2 product provides
- **Types of data are useful to collect**
  - A basic set of DB2 system performance monitoring metrics
  - DB2 configuration information
  - Overall system load
  - Throughput and response time measured at the business logic level

PRESENTATION NOTES FOR PAGE 5

Performance Monitoring Methodology - Continued

§ Basic set of system performance monitor elements

- The number of transactions executed
- Analysis and comparison of monitoring data
- Buffer pool hit ratios, measured separately for data, index, XML storage object, and temporary data
- Buffer pool physical reads and writes per transaction
- The ratio of database rows read to rows selected
- The amount of time spent sorting per transaction
- The amount of lock wait time accumulated per thousand transactions
- The number of deadlocks and lock timeouts per thousand transactions
- The number of dirty steal triggers per thousand transactions
- The number of package cache inserts per thousand transactions
- The time an agent waits for log records to be flushed to disk
- In partitioned database environments, the number of fast communication manager (FCM) buffers sent and received between partitions

PRESENTATION NOTES FOR PAGE 6

A "dirty steal" is the least preferred way to trigger buffer pool cleaning. Essentially, the processing of an SQL statement that is in need of a new buffer pool page is interrupted while updates on the victim page are written to disk. If dirty steals are allowed to happen frequently, they can have a significant impact on throughput and response time.

Essential Monitoring Targets

§ Track key performance indicators

- Practical approach, too many = diminished returns
- Key DB2 monitoring elements as indicators
  - Stand alone elements
  - Calculate ratios
- DB2 monitoring elements can be captured from:
  - Snapshot monitoring
    - Switch based
    - Some CPU overhead (1-10%)
    - Easy to reset counters
  - Monitor table functions and views via SQL
    - SQL based, easy to tabularize
    - Monitoring elements queried from memory, lower overhead than Snapshot based monitoring

PRESENTATION NOTES FOR PAGE 7

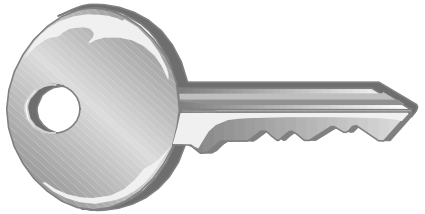
Now you know about some of the basic tools and interfaces for monitoring your DB2 systems, like the new SQL based, in memory monitoring framework, Event based Monitoring, and snapshot monitoring. The question is exactly what should you monitor? Certainly not everything, certainly not nothing. This next section will define some of the key monitoring information to gather. We will look at how to formulate ratio data points with to keep “good tabs” on your system. Of course what you choose to monitor may depend on the application and database layout at hand, but these basic monitoring elements should give you a good Idea as to what to monitor, along with emphasizing the old axiom less can be more.

Depending on the version of DB2 that you are working with you may not have a choice, but let’s say you have a shiny new version of DB2 that has the monitoring framework available but perhaps consider it a bit inchoate or unfamiliar compared to the SNAP SHOT monitoring. But then you start thinking about the fact that snapshot monitoring it is not the monitoring choice of the future, and less and less will rely upon it, and overall, the fact that Snapshot monitoring is not as rich of a tool as the new ones. In general, it is recommended to use the monitoring framework where practical and where possible even if you are not currently using the Work Load Manager feature. Mainly because this is the direction that monitoring is taking. You may want to start by using the monitor table functions that deal with gathering performance metrics on “objects” like tables, indexes and bufferpools. These are the simplest of the new functions to work with and are similar in scope to some of the snapshot monitoring you may be familiar with. One thing to look out for is the fact that some of the new metrics are not dynamically re-settable.

The following slides will present some key performance monitoring data points to collect on a routine basis, and some sample SQL. These metrics will serve you well when saved in tables to help identify trends over time.

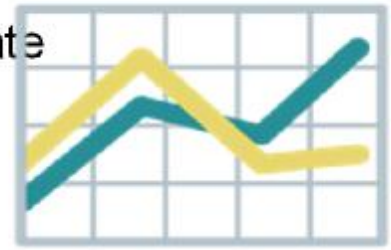
Essential Monitoring Targets – Database Level

- **Key areas**
  - **TOTAL TRANSACTIONS**
  - **BUFFER POOL HIT RATIOS (DATA, INDEX, TEMP)**
  - **READS AND READ EFFICIENCIES**
  - **SORTING**
  - **LOCKS**
  - **PAGE CLEANING**
  - **PACKAGE CACHE CAPACITY**
  - **TRANSACTION LOGS**



- **Save data to tables**
  - Aggregate, differentiate, interpolate, extrapolate

modulate



- **Look for trends**
  - Avoid catastrophes
  - Easier to fix before broken

PRESENTATION NOTES FOR PAGE 8

Performance monitoring metrics should be sampled on a regular basis. What will make performance monitoring effective is to focus on the metrics that give you a good picture of how things are performing and be able to compare past performance with present performance and project future performance. One way this can be achieved is to save your data to tables where samples can be compared and trended over time. You might want to find an application or program that can convert this data into some simple graphs or histograms to make it easier to see any trends. We will look at some of the key areas that should be sampled regularly:

Total Transactions – This metric helps to keep track of just how many “pieces” of work the database system is doing. It can be useful in determining overall activity on the system. It also serves as a useful denominator when calculating ratios of particular ordinal based metrics that are critical to performance.

BufferPools are the main memory heaps critical to make data readily available from memory instead of having to go to disk, which of course is supremely faster than disk access. Memory rates are measured in nanoseconds while disk transfer rates are in the milliseconds, 100s of thousands of times slower.

Reads and Reads efficiency – the most common and costly activity on a database is read activity. It is critical to only read what you need to make your system most efficient

Sorting – The processes that establish the data in a specified order, can be costly when disk resources are required to complete sort requests.

Locks – Locks restrict and in some cases prevent concurrent access to data. In doing so they can slow a transactions to a point where transactions start queueing up waiting for a series or chain of logs to subside.

Page Cleaning – Keeping data that is going to be accessed multiple times in memory while leaving enough room for new data to be read into from disk is a precarious thing to achieve, given the variability in the requests and kind of requests being made for the data in a database. When there is no “room” in the bufferpool to read in new data a set of victim pages must be established and written to disk before the new data can be retrieved. Monitoring this kind of activity is important to maintaining good performance on your database system.

Package Cache Capacity – As with most relational database systems there is a cache for the data, and a cache for the code that executes against the database, it is important to try to keep this package cache sized so as not to cause package cache overflows.

Transaction Logs - With regards to transaction logs we want to make sure that they have very low write times while maintaining sufficiently low recovery times in the event of a crash.

Essential Monitoring Targets – TOTAL TRANSACTIONS

- **Total number of transactions executed by applications:**

```
COMMIT_SQL_STMTS + ROLLBACK_SQL_STMTS
```

  - Snapshot monitoring: database or application level
  - Event Monitor: database or connection level

- **Total number of units of work:**

```
COMMIT_SQL_STMTS + INT_COMMITS +
ROLLBACK_SQL_STMTS + INT_ROLLBACKS
```

- **Total number of commit and rollback statements issued by the client application**
  - System Monitor Table Functions or Event Monitors:

```
TOTAL_APP_COMMITS + TOTAL_APP_ROLLBACKS
```

- **Useful for creating key ratios like reads/commit**
  - Adds element of “relativity” to monitoring

PRESENTATION NOTES FOR PAGE 9

The number of transactions executed:

TOTAL_TRXs This provides an excellent base level measurement of system activity.

It can be derived by adding together the commits and rollbacks issued from an application. There are a number of metrics that gain significance when this is used as a denominator to create ratio data. Currently there are no comparable elements in the Monitoring Framework.

Essential Monitoring Targets – BP HIT RATIO

- **BUFFER POOL HIT RATIOS, measured separately for DATA, INDEX and XDA**
  - **MON_BP_UTILIZATION Administrative View**
- **For each Bufferpool:**
  - **DATA_HIT_RATIO_PERCENT**
    - Percentage of time that the database manager did not need to load a page from disk to service a data page request
  - **INDEX_HIT_RATIO_PERCENT**
    - Percentage of time that the database manager did not need to load a page from disk to service an index data page request
  - **XDA_HIT_RATIO_PERCENT**
    - Auxiliary storage objects hit ratio, that is, the percentage of time that the database manager did not need to load a page from disk to service a data page request for XML st

```
SELECT SUBSTR(bp_name ,1,30)   as BPNAME,
       data_hit_ratio_percent  as DATA_HR,
       index_hit_ratio_percent as INDEX_HR,
       xda_hit_ratio_percent   as XDA_HR
  FROM SYSIBMADM.MON_BP_UTILIZATION
```

PRESENTATION NOTES FOR PAGE 10

The MON_BP_UTILIZATION administrative view returns key monitoring metrics, including hit ratios and average read and write times, for all buffer pools and all database partitions in the currently connected database. It provides information that is critical for performance monitoring, because it helps you check how efficiently you are using your buffer pools.

Data hit ratio, that is, the percentage of time that the database manager did not need to load a page from disk to service a data page request.

Index hit ratio, that is, the percentage of time that the database manager did not need to load a page from disk to service an index data page request.

Auxiliary storage objects hit ratio, that is, the percentage of time that the database manager did not need to load a page from disk to service a data page request for XML storage objects (XDAs). In a DB2® pureScale® environment, this value is the percentage of time the database manager used to locate a data page for an XDA in the local buffer pool.

Essential Monitoring Targets – BP HIT RATIO

- **BUFFER POOL HIT RATIOS, measured globally and separately for DATA, INDEX and XDA**
  - **BP_HITRATIO Administrative View**
  - **EXAMPLE**: Returns bufferpool hit ratios, including total hit ratio, data hit ratio, XDA hit ratio and index hit ratio, for all bufferpools and all database partitions in the currently connected database

```
SELECT substr(db_name,1,8) as db_name
     , substr(bp_name,1,14) as bp_name
     , total_hit_ratio_percent
     , data_hit_ratio_percent
     , index_hit_ratio_percent
     , xda_hit_ratio_percent
     , dbpartitionnum
  FROM SYSIBMADM.BP_HITRATIO
 ORDER BY dbpartitionnum
```

OLTP

GOOD HIT RATIO:

Data: > 80-85%

Indexes: > 90-95%

  - **SNAP_GET_BP Table Function**
    - Use to aggregate results from all partitions or report on single partition
  - **GET SNAPSHOT FOR ALL BUFFERPOOLS Command**

PRESENTATION NOTES FOR PAGE 11

Here we see an example of capturing the bufferpool hit ratio with the BP_HITRATIO administrative view.

Can use administrative table function and Get snapshot command too.

Note: This administrative view works only in DB2® environments without the IBM® DB2 pureScale® Feature.

Usage notes

The ratio of physical reads to total reads gives the hit ratio for the bufferpool. The lower the hit ratio, the more the data is being read from disk rather than the cached buffer pool which can be a more costly operation.

The following is an example of output for this query.

DB_NAME BP_NAME TOTAL_HIT_RATIO_PERCENT DATA_HIT_RATIO_PERCENT INDEX_HIT_RATIO_PERCENT XDA_HIT_RATIO_PERCENT DBPARTITIONNUM

TEST	IBMDEFAULTBP	63.09	68.94	43.20	-	0
TEST	IBMSYSTEMBP4K	- -	...	-	-	0
TEST	IBMSYSTEMBP8K	- -	...	-	-	0
TEST	IBMSYSTEMBP16K	- -	...	-	-	0
TEST	IBMSYSTEMBP32K	- -	...	-	-	0

Essential Monitoring Targets – I/O EFFICIENCY

§ Number of ROWS READ PER TRANSACTION

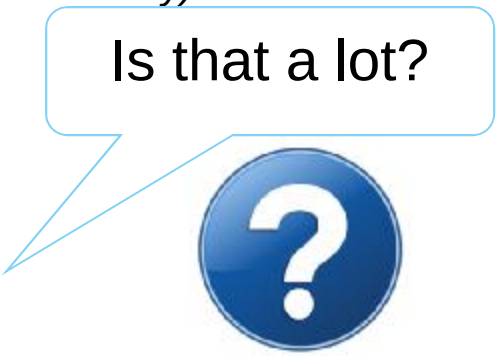
NOTE: is not the # of rows that were returned to the calling application, but the # of rows that had to be read from the table in order to return the result set (table scan vs index access only)

- **SNAPDB Administrative View**
- **SNAP_GET_DB Table Function**
- **GET SNAPSHOT FOR DATABASE Command**

OLTP

< 10 Excellent **10 - 20 Very Good**

20 - 40 Fair **> 50 Tune**




```
SELECT VARCHAR(db_name,10)
     , CASE WHEN (commit_sql_stmts + rollback_sql_stmts) > 0
       THEN DEC(((rows_read)
                  / commit_sql_stmts + rollback_sql_stmts), 13, 2)
       ELSE NULL
        END AS READS_PER_TRANSACTION
     , rows_read  as ROWS_READ
     , commit_sql_stmts + rollback_sql_stmts as TOTAL_TRX
     , db_conn_time as FIRSTDB_CONN
     , last_reset as LAST_RESET
  FROM SYSIBMADM.SNAPDB;
```

PRESENTATION NOTES FOR PAGE 12

Fast and efficient I/O is the most critical area and resource needed for a well performing database. The most costly I/O stems from read operations, of course you must perform reads to retrieve your data, but you should keep a watchful eye on how efficient the reads on a system are. Here we use the number of transactions as a denominator so as not to just measure how many reads but how many reads per transaction. On an oltp system a range of <10 is excellent, 10-20 good, 20-40 fair and above 50 you may want to start looking at your SQL, indexes and tables that you store your data in. Data Warehouse systems the numbers go way up …… (Need some #s)

File = Reads_per_trx.db2

Essential Monitoring Targets – I/O EFFICIENCY

§ Total amount of CPU TIME

– **MON_PKG_CACHE_SUMMARY Administrative View**

– **Aggregate metrics overall executions of each SQL statement (static or dynamic) in the cache:**

- **TOTAL_CPU_TIME**

  Total amount of CPU time, in microseconds, used while within the DB2® database manager (combined total of both user and system CPU time)

- **TOTAL_LOCK_WAIT_TIME**

  Total elapsed time, in milliseconds, spent waiting for locks

- **TOTAL_IO_WAIT_TIME**

  The t… …tions

```
SELECT total_cpu_time
     , total_lock_wait_time
     , total_io_wait_time
     , avg_io_wait_time
     , avg_lock_wait_time
  FROM SYSIBMADM.MON_PKG_CACHE_SUMMARY
 ORDER BY total_cpu_time DESC
 FETCH FIRST 20 ROWS ONLY
```

PRESENTATION NOTES FOR PAGE 13

The MON_PKG_CACHE_SUMMARY administrative view returns key metrics for both static and dynamic SQL statements in the cache, providing a high-level summary of the database package cache. The metrics returned are aggregated over all executions of the statement across all members of the database.

TOTAL_CPU_TIME. The total amount of CPU time, in microseconds, used while within the DB2® database manager. This value represents the combined total of both user and system CPU time.

TOTAL_LOCK_WAIT_TIME. The total elapsed time, in milliseconds, spent waiting for locks.

TOTAL_IO_WAIT_TIME. The total elapsed time, in milliseconds, spent on I/O operations.

AVG_ACT_WAIT_TIME. Average time spent waiting for database activities per statement execution.

Essential Monitoring Targets - LOCK WAIT TIME

§ Information for each workload

- SYSPROC.MON_GET_WORKLOAD Table Function

```
SELECT varchar(workload_name,30) as workload_name
     , sum(lock_wait_time) as total_lock_wait_time
     , sum(lock_waits) as total_lock_waits
     , sum(lock_timeouts) as total_lock_timeouts
     , sum(lock_escals) as total_lock_escals
  FROM TABLE(MON_GET_WORKLOAD('',-2)) AS t
 GROUP BY workload_name
 ORDER BY total_lock_wait_time DESC;
```

-

```
SELECT varchar(wlmetrics.workload_name,30) as workload_name,
       sum(detmetrics.lock_wait_time) as total_lock_wait_time,
       sum(detmetrics.lock_waits) as total_lock_waits,
       sum(detmetrics.lock_timeouts) as total_lock_timeouts,
       sum(detmetrics.lock_escals) as total_lock_escals
FROM TABLE(MON_GET_WORKLOAD_DETAILS('',-2)) AS WLMETRICS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
          '$detmetric/db2_workload' PASSING
          XMLPARSE(DOCUMENT WLMETRICS.DETAILS)
          as "detmetric"
COLUMNS "LOCK_WAIT_TIME" INTEGER PATH 'system_metrics/lock_wait_time',
        "LOCK_WAITS" INTEGER PATH 'system_metrics/lock_waits',
        "LOCK_TIMEOUTS" INTEGER PATH 'system_metrics/lock_timeouts',
        "LOCK_ESCALS" INTEGER PATH 'system_metrics/lock_escals'
) AS DETMETRICS
```

PRESENTATION NOTES FOR PAGE 14

The MON_GET_WORKLOAD table function returns metrics for one or more workloads.

An input argument of type VARCHAR(128) that specifies a specific workload for which the metrics are to be returned. If the argument is NULL or an empty string, metrics are returned for all workloads. member An input argument of type INTEGER that specifies a valid member in the same instance as the currently connected database when calling this function. Specify -1 for the current database member, or -2 for all database members. If the NULL value is specified, -1 is set implicitly.

Usage notes

The metrics returned by the MON_GET_WORKLOAD table function represent the accumulation of all metrics for requests that were submitted by connections mapped to the identified workload object. Metrics are rolled up to a workload on unit of work boundaries, and periodically during the execution of requests. Therefore, the values reported by this table function reflect the current state of the system at the time of the most recent rollup. Metrics are strictly increasing in value. To determine the value of a given metric for an interval of time, use the MON_GET_WORKLOAD table function to query the metric at the start and end of the interval, and compute the difference.

Request metrics are controlled through the COLLECT REQUEST METRICS clause on service superclass and the mon_req_metrics database configuration parameter at the database level. Metrics are only collected for a request if the request is processed by an agent in a service subclass whose parent service superclass has request metrics enabled, or if request metrics collection is enabled for the entire database. By default, request metrics are enabled at the database level. If request metrics have been disabled at the database level, and for a service superclass, then the metrics reported for each workload that is mapped to that service superclass stop increasing (or remain at 0 if request metrics were disabled at database activation time).

The MON_GET_WORKLOAD table function returns one row of data per workload and per member. No aggregation across workloads (on a member), or across members (for a service class or more), is performed. However, aggregation can be achieved through SQL queries (see the example).

The MON_GET_WORKLOAD_DETAILS table function returns detailed metrics for one or more workloads.

Essential Monitoring Targets - SORTING

- **SORTING metrics**
  - **SYSPROC.MON_GET_WORKLOAD Table Function**
  - **TOTAL_SECTION_SORT_TIME / (TOTAL_APP_COMMITS + TOTAL_APP_ROLLBACKS)**
  - **SORT_OVERFLOWS / TOTAL_SORTS = % of sorts that need more heap space**
  - **TOTAL_SECTION_SORT_TIME / TOTAL_SORTS = average sort time**

```
SELECT VARCHAR(workload_name,30)
, CASE WHEN (total_app_commits + total_app_rollbacks) > 0
  THEN DEC((total_section_sort_time) / (
              (total_app_commits) + (total_app_rollbacks)),8,5)
  ELSE NULL
   END AS SORTTIME_PER_TRX
, CASE WHEN total_sorts > 0
  THEN ((total_section_sort_time) *.001)/(total_sorts)
  ELSE NULL
   END as AVG_SORTTIME
, total_sorts as TOTAL_SORTS
, total_section_sort_time as TOTAL_SORTTIME
, sort_overflows as TOTALSORTOVERFL
, (total_app_commits + total_app_rollbacks) as TotalTransactions
  FROM TABLE(SYSPROC.MON_GET_WORKLOAD(‘’,-2)) AS T;
```

PRESENTATION NOTES FOR PAGE 15

The amount of time spent sorting per transaction:TOTAL_SECTION_SORT_TIME / TOTAL_COMMITS This is an efficient way to handle sort statistics, because any extra overhead due to spilled sorts automatically gets included here. That said, you might also want to collect TOTAL_SORTS and SORT_OVERFLOWS for ease of analysis, especially if your system has a history of sorting issues.

Essential Monitoring Targets – AVERAGE LOG DISK WAIT TIME

- **Time an agent spends waiting for log records to be flushed to disk**
  - **MON_GET_WORKLOAD Table Function**
  - **LOG_DISK_WAIT_TIME**
    - The amount of time (in milliseconds) an agent spends waiting for log records to be flushed to disk
  - **LOG_DISK_WAITS_TOTAL**
    - The number of times agents have to wait for log data to write to disk while copying log records into the log buffer

```
SELECT varchar(workload_name, 30) as WORKLOAD_NAME
     , CASE WHEN log_disk_wait_time > 0
       THEN DEC(FlOAT(log_disk_waits_total)/
                              FLOAT(log_disk_wait_time), 10, 7)
       ELSE NULL END as AVG_LOGDISK_WAIT_TIME_MS
     , log_disk_wait_time as LOG_DISK_WAIT_TIME
     , log_disk_waits_total as LOG_WAITS_TOTAL
  FROM TABLE(MON_GET_WORKLOAD('', -2)) AS T;
```

PRESENTATION NOTES FOR PAGE 16

log_disk_wait_time

The amount of time an agent spends waiting for log records to be flushed to disk. The value is given in milliseconds.

log_disk_waits_total

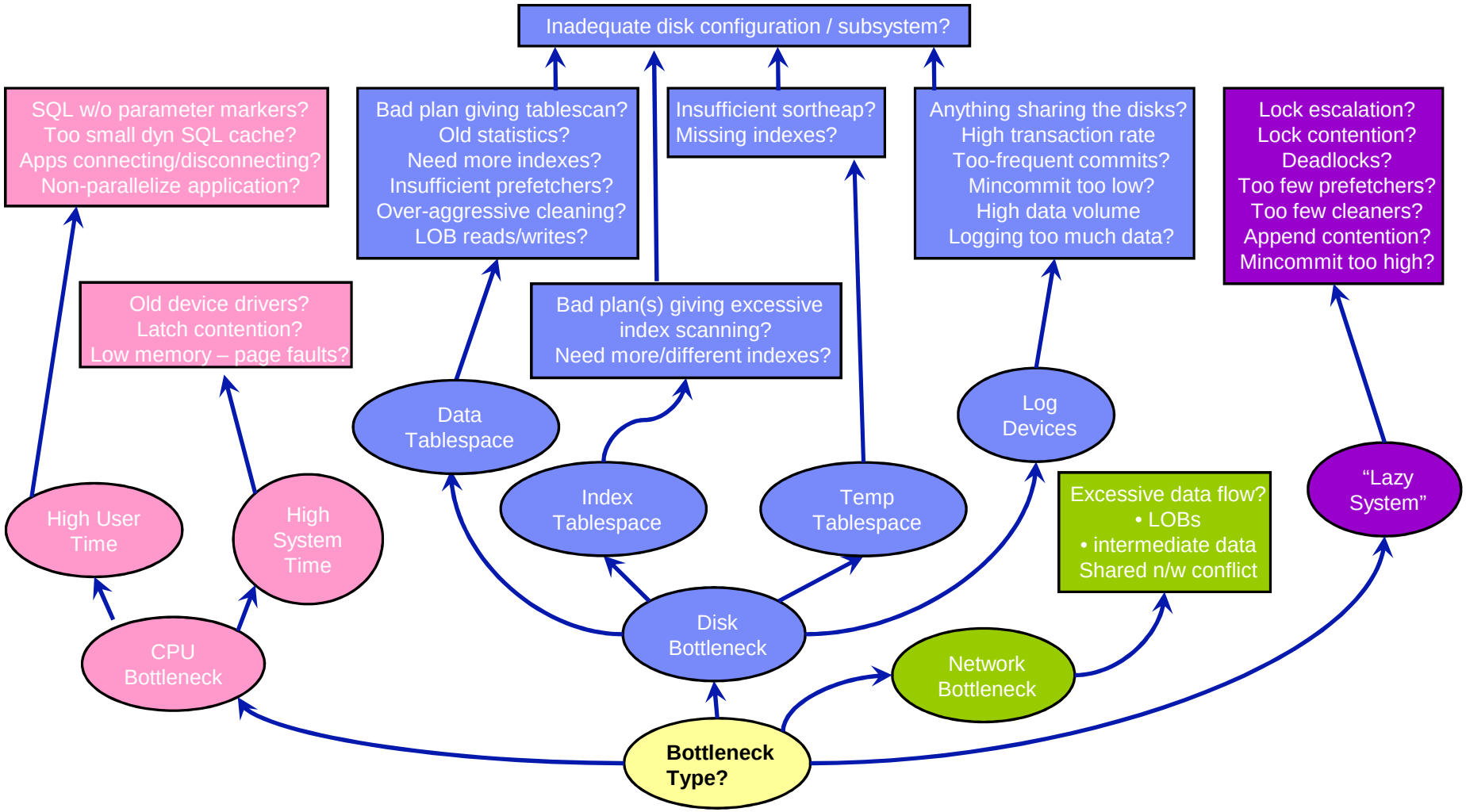
The number of times agents have to wait for log data to write to disk.

Agenda

- **Introduction to Monitoring Essentials**
- **Monitoring Interfaces available with DB2**
- **Essential Monitoring Targets**
- **Performance Monitoring Tools**
  - System Monitoring Tools
  - DB2 Monitoring Tools
    - db2top
    - IBM Optim Performance Manager
- **Summary**

PRESENTATION NOTES FOR PAGE 17

How Can System Monitoring Lead You To The Root Activity?



PRESENTATION NOTES FOR PAGE 18

This picture shows why you might want to start your diagnosis of a performance issue with looking for system bottlenecks with system monitoring tools.

Basically by determining where the bottleneck is occurring you can effectively isolate what kind of DB2 activity that might be a root cause for performance issues.

System Monitoring tools ß à System Bottlenecks

§ **Common System Monitoring Tools**

- **iostat ß à** Disk Bottlenecks
  - Useful for monitoring I/O activities
- **vmstat/sar ß à** CPU, Memory, Disk Bottlenecks
  - Useful for determining if something is suspended or just taking a long time
- **Perfmon** (windows) **ß à** CPU, Memory, Disk Bottlenecks
- **nmon** (aix et.al) **ß à** CPU, Memory, Disk Bottlenecks
- **netstat ß à** network Bottlenecks
  - Useful for isolating network problems
- **top ß à** CPU, Memory Bottlenecks

PRESENTATION NOTES FOR PAGE 19

There are system monitoring tools a plenty, we will discuss some of the more commonly used tools and try to give you a good idea as to what each tool excels at.

System Monitoring tools - IOSTAT

- Useful for monitoring I/O activities
  - Read and write rate can be used to estimate the amount of time required for certain SQL operations (if they are the only activity on the system).
- Are you I/O bound?
- Run at regular intervals 3-5 seconds
- Where is DB2 making use of this device?
- EXAMPLE:
  - List containers on all database members that have the highest read time

```
CONTAINER_NAME                              TBSP_NAME   POOL_READ_TIME
------------------------------------------- ----------- --------------
/home/swalkty/NODE0000/T0000000/C0.CAT      SYSCATSPACE            597
/home/swalkty/NODE0000/T0000002/C0.LRG      USERSPACE1              42
/home/swalkty/NODE0000/T0000001/C0.TMP      TEMPSPACE1               0
```

```
SELECT varchar(container_name,70) as container_name
     , varchar(tbsp_name,20) as tbsp_name
     , pool_read_time
  FROM TABLE(MON_GET_CONTAINER('',-2)) AS t
 ORDER BY pool_read_time DES
```

PRESENTATION NOTES FOR PAGE 20

iostat

This command is useful for monitoring I/O activities. You can use the read and write rate to estimate the amount of time required for certain SQL operations (if they are the only activity on the system).

The MON_GET_CONTAINER table function returns monitor metrics for one or more table space containers.

An input argument of type VARCHAR(128) that specifies a valid table space name in the same database as the one currently connected to when calling this function. If the argument is null or an empty string, metrics are returned for all containers in all table spaces in the database.

An input argument of type INTEGER that specifies a valid member in the same instance as the currently connected database when calling this function. Specify -1 for the current database

Usage notes

The MON_GET_CONTAINER table function returns one row of data per container and per database member. Data can be returned for all containers in a given table space, or for all containers in the database. No aggregation across database partitions is performed. However, aggregation can be achieved through SQL queries.

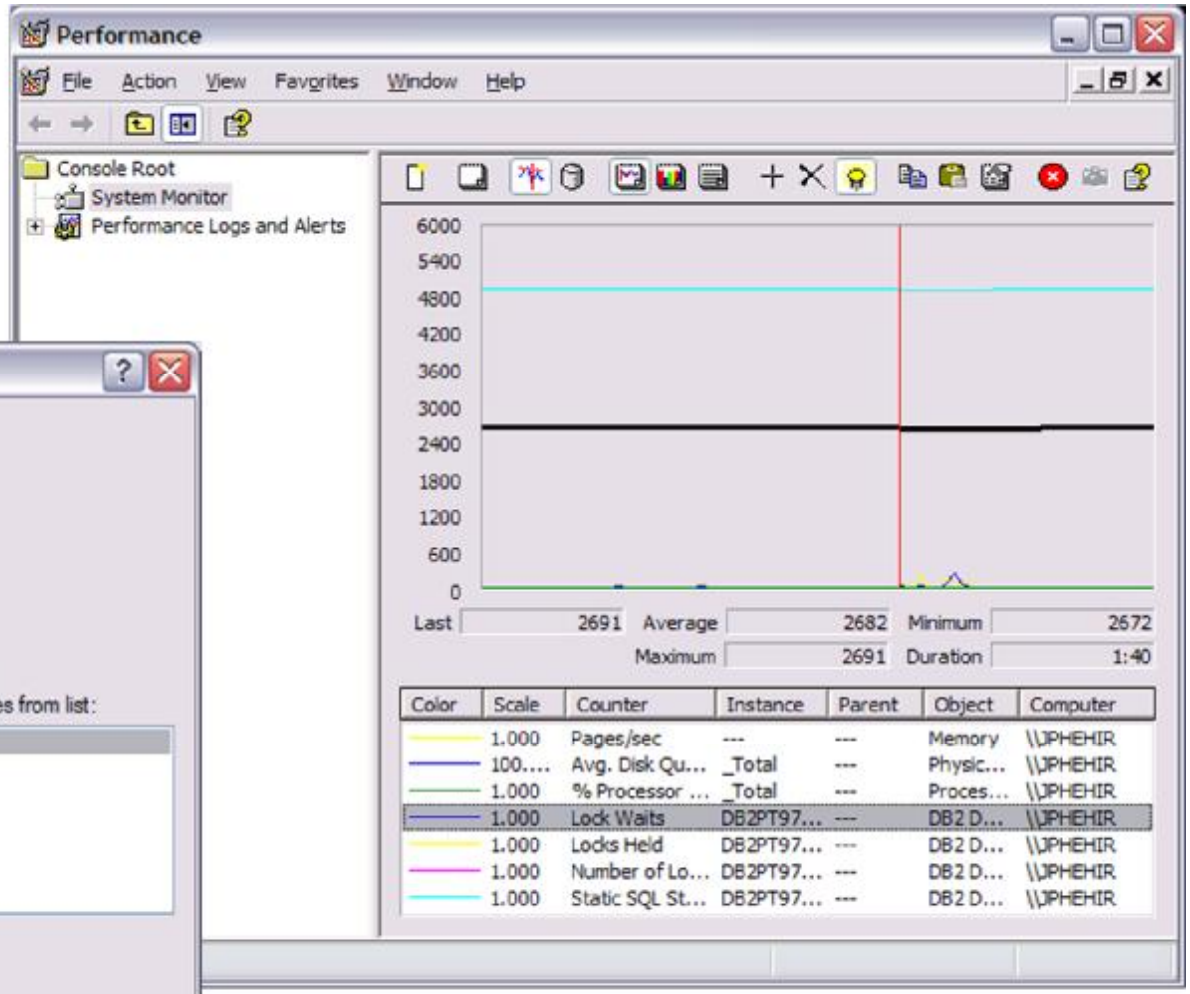
e member, or -2 for all database members. If the null value is specified, -1 is set implicitly.

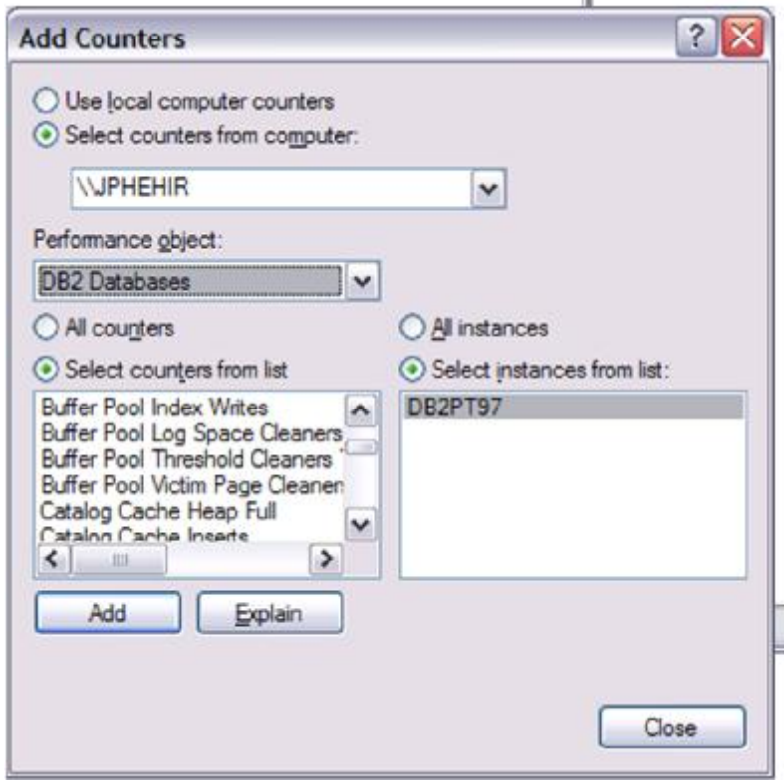
pool_read_time - Total buffer pool physical read time monitor element

Indicates the total amount of time spent reading in data and index pages from the table space containers (physical) for all types of table spaces. This value is given in milliseconds.

System Monitoring tools - PERFMON (Windows)

- **Integrated DB2 Metrics**
  - Application level
  - DB Level
  - DBM Level





PRESENTATION NOTES FOR PAGE 21

A useful tool on Windows based operating systems is Perfmon. Just the default indicators can tell you a lot. Disk queuing can help to define I/O bottlenecks, paging is due to a dearth of real memory being available and can lead to I/O contention on especially on small systems with few disk resources.

Db2 monitoring elements can be integrated directly into performance monitor. Performance monitoring can be scheduled and pre-configured. Results can be saved in archives and replayed.

Processor utilization

System\Processor Queue Length - number of threads queued and waiting for time on the CPU. Divide this by the number of CPUs in the system. If the answer is less than 10, the system is most likely running well.

Memory utilization

Memory\Pages Input/Sec - The best indicator of whether you are memory-bound, this counter shows the rate at which pages are read from disk to resolve hard page faults. In other words, the number of times the system was forced to retrieve something from disk that should have been in RAM. Occasional spikes are fine, but this should generally flat line at zero.

Disk Utilization

PhysicalDisk\Current Disk Queue Length\driveletter - this is probably the single most valuable counter to watch. It shows how many read or write requests are waiting to execute to the disk. For single disks, it should idle at 2-3 or lower, with occasional spikes being okay. For RAID arrays, divide by the number of active spindles in the array; again try for 2-3 or lower. Because a shortage of RAM will tend to beat on the disk, look closely at the Memory\Pages Input/Sec counter if disk queue lengths are high.

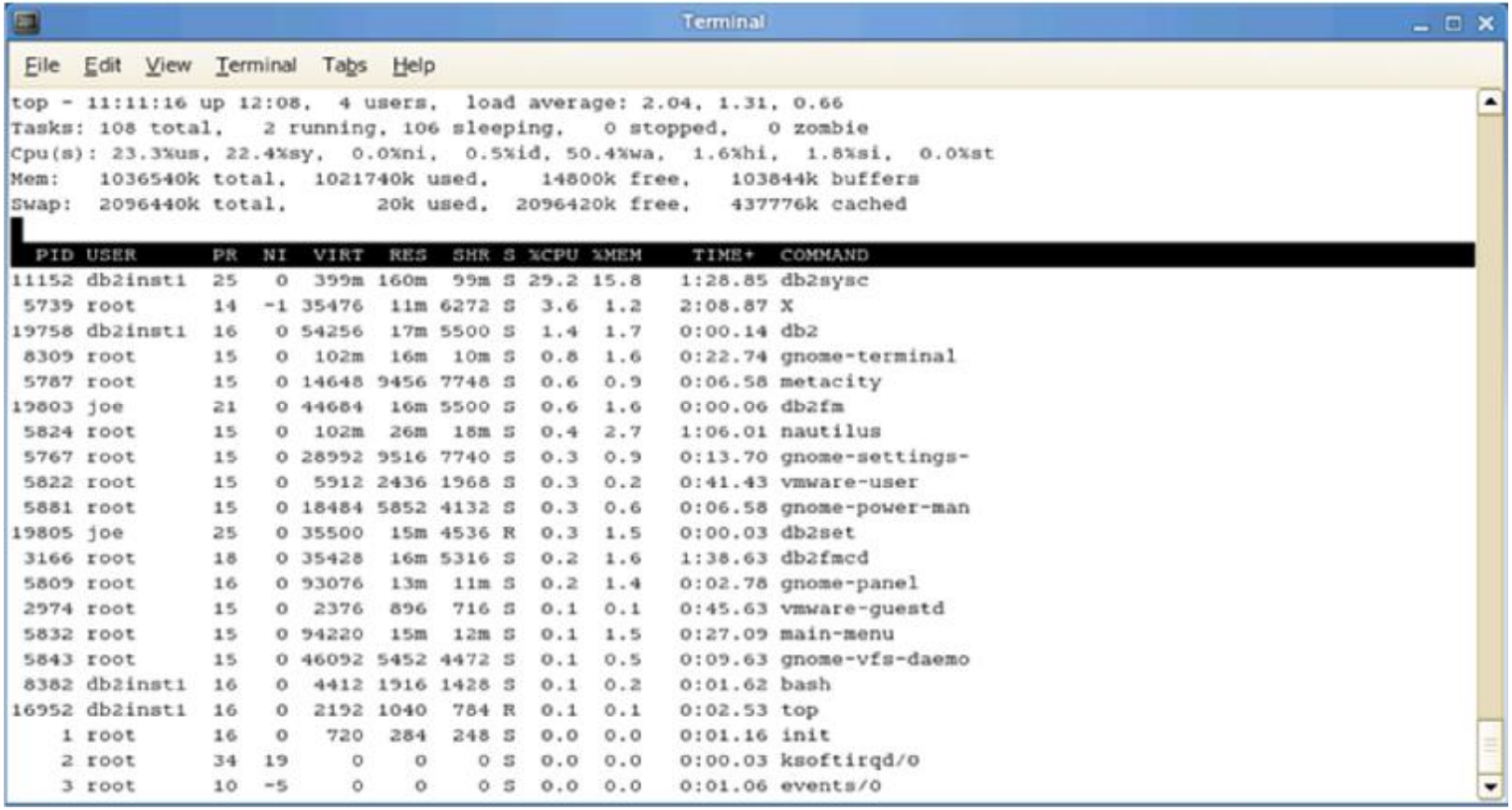
Network Utilization

Network Interface\Output Queue Length\nic name - is the number of packets in queue waiting to be sent. If there is a sustained average of more than two packets in queue, you should be looking to resolve a network bottleneck.

Network Interface\Packets Received Errors\nic name - packet errors that kept the TCP/IP stack from delivering packets to higher layers. This value should stay low.

System Monitoring tools - TOP

- **CPU consumption**
- **Configurable output**



PRESENTATION NOTES FOR PAGE 22

TOP Can be run on Unix based operating systems, the top command is a system monitor tool that produces a frequently-updated list of processes and the resources they are consuming. By default, the processes are ordered by percentage of CPU usage, with only the "top" CPU consumers shown. The top command shows how much processing power and memory are being used, as well as other information about the running processes. Some versions of top allow extensive customization of the display, such as choice of columns or sorting method.

The top command is useful for system administrators, as it shows which users and processes are consuming the most system resources at any given time.

It is also possible to redirect the output of top in a text file.

More System Tools

§ netstat
- – Monitor network activity, open ports

§ nmon
- – Free from sourceforge.net, Open Source
- – Comprehensive system monitoring
- – Formatted output (CSV) or Screen interface
- – Easy to make reports, trend performance data

§ vmstat/sar
- – Command line tools to monitor memory, paging
- – CPU usage by user, system, idle, wait times
  - • > 25% wait could indicate I/O bottleneck
  - • User CPU **à** compilation and execution of SQL
  - • System CPU **à** calls to OS kernel **à** are microcode and drivers current?

PRESENTATION NOTES FOR PAGE 23

netstat (network statistics) is a command-line tool that displays network connections (both incoming and outgoing), routing tables, and a number of network interface statistics. It is available on Unix, Unix-like, and Windows NT-based operating systems.

It is used for finding problems in the network and to determine the amount of traffic on the network as a performance measurement.

Parameters used with this command must be prefixed with a hyphen (-) rather than a slash (/).

-a : Displays all active TCP connections and the TCP and UDP ports on which the computer is listening.

-b : Displays the binary (executable) program's name involved in creating each connection or listening port. (Windows XP, 2003 Server only (not Microsoft Windows 2000 or other non-Windows operating systems))

-e : Displays ethernet statistics, such as the number of bytes and packets sent and received. This parameter can be combined with -s.

-f : Displays fully qualified domain names <FQDN> for foreign addresses.(only available on Windows Vista and newer OS'es)

-i : Displays network interfaces and their statistics (not available under Windows)

-n : Displays active TCP connections, however, addresses and port numbers are expressed numerically and no attempt is made to determine names.

-o : Displays active TCP connections and includes the process ID (PID) for each connection. You can find the application based on the PID on the Processes tab in Windows Task Manager. This parameter can be combined with -a, -n, and -p. This parameter is available on Microsoft Windows XP, 2003 Server (and Windows 2000 if a hotfix is applied[2]).

-p Windows and BSD: Protocol : Shows connections for the protocol specified by Protocol. In this case, the Protocol can be tcp, udp, tcpv6, or udpv6. If this parameter is used with -s to display statistics by protocol, Protocol can be tcp, udp, icmp, ip, tcpv6, udpv6, icmpv6, or ipv6.

-p Linux: Process : Show which processes are using which sockets (similar to -b under Windows) (you must be root to do this)

-P Solaris: Protocol : Shows connections for the protocol specified by Protocol. In this case, the Protocol can be ip, ipv6, icmp, icmpv6, igmp, udp, tcp, or rawip.

-r : Displays the contents of the IP routing table. (This is equivalent to the route print command under Windows.)

-s : Displays statistics by protocol. By default, statistics are shown for the TCP, UDP, ICMP, and IP protocols. If the IPv6 protocol for Windows XP is installed, statistics are shown for the TCP over IPv6, UDP over IPv6, ICMPv6, and IPv6 protocols. The -p parameter can be used to specify a set of protocols.

-v : When used in conjunction with -b it will display the sequence of components involved in creating the connection or listening port for all executables.

Interval : Redisplays the selected information every Interval seconds. Press CTRL+C to stop the redisplay. If this parameter is omitted, netstat prints the selected information only once.

/? : Displays help at the command prompt. (only on Windows)

[edit] Statistics provided

Netstat provides statistics for the following:

Proto - The name of the protocol (TCP or UDP).

Local Address - The IP address of the local computer and the port number being used. The name of the local computer that corresponds to the IP address and the name of the port is shown unless the -n parameter is specified. If the port is not yet established, the port number is shown as an asterisk (*).

Foreign Address - The IP address and port number of the remote computer to which the socket is connected. The names that corresponds to the IP address and the port are shown unless the -n parameter is specified. If the port is not yet established, the port number is shown as an asterisk (*).

State - Indicates the state of a TCP connection. The possible states are as follows: CLOSE_WAIT, CLOSED, ESTABLISHED, FIN_WAIT_1, FIN_WAIT_2, LAST_ACK, LISTEN, SYN_RECEIVED, SYN_SEND, and TIME_WAIT. For more information about the states of a TCP connection, see RFC 793.

[edit] Examples

To display the statistics for only the TCP or UDP protocols, type one of the following commands:

netstat -sp tcp

netstat -sp udp

To display active TCP connections and the process IDs every 5 seconds, type the following command (works on Microsoft XP and 2003 only, or Windows 2000 with hotfix):

netstat -o 5

Mac OS X version

netstat -w 5

To display active TCP connections and the process IDs using numerical form, type the following command (works on Microsoft XP and 2003 only, or Windows 2000 with hotfix):

netstat -no

To display all ports open by a process with id pid

netstat -ao | find "pid"

DB2 Monitoring Tools – DB2TOP

§ DB2TOP

- Provides a unified, single-system view of a multi-partition database or single-partition database on the AIX®, Linux, HP-UX, and Solaris operating systems
- Can be run in interactive mode or in batch mode

PRESENTATION NOTES FOR PAGE 24

Its good to know what to monitor, how to delve into specific areas activity and resolve problems but it can be much easier to make assessments with a good tool to help you monitor your database system. There are many tools on the market that can provide very powerful and valuable utilities and monitoring functions. IBM is featuring OPTIM based solutions, an integrated approach to ensuring database system performance over the course of an applications life cycle.

There are some free tools available for you to use, they are generally less flexible and integrated than a tool you might purchase… but they are FREE.

Let's take a brief look at two of them db2top and Performance Expert.

DB2 Monitoring Tools – DB2TOP

- **View delta or cumulative snapshot counters**
- **Monitor interactively or collect data for analysis on:**
  - Database (d)
  - Tablespace (t)
  - Dynamic SQL (D)
  - Sessions (l)
  - Bufferpool (b)
  - Lock (U)
  - Table (T)
  - Bottlenecks (B)

PRESENTATION NOTES FOR PAGE 25

The db2top command provides a unified, single-system view of a multi-partition database or single-partition database on the AIX®, Linux and Solaris operating systems. It can be used to quickly identify global problems, or specific database partition problems in the system. By combining snapshot information from each database partition, the utility can provide a dynamic real-time view of a running DB2 system.

You will practice some simple tasks with db2top in one of the lab exercises.

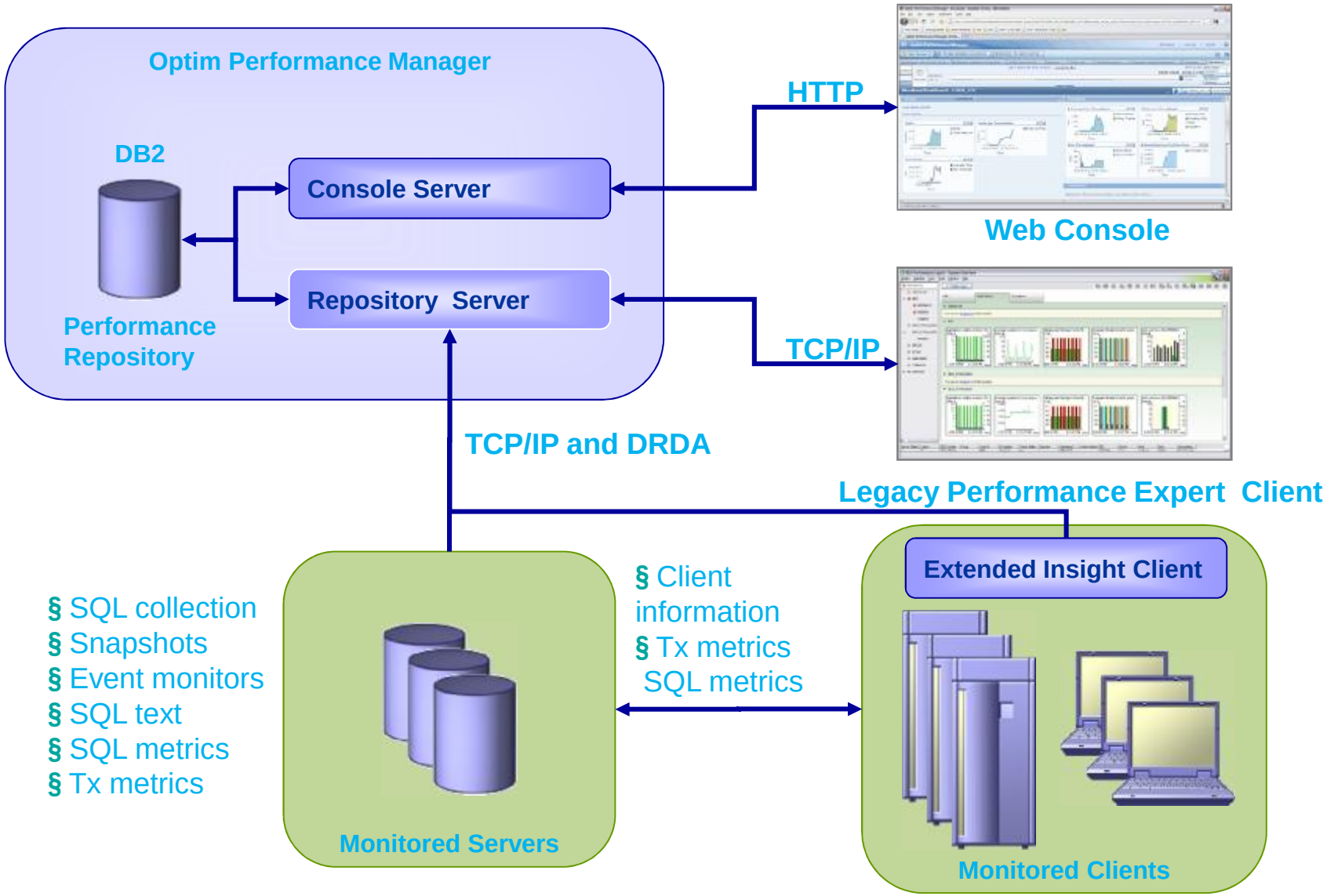
DB2 Monitoring Tools – IBM Optim Performance Manager

- This tool has a web-based interface to view system health at any time from any location
- It will help you prevent problems by monitoring performance indicators for emergent problems with easy to understand dashboards
- It enables DBA to plan for business growth and also access historical data to generate key reports
- It has out of the box DB2 and application monitoring for:
  - SAP™
  - Cognos™
  - DataStage®
  - Java® (WebSphere®)
  - CLI applications
- With this tool, identify, diagnose and solve problems quickly, pinpointing them in minutes instead of days

PRESENTATION NOTES FOR PAGE 26

This tool has a web-based interface to view system health at any time from any location. It will help you prevent problems by monitoring performance indicators for emergent problems with easy to understand dashboards. It enables DBA to plan for business growth and also access historical data to generate key reports. It has out of the box DB2 and application monitoring for SAP™, Cognos™, DataStage®, Java® (WebSphere®) and CLI applications. With this tool, identify, diagnose and solve problems quickly, pinpointing them in minutes instead of days.

Architecture Overview (OPM 5.1)



PRESENTATION NOTES FOR PAGE 27

Let me give you an overview of the architecture as a backdrop to the discussion

InfoSphere Optim Performance Manager consists of a console server and a repository server component.

The Repository server collects the data from the monitored database and stores it in a DB2 repository database. It also support aggregation and pruning of the performance data in the repository..

The Console server reads the data from the repository database and displays it in the Web UI on users request.

[click]

For monitoring DB2 V9.7 databases the repository server connects to the monitored database and issues SQL statements to collect the in-memory metrics. Some metrics are also still collected via snapshots. The user specifies how often to collect the data. Additionally depending on the configuration some event monitors may be started on the monitored database, for example to collect locking information.

For monitoring DB2 V9.1 or DB2 V9.5 database the repository server uses instance attachments to get the snapshot data in addition to connections for starting event monitors.

[click]

The main user interface is the web UI. It provides a series of dashbaords for viewing all the data in the performance repository.

[click]

There is also a legacy Performance Expert client. It is typically used by customers who have migrated from DB2 Performance Expert, the predecessor product to InfoSphere Optim Performance Manager. [Or by users monitoring DB2 9.1 or 9.5 database who want to retain long term data in a Performance Warehouse for reporting.]

[click]

When deploying Extended Insight, users can optionally deploy the Extended Insight client (aka Data Tools Runtime Client) on the monitored client envirnments. These are installed on the systems where your database applications are runningand support Java, CLI and .Net applications clients. The Extended Insight clients send data about executed SQL statements and transactions to the repository server.

In parallel, the repository server collects data server execution data for SQL and transactions (9.7 only) from the monitored database and maps them to the data sent by the clients. This way you get a complete picture about the transaction and statement response times of your application. To collect the Extended Insight server data, Performance Manager starts the package cache and unit of work event monitors on the monitored database to get the SQL and transaction details in addition to the listing of all statements within the package cache. Note that EI client and server data collection is independnet and you can activate one without the other.

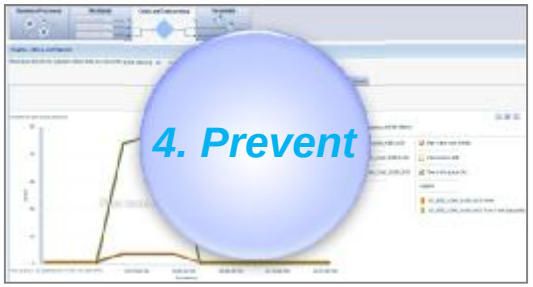
Monitoring And Optimizing Performance – OPM

Get early warning of potential problems

Diagnose database problems with resource specific dashboard

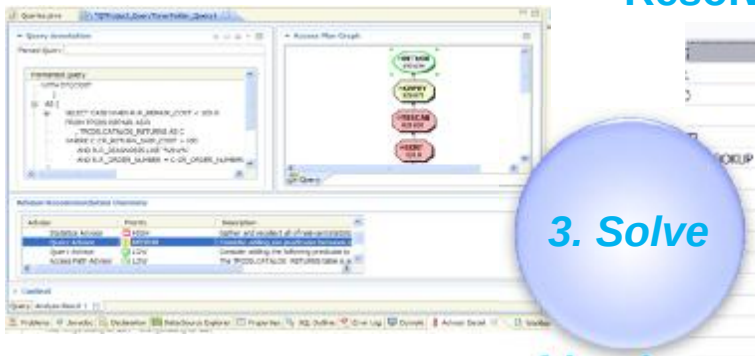
InfoSphere Optim Performance Manager
InfoSphere Optim Configuration Manager

Prevent problems



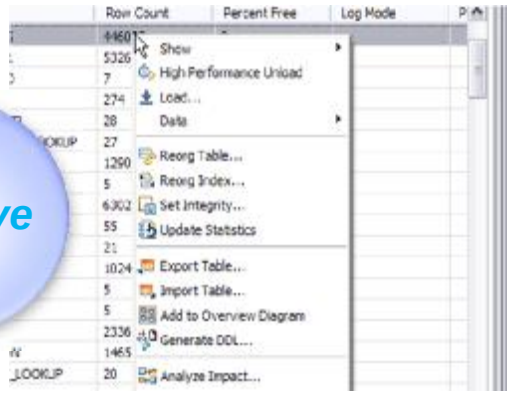
DB2 Workload Manager

Resolve query problems



InfoSphere Optim Query Workload Tuner
InfoSphere Optim pureQuery Runtime

Resolve database problems



Data Studio

PRESENTATION NOTES FOR PAGE 28

The Performance Management solution is composed of several offerings as show here. Performance Manager provides the core monitoring infrastructure and metrics collection that provides the basis for identifying that you have a problem. You can drill down in to detailed resource-specific dashboards to isolate problems and see what may have changed in the configuration with integration to Configuration Manager.

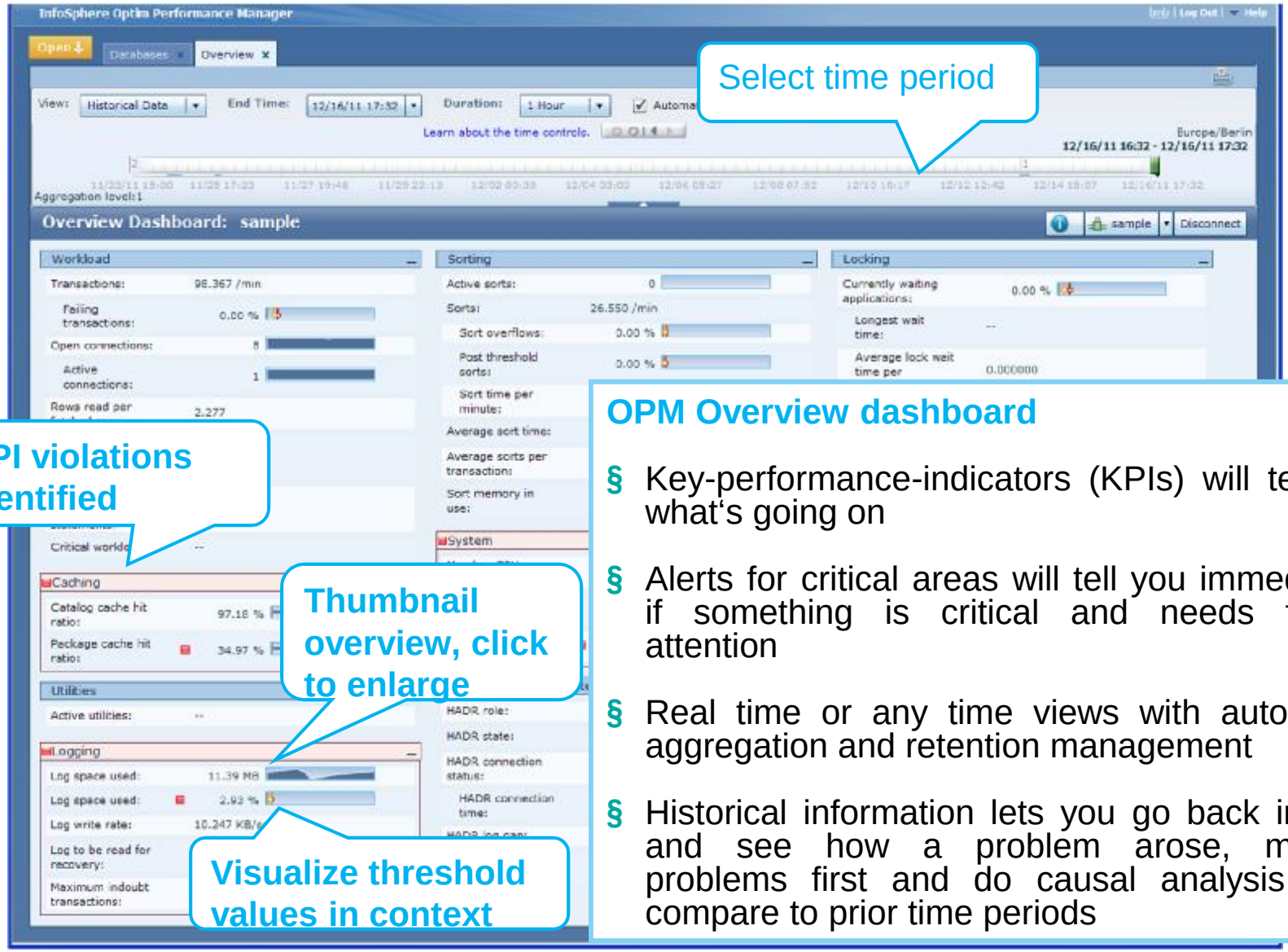
Data Studio, included with DB2, provides the means to make database changes, while Query Workload Tuner provide the expert advice on how to improve queries or workloads.

pureQuery Runtime supports improvements in performance, performance stability, and security. And all of the tools together with DB2 Workload Manager support preventative actions to keep systems running smoothly.

The core of the Performance Management solution though is InfoSphere Optim performance Manager and InfoSphere Optim Query Workload Tuner and those are the offerings we'll focus on

[Optional: We will look at each of these in detail]

Database Overview Dashboard At A Glance



OPM Overview dashboard

- Key-performance-indicators (KPIs) will tell you what's going on
- Alerts for critical areas will tell you immediately if something is critical and needs further attention
- Real time or any time views with automated aggregation and retention management
- Historical information lets you go back in time and see how a problem arose, mitigate problems first and do causal analysis later, compare to prior time periods

PRESENTATION NOTES FOR PAGE 29

Rather than a cross-database view, you can also get an overview for a specific database on the Overview Dashbaod.

Here you can see key performance indicators for he critical database activities including workload, sorting, locking, logging, I/O etc. Key performance indicators (KPIs) will tell you what‘s going on in each category. Thumbnail graphs give you mini-trends and are easily enlarged to get a better view. A grid is also available with all the details.

Alerts for critical areas will tell you immediately if something is critical and needs further attention. Title bars change colors and thresholds violations are shown. You can also see the warning and critical threshold levels in context and even adjust them for fine tuning.

Real time or any time views with automated aggregation and retention management. Available as part of DB2 9.7 monitoring, real time views allow you to see current data for problem determination. You can get either instant refresh of monitoring data or specify a sub-minute refresh rate. In addition, Performance Manager automatically performs metrics aggregation at 15 minute, 1 hour, and 1 day intervals and supports automated retention manage at each aggregate level making it easy and cost effective to save data long term.

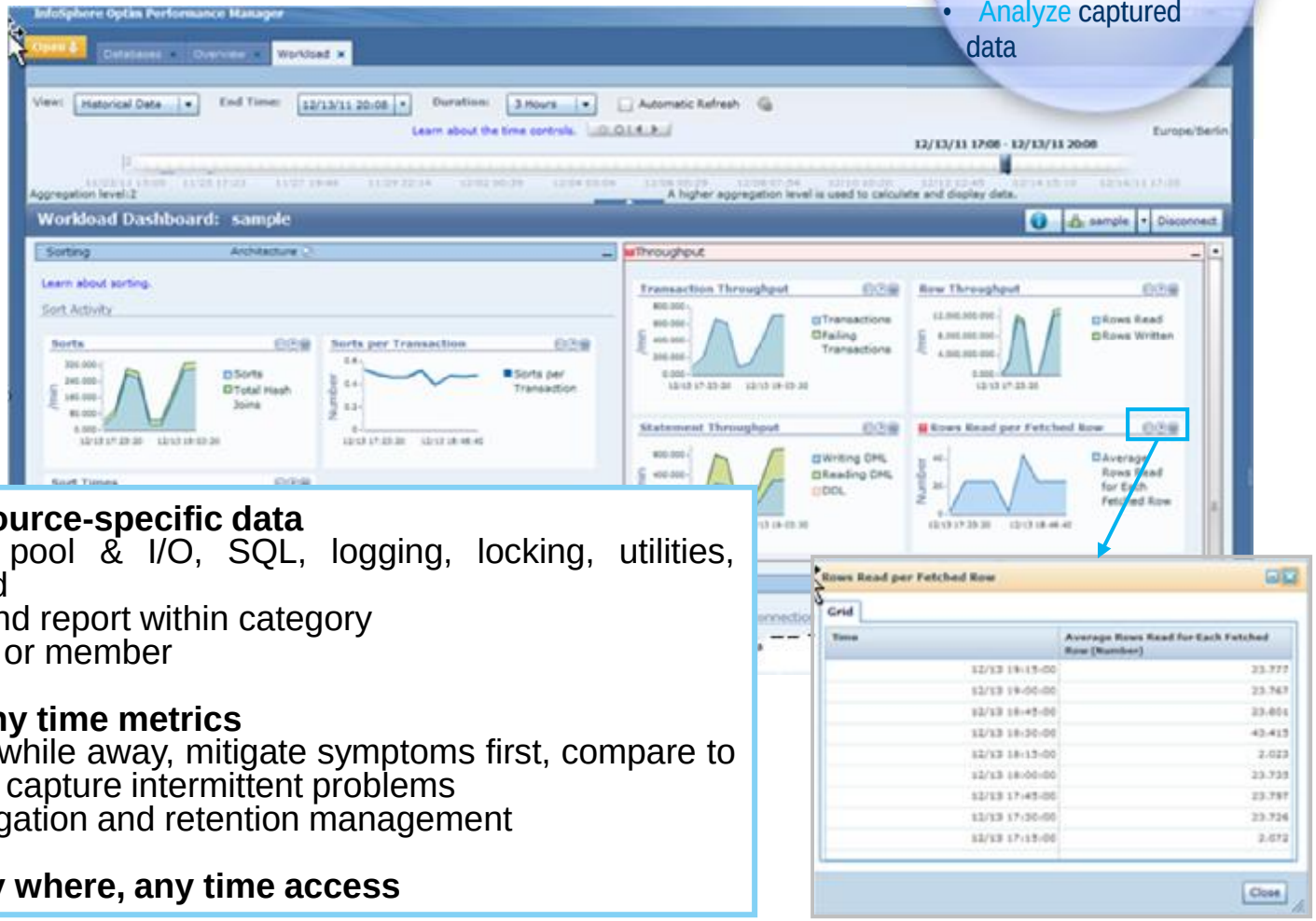
You can go back to any time period stored in the repository by just adjusting the time slider to the period of interest. Compare current values to prior values to see if this is out of the norm. Mitigate symptoms first and defer root cause analysis with historical data. Widen the viewed interval to hours, days, or weeks to instantly view metric trends on dashboards to spot spikes, emergent problems, or significant changes to steady state.

Guided Problem Solving Approach

Drilldown to diagnose alerts to find the problem

Diagnose

- Drill down into problem detail and related content
- Analyze captured data



- **Drill down into resource-specific data**
  - Memory, buffer pool & I/O, SQL, logging, locking, utilities, system, workload
  - List, filter, sort, and report within category
  - View by partition or member
- **View real time to any time metrics**
  - What happened while away, mitigate symptoms first, compare to historical values, capture intermittent problems
  - Automatic aggregation and retention management
- **Browser-based any where, any time access**

PRESENTATION NOTES FOR PAGE 30

InfoSphere Optim links alerts to the detailed diagnostic data following a natural flow.

Drill into contextual, resource-specific data for performance analysis. Dashboards are available for all key resources including memory, buffer pool and I/O. SQL processing etc. You can list, filter, sort, analyze, and report on database resources by category. Drill through contained and container relationships. View connection, SQL, and utility activity across the database or drill into the detail for a specific one. Identify most costly SQL either from single executions or in aggregate. You can easily drill into partition or member level detail as well.

[Note: much of this information is duplicated from the Database Overview dashboard, but I like to emphasize these points]

View performance data from real time to any time. Like the Database Overview dashboard, all dashboards let you view real time data or historical data in the performance repository whether now, yesterday, last week, or last year. A sliding historic scale not only allows for the size of an interval, from 1 minute to any period, but it allows that same interval to be moved by sliding the displayed interval along the time toolbar. As mentioned, you can

Check what happened while you were away or analyze a reported slow down that occurred over the weekend

Capture details for intermittent problems

Mitigate impactful symptoms first and defer root cause analysis to a later time

Compare current values to prior values to see if this is out of the norm

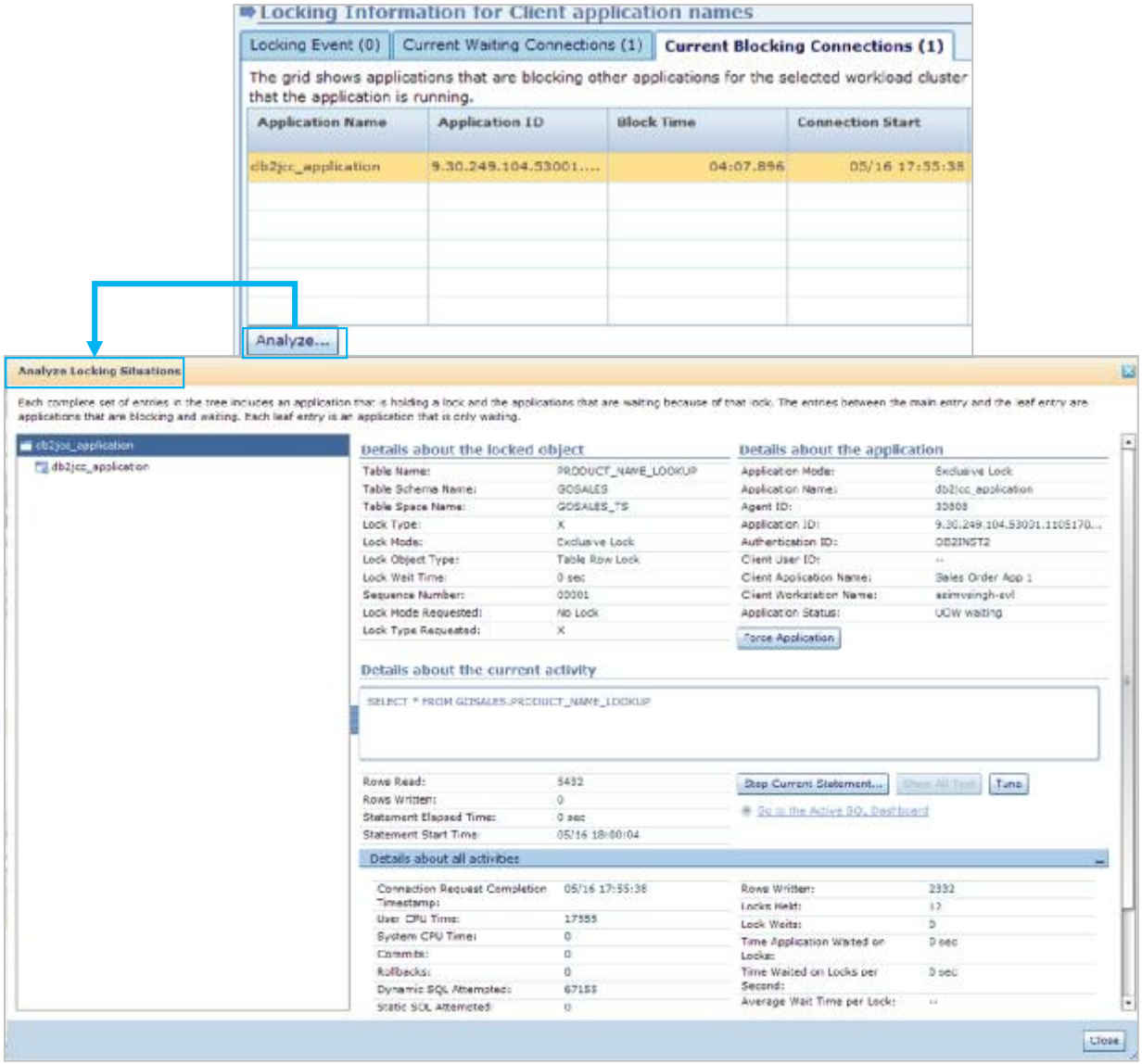
Instantly view metric trends on dashboards, such as shown here, to spot spikes, emergent problems, or significant changes to steady state. Use the chart controls highlighted to expand the charts to zoom out or access grid views for details.

Automatic rolling aggregation with user-defined retention provides data necessary for comparisons, trend detection, capacity planning, and service reporting. Performance Manager automatically aggregates at the 15 minute, 1 hour, and 1 day levels and you can specify retention periods at each level. Performance Manager automatically determines and displays the aggregate level used for the display on the time slides (left under time slider this is showing aggregation level 2 i.e.15 minute aggregate. Agregation level 1 is raw data coming from DB2 which is itself pre-aggregated)

Dashboards are all browser-based so you can access them from anywhere you have a browser and access to your network.

Analyze The Lock Conflict

- **View**
  - Lock holder
  - Lock waiters
  - Locked object
  - Application details
  - SQL statement
- **Take action**
  - Force application

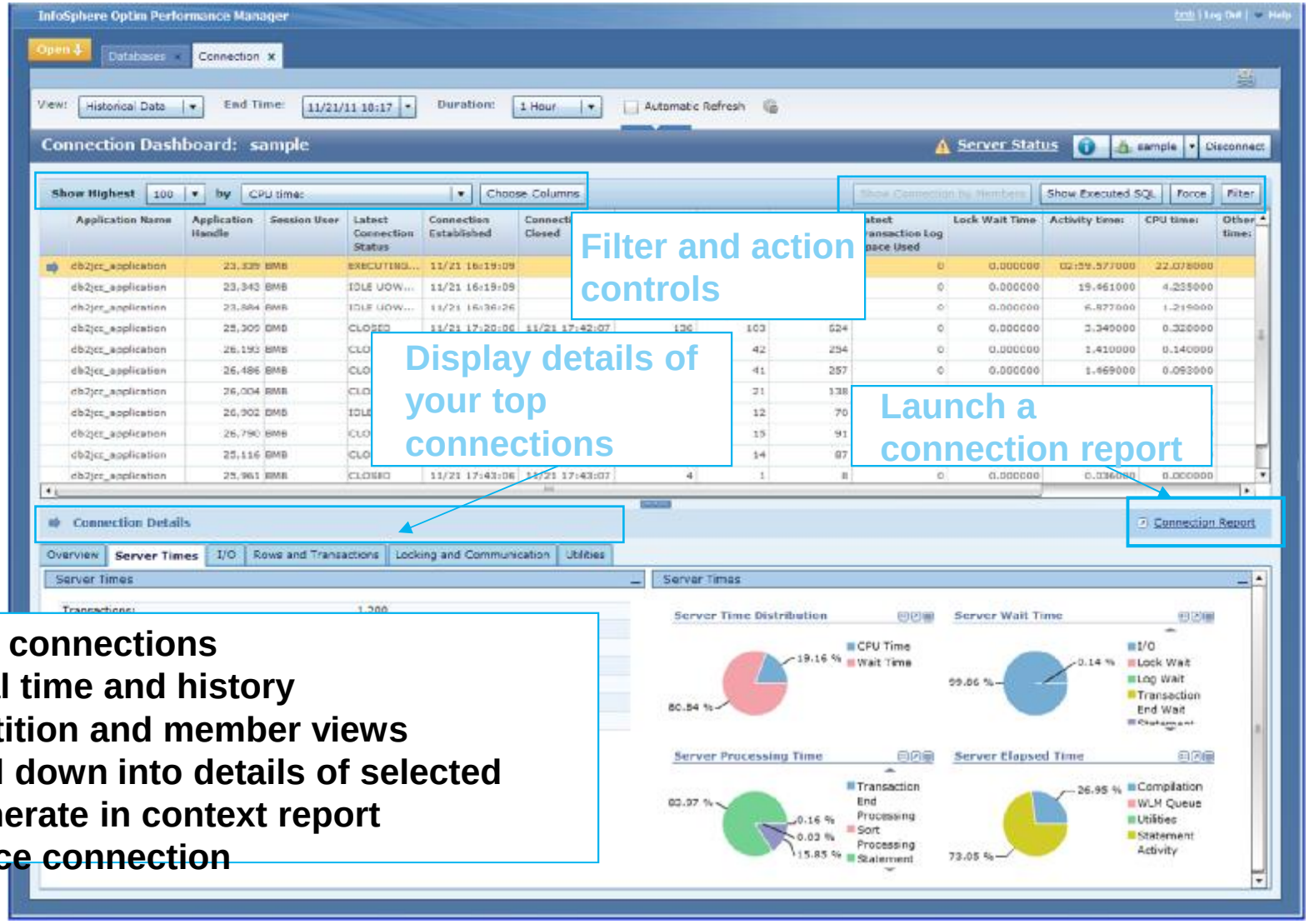


PRESENTATION NOTES FOR PAGE 31

The locking analysis provides details on the locking situation. Clearly view lock holders and waiters, the locked object, details about the application, the SQL statement involved and so forth. You can also take action to break the conflict by forcing the application directly from here.

This has provided an example of how workflow works within InfoSphere Optim Performance Manager.

Analyze Connections



- List connections
- Real time and history
- Partition and member views
- Drill down into details of selected
- Generate in context report
- Force connection

PRESENTATION NOTES FOR PAGE 32

The Connection dashboard shows the top N current connections (in real-time mode) or the connections for a selected timeframe. You can select from various top criterias to determine the which connection view you want to see, e.g. You might view connections by top CPU time or by most rows read.

Filter and control actions shown near the top of the dashboard allow you to further filter connections based on any column or to change the columns shown (adding, deleting or re-ordering columns).

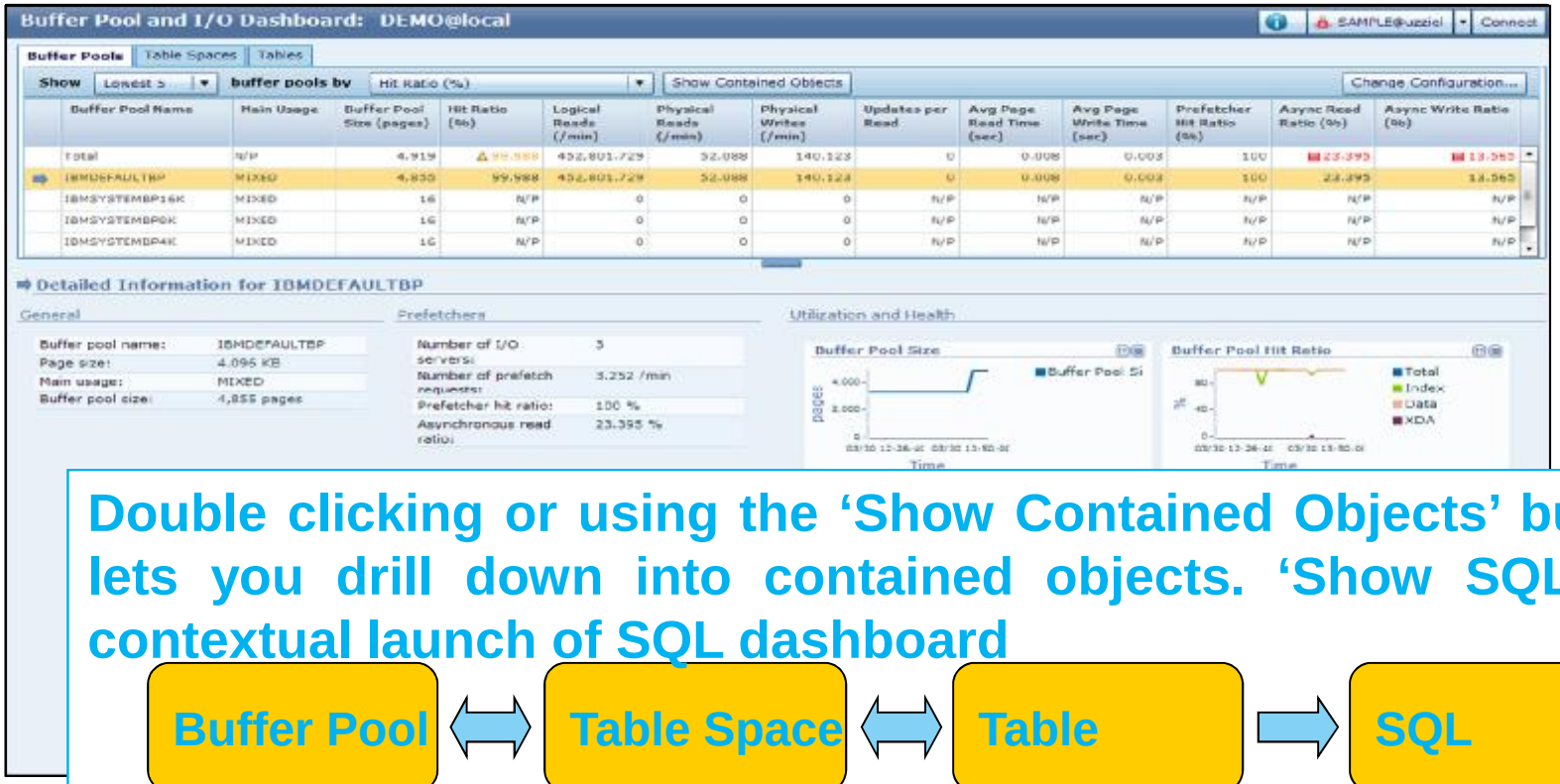
Most dashboards have this format displaying a grid that provides a list at the top, here are list of connections. Once you select in item in the list, then the lower part of the dashboard provides detail on the selected item. So for this example, we have displayed the top 100 connections by CPU time and selection the top connection representing the connection consuming the most CPU. Details about the connection are shown in the tabbed area below where you can display details about the various areas, like times, I/O, locking, row and transaction details, application details, and Workload Manager.

From the dashboard you can also launch in-context connection report.

Use the ‚Show executed SQLs‘ button (upper right) to list the SQL statements executed by the connection. This button opens the SQL dashboard in the Top Execution mode filtered by the application id of the selected connection.

You can also force a connection from here or generate a connection report directly from this dashboard.

Analyze I/O



Double clicking or using the 'Show Contained Objects' button lets you drill down into contained objects. 'Show SQL' for contextual launch of SQL dashboard

Buffer Pool ⟺ **Table Space** ⟺ **Table** ➡ **SQL**

- **Check buffer pools, table spaces, and tables**
  - Identify hot objects that need dedicated buffer pools
  - Check whether buffer pools are appropritely sized
  - Check the disk space and container definition of table spaces
  - See what SQL is accessing a table

PRESENTATION NOTES FOR PAGE 33

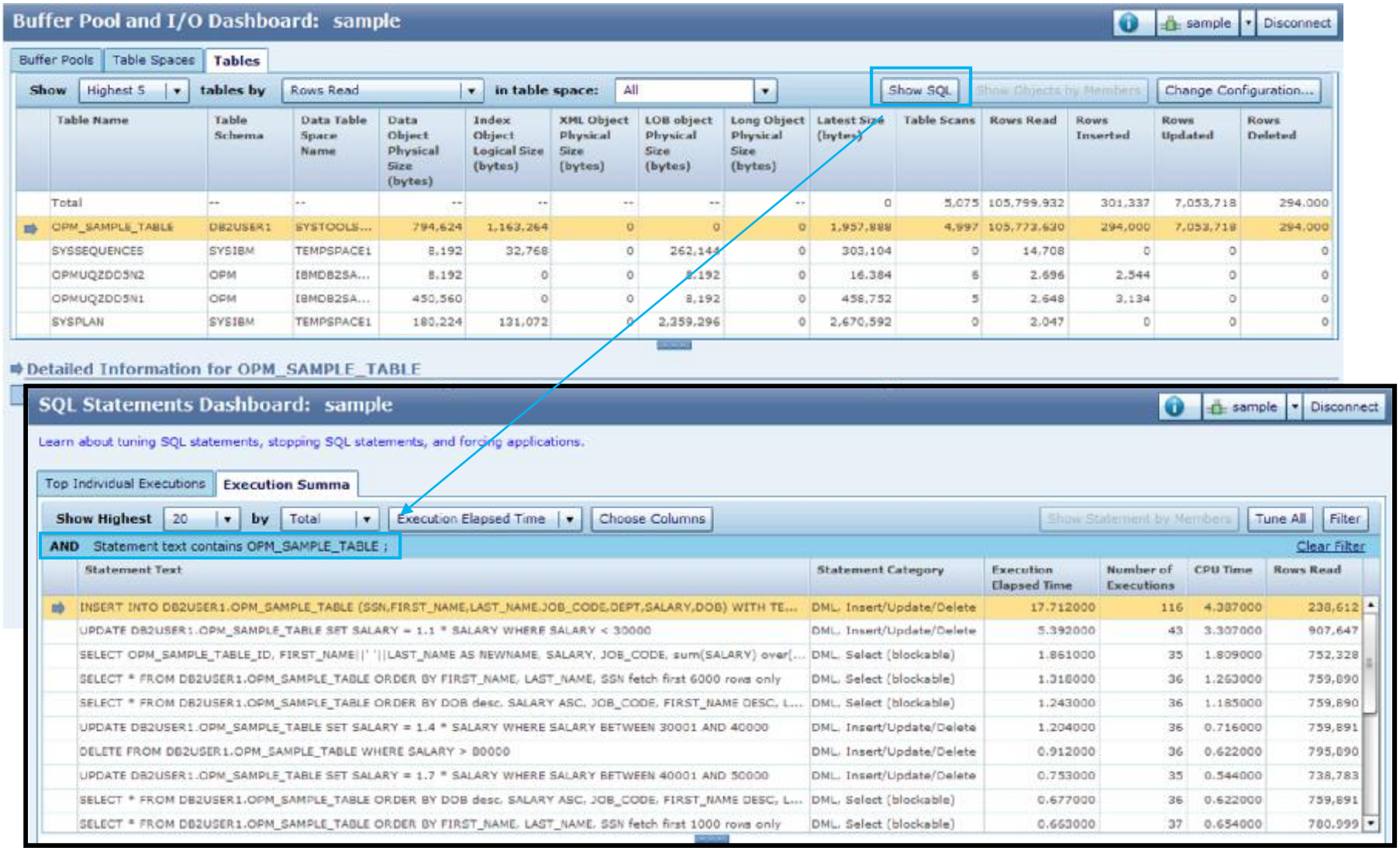
The Buffer Pool and I/O Dashboard is available to analyze I/O activity and efficiency. You can identify is hot objects that need dedicated buffer pools, check whether buffer pools are appropriately sized based on it ratios, check on table space utilization, etc.

Similar to other dashboard, you can select the display criteria, for example higher 5 bufferpools by rows read to identify hot buffer pools or lowest 5 by hit ratio to check of adequate sizing. Selecting a specific bufferpool gives you the detail about the buffer pool in the lower half of the dashboard to see related buffer pool metrics, activity, and trends e.g. hit ratio, logical and physical I/O, read and write activity and more.

Its tabs provide easy access to view metrics for buffer pools, table spaces, and tables. The Show Contained Objects buttons (upper middle) allow drill down through from container to contained objects, that is buffer pools contain tables paces and table spaces contain tables.

You can also drill through to the SQL dashboard to see SQL filtered by table name.

From A Table To The SQL Using The Table



PRESENTATION NOTES FOR PAGE 34

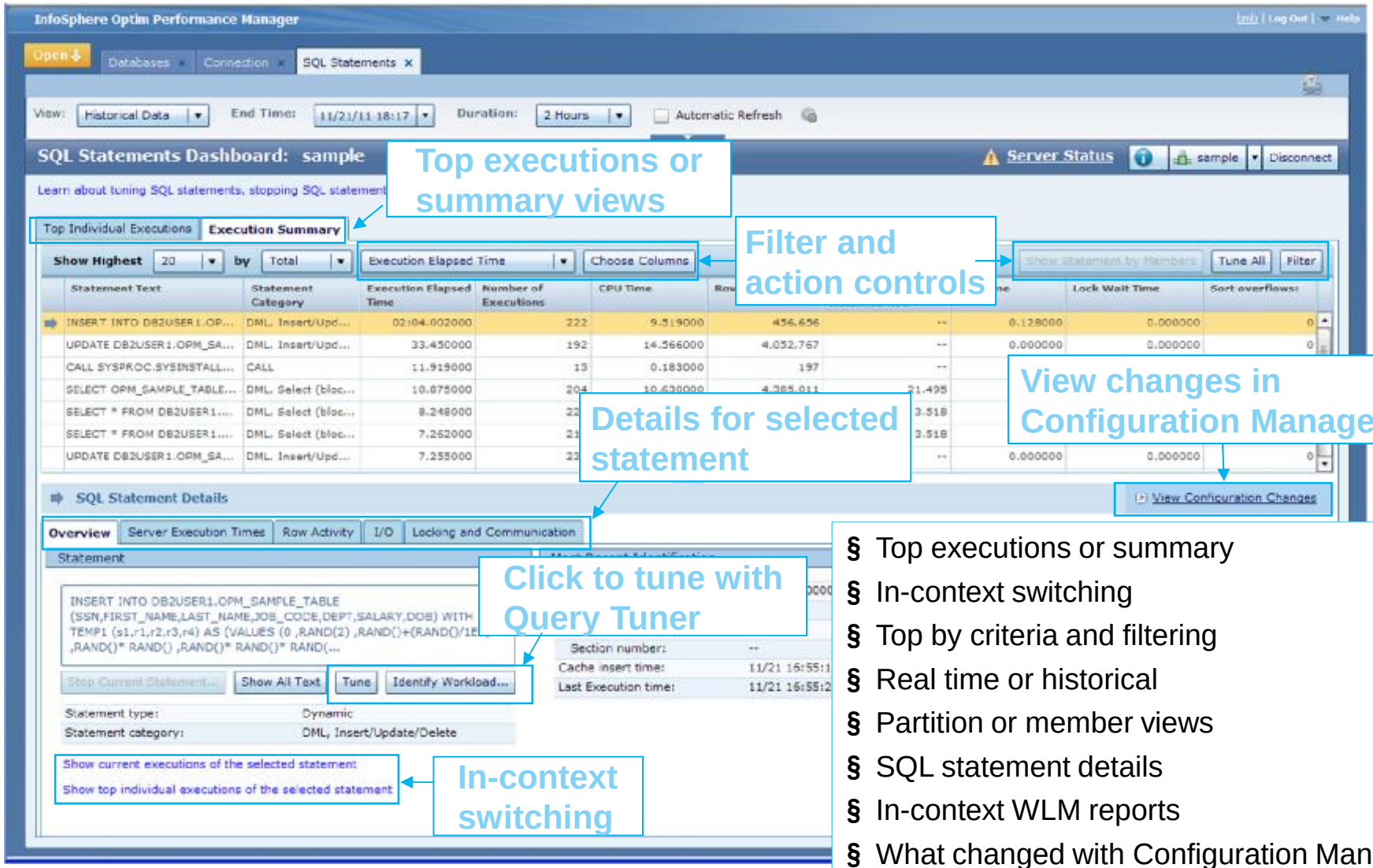
[Optional slide]

For example, if you see hot tables in the Buffer Pool and I/O Dashboard that show a high number of rows read, then you show the contained tables, select the table with the highest number of rows read, and drill down to the SQL statements that use that table using the Show SQL button.

The Execution Summary view of the SQL Dashboard opens with a filter is set using the table name that you selected before. All SQL statements having the table name in the statement text are shown on the SQL Dashboard and can be analyzed further.

[Note: this does not access ALL SQL access the table as it does not identify SQL access through aliases such as Views or MQTs]

Analyze SQL



- Top executions or summary
- In-context switching
- Top by criteria and filtering
- Real time or historical
- Partition or member views
- SQL statement details
- In-context WLM reports
- What changed with Configuration Manager
- Tune with Query Tuner

PRESENTATION NOTES FOR PAGE 35

The SQL Dashboard provides detailed analysis of SQL executing against a database.

The SQL dashboard offers 2 views: Top executions and Summary

The Top Executions view shows single executions of statements order by a specified criteria, e.g. CPU time, rows read. (This is the same as the Active SQL dashboard in OPM 4.1).

The Summary view shows aggregated information over all executions of a statement also ordered by a specified criteria.

You can easily switch between views in context (in lower left of image). For example, you can select a statement in the Summary view and switch in context to the executions view to see each execution of the statement within the selected interval. Or from the Executions view, you might see the same statement repeatedly, select one, and switch to the Summary view to see the aggregated costs over the selected time period.

You can further customize the view by adding, removing, or reordering columns in the grid and sorting or filtering on any column value. (Note to presenter: column width, addithing/deletions, and ordering are saved. Filtering is not)

Like all dashbaords, you can use this dashboard in realtime mode to monitor current statement executions or you can select a timeframe in history to see what statements were executed during a past period.. And you can view details for the entire database, for specific partitions or members, or for partition categories.

When you select a statement in the upper grid, you get more details for the statement in the lower part of the dashboard shown in different tabs, e.g. execution times, row activity, locking, I/O and so on.

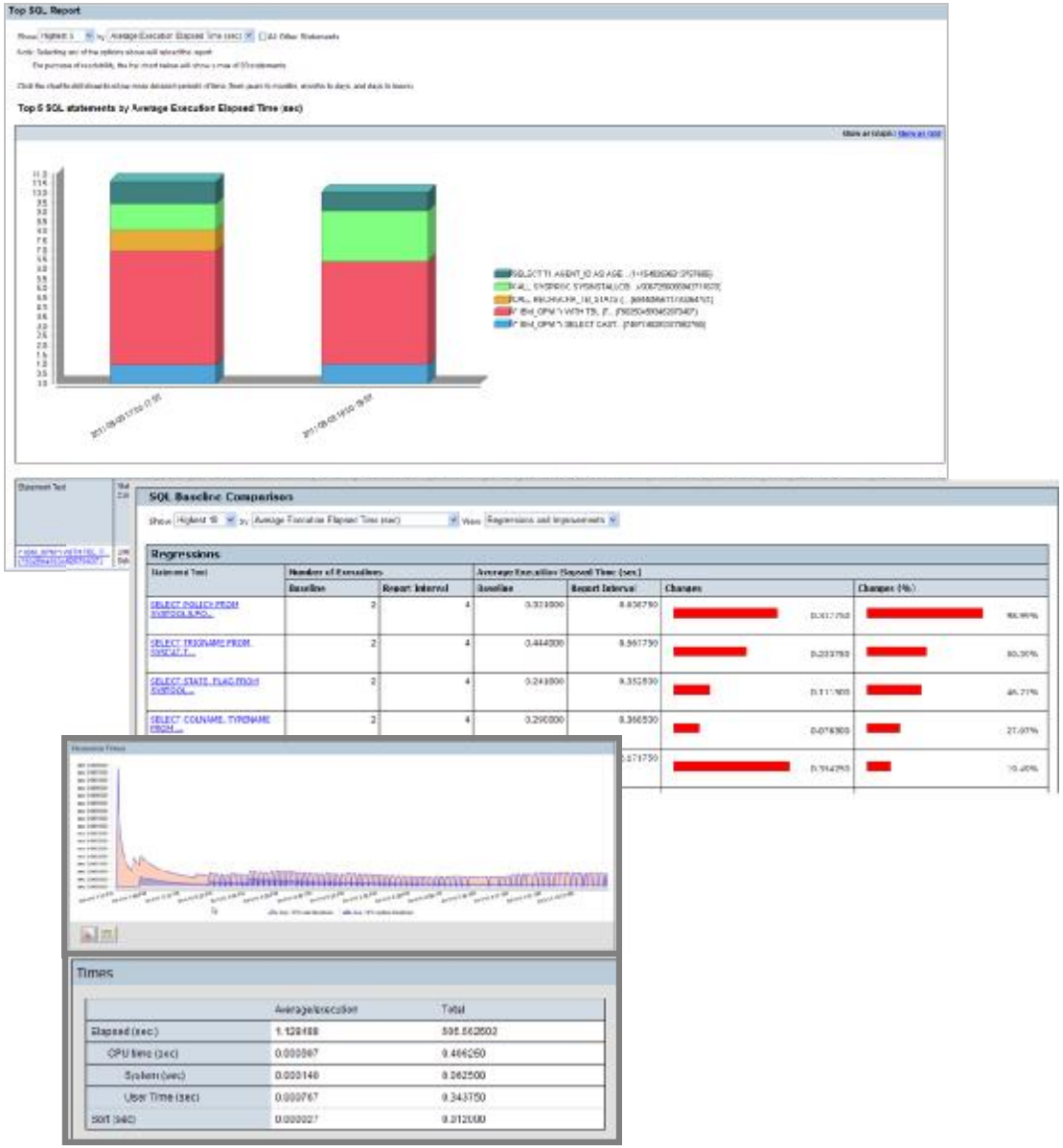
Additionally you can launch-in context reports, e.g. You can generate WLM reports about the service class or workload this statement is running on from the Top Executions view, Application tab.

Frequently when performance changes, the first question you ask is “What changed?” Click View Configuration Changes to launch InfoSphere Optim Configuration Manager in context to view changes relevant changes so DBAs can quickly determine whether recent changes may be the underlying cause of performance issues.

And if you identify one or more SQL statements that you'd like to Tune, you can click the Tune or Tune All buttons to begin tuning with InfoSphere Optim Query Workload Tuner or simply format the SQL or view the access plan in Data Studio.

Reporting

- **General**
  - Performance Overview
  - Configuration
  - Connection
- **SQL analysis**
  - Top SQL
  - Top Package
  - SQL comparison
- **Resource analysis**
  - Disk space consumption
  - Table
  - Workload Manager



PRESENTATION NOTES FOR PAGE 36

Performance manager includes out-of-the-box reporting against the retained performance metrics. Users can report over any timeframe and can schedule, browse, print, export, present, and share their analysis and results. Each interactive report is like multiple reports in one. Reports typically provide a graphical view with zoom in and drill down capabilities to additional detail or through to a contextually launched SQL report.

Reports can be broadly categorized into three main area: General reporting, SQL analysis, and Resource consumption

General reports include:

The Performance Overview Report which shows the overall health of the monitored database for all major metrics. Users can view the complete database system or only parts of it.

The database or database manager configuration report contains details about capacity management, communications, logging and recovery, and database management including detail about partition or member level configuration and configuration changes.

The Database Connection Report provides an overview of the active database connections for a given time frame. The report displays key performance indicators, such as lock wait times, physical and logical reads and writes, and other connection statistics. This report can identify applications that are not performing well or applications that are causing problems in other database applications. DBAs can drill down into a specific connection to view complete identification details, timing information, SQL activity, locks, cache, buffer pool, sorts, and agent-related activity.

For SQL Analysis there are 3 main reports:

The Top SQL Report shows top resource consumers, both long running large queries and frequently run short queries. You can select the criteria by which the data is organized and then drill into specific statements, see response time patterns in histograms or performance trends. When you identify queries that need to be tuned, you can launch Query Tuner right from the report.

The Top SQL Package Report allows you to identify top resource consumers by package name in a specified time interval. DBAs can look at a graphic representation of the workload by day to identify heavy duty, critical, or rogue packages. Then you can select a specific package and drill through to the SQL report for additional detail.

The SQL Baseline Comparison Report lets you compare top SQL statements in the same database regarding the maximum improvement, or regression, or both, within two time frames. Users can drill down for a detailed analysis of a specific SQL statement. The report includes the complete statement text, general statement information, response time analysis, sort performance, I/O activity, and buffer pool activity. Similarly, you can drill through to the SQL report for additional detail.

For analyzing resources there are the Disk Space Consumption Report, Tables usage report, and Workload manager report

The Disk Space Consumption report provides an overview of the current disk space usage by table space. You can analyze information about growth rates to plan for future disk space requirements or table space configuration changes. You drill down into the details of a given table space, such as configuration, containers and tables, container layout, and data skew.

The Table Usage Report lets you identify "hot" tables or fast growing tables that might cause disk contention or that might need reorganization.

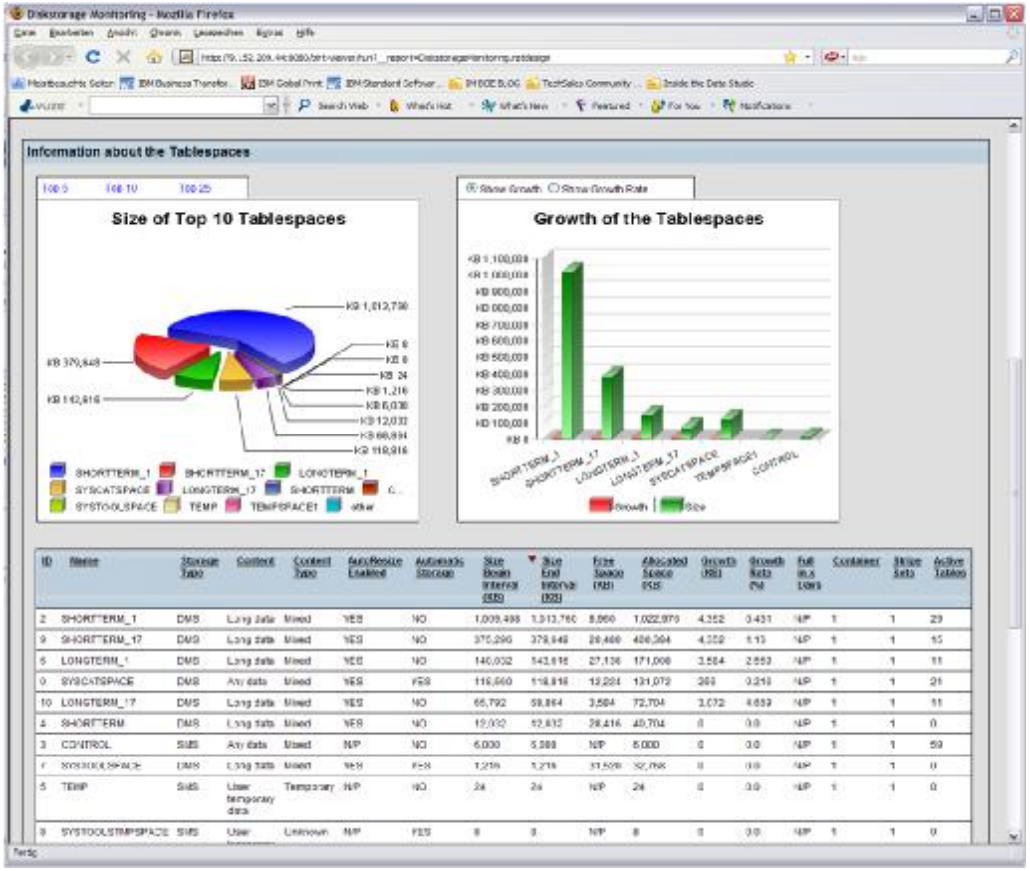
The Workload Manager Reports provide detail on workload configurations and drill through to specific workloads or Service SuperClass, SubClass, and Work Action Sets. They include details on queue time, execution time, lifetime, and histograms for advanced tuning of Workload Manager configurations.

Be Proactive. Focus On Prevention

Prevent problems by leveraging historical information

Prevent

- Monitor and analyze historical trends for planning
- Auto manage workloads



- Tune proactively
  - Identify heavy hitter SQL
  - Get expert advice
  - Compare results
- Add value to developers
  - Identify hot spots for developers
  - Give them tuning advice
- Plan for growth
  - Assess workload and storage growth trends
- Allocate resources by business priority
  - Ensure critical work has resource preference
  - Improve server utilization
  - Protect data server from overload

PRESENTATION NOTES FOR PAGE 37

Finally, the IBM Performance Management solution helps you be proactive and focus on problem prevention.

Tune queries and workload proactively. Rather than wait for service to degrade, get a report to identify high cost SQL, get actionable, expert advice to improve them, and deploy and validate performance improvements.

Help get a focus on SQL performance in the development cycle. The Performance Management solution helps developers identify performance SQL hot spots in their application and gives them the ability to improve their queries and skills before application deployment

IBM's solution helps you plan for growth by analyzing workload and storage growth trends.

And our solution helps you allocate resources by business priority. DB2 Workload Manager helps to assure that critical work gets resource preference, it helps to improve server utilization, and protects the data server from being overloaded.

IBM's solution helps new users to get started with Workload Manager best practices with a single click. Our research shows that even simple deployments in warehouse environments can yield significant value.

Summary

§ In this Module you learned about:
- Monitoring is a vital activity for the maintenance of the performance
- Choose the monitoring interfaces: monitoring framework, snapshot monitoring, and event monitoring
- Monitor DB2 metrics and system metrics
- Leverage tools like Optim Performance Manager

PRESENTATION NOTES FOR PAGE 38

The next steps…

PRESENTATION NOTES FOR PAGE 39

The Next Steps…

- Complete the Hands on Lab for this module
  - Log onto SKI, go to “My Learning” page, and select the “In Progress” tab.
  - Find the module
  - Download the workbook 10302_WB1_DB2_Performance_Monitoring_Essentials.pdf and the virtual machine image 10300_VM1_DB2_PerformanceMonitoringAndTuning_10.5.part#.rar
  - Follow the instructions in the workbook to complete the lab
- Complete the online quiz for this module
  - Log onto SKI, go to “My Learning” page, and select the “In Progress” tab.
  - Find the module and select the quiz
- Provide feedback on the module
  - Log onto SKI, go to “My Learning” page
  - Find the module and select the “Leave Feedback” button to leave your comments

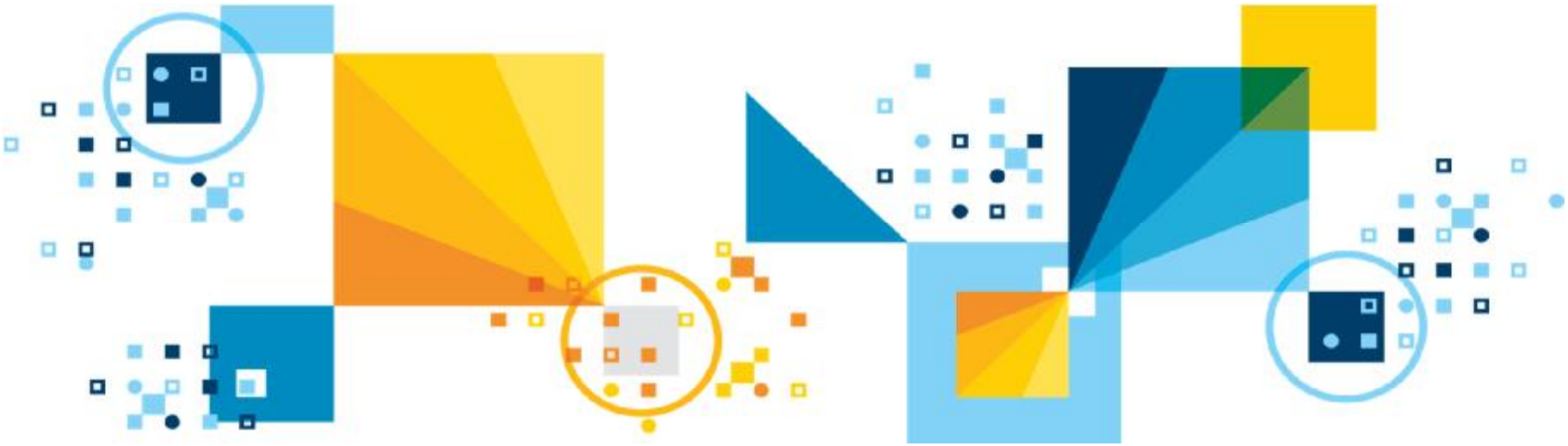
PRESENTATION NOTES FOR PAGE 40

The Next Steps…

- The next set of modules to consider :
  - Module DB2 SQL Query Tuning with BLU acceleration

PRESENTATION NOTES FOR PAGE 41

Questions?
askdata@ca.ibm.com



PRESENTATION NOTES FOR PAGE 42