Do more with less – Part Two

Resource and cost saving approach for
DB2 SQL workloads on z/OS

Agenda

- Recap of Part One

- Use Cases run through – with real world examples

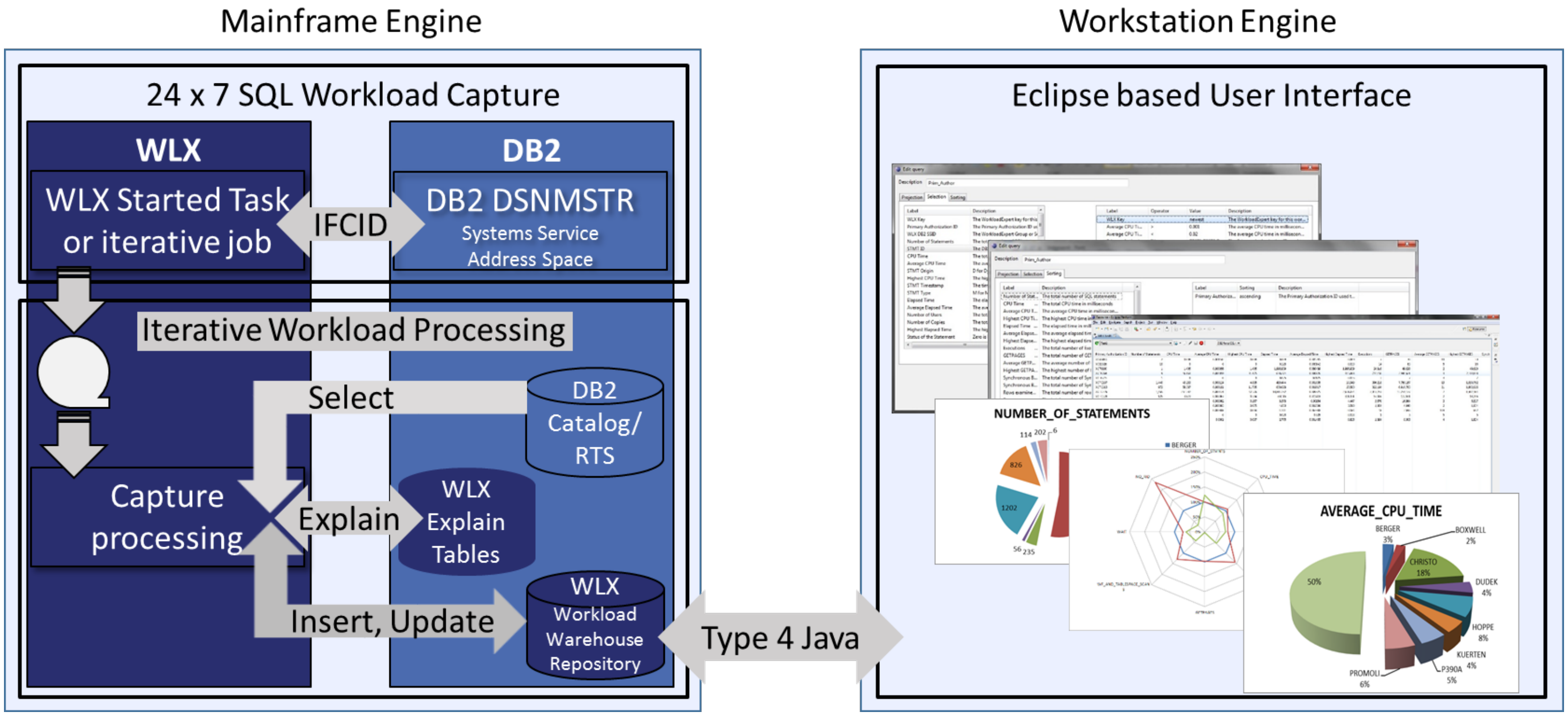
- Online Chat – Vote for the deep dive in the next webinar

Recap from Part One

In part one we discussed the new and enhanced IFCIDs and how they, cheaply, enable a new way to collect SQL information from the Enterprise

We also discussed possible uses of this data which led to the term „Use Case“ – The idea being a set of sample templates that help the DBA or developer find needed tuning/performance data out-of-the-box – As if the expert was by your side!

WLX Architecture



Mainframe Engine

24 x 7 SQL Workload Capture

WLX

WLX Started Task or iterative job

IFCID

DB2

DB2 DSNMSTR Systems Service Address Space

Iterative Workload Processing

Select

DB2 Catalog/ RTS

Capture processing

Explain

WLX Explain Tables

Insert, Update

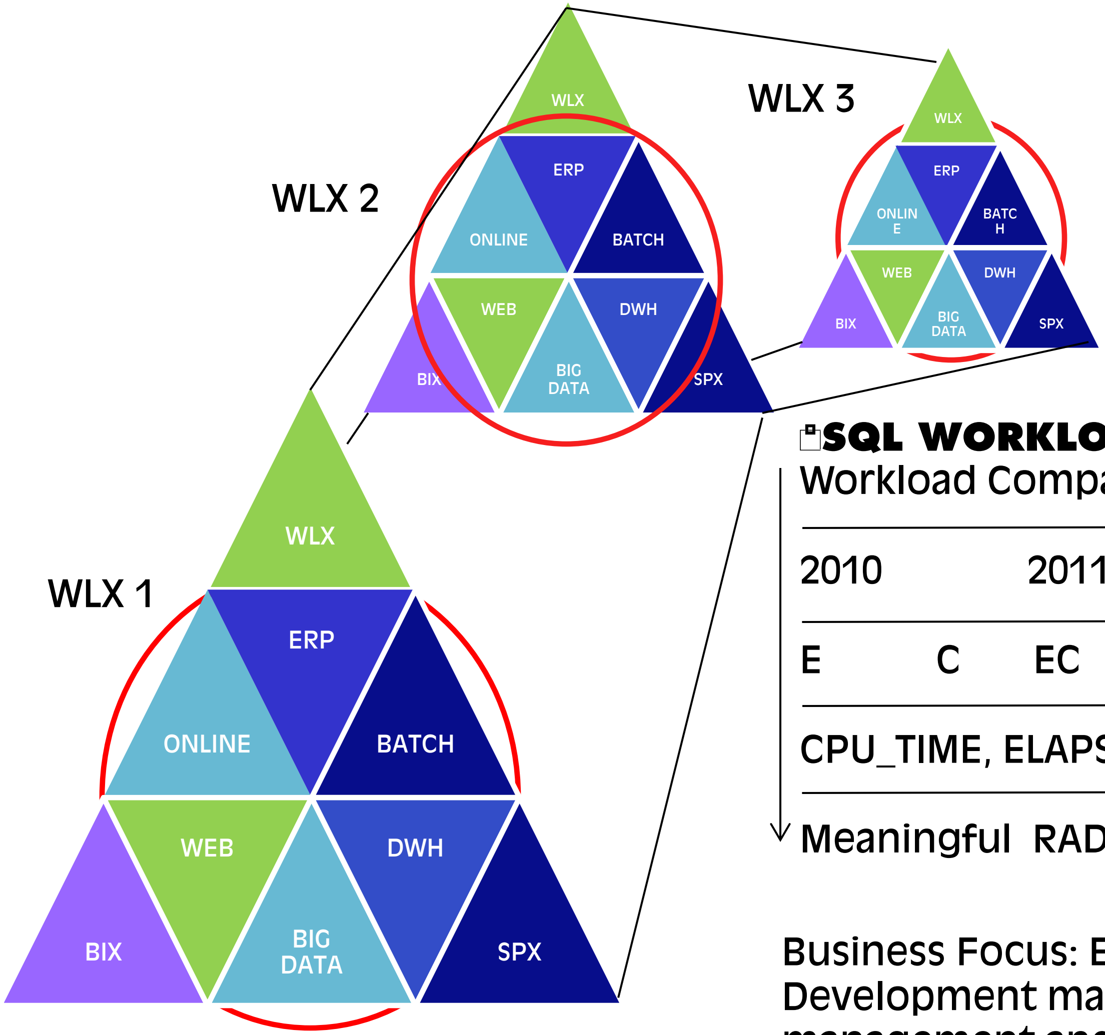
WLX Workload Warehouse Repository

Type 4 Java

Workstation Engine

Eclipse based User Interface

Correlate business peaks with MSU-needs



SQL WORKLOAD WAREHOUSE

Workload Comparison & Application Trending

2010 2011 2012 2013

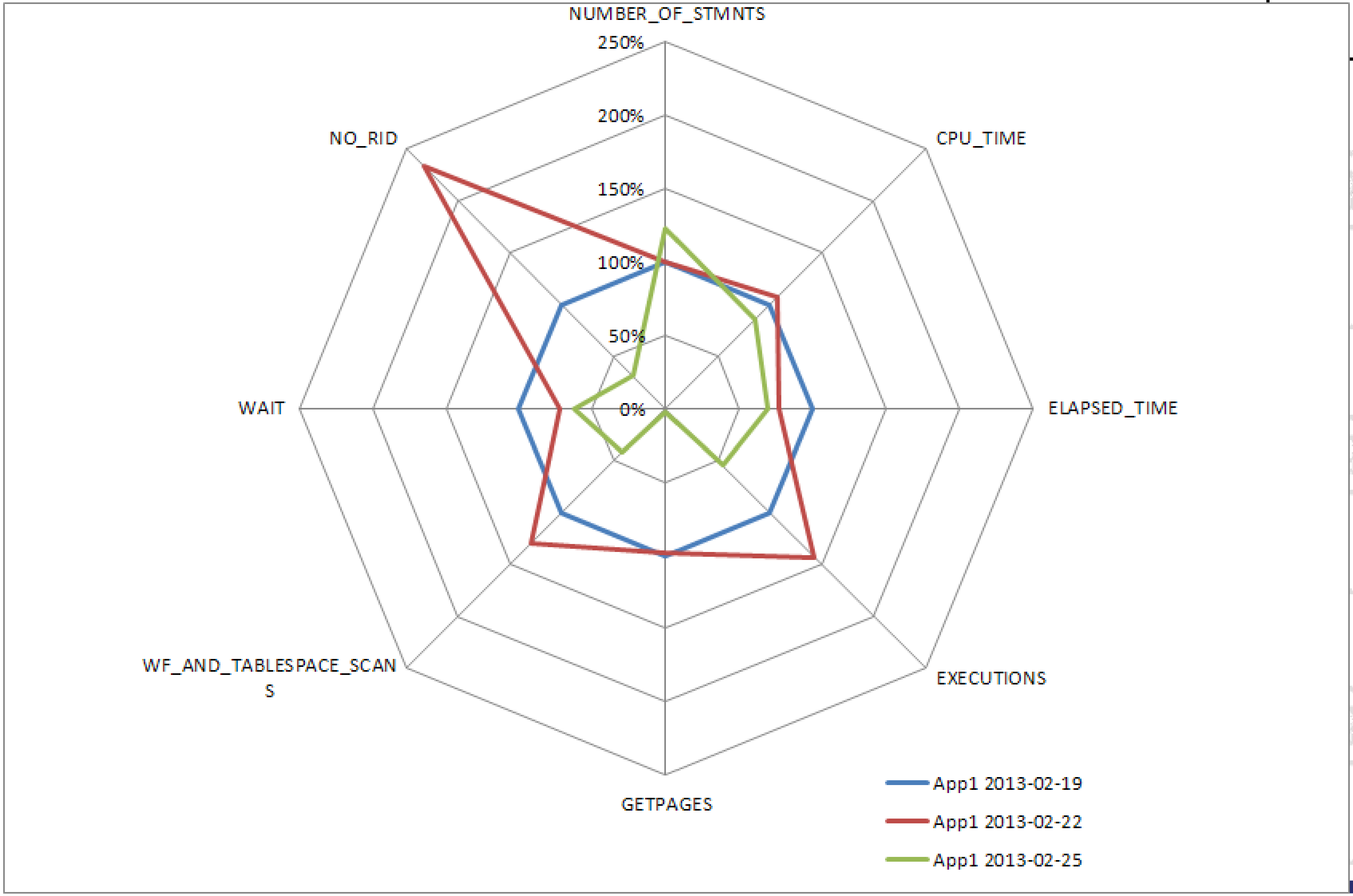
E C EC EC E ....

CPU_TIME, ELAPSED TIME, GETPAGES etc.

Meaningful RADAR Graphics

Business Focus: Easter & Christmas Sales
Development mapped with MSU usage for licence management and capacity planning

Compare releases using Radar charts



What matters for our needs

Counters # EXECUTIONS OF THE STATEMENT. FOR A CURSOR STATEMENT, THIS IS THE # OF OPENS. # OF SYNCHRONOUS BUFFER READS PERFORMED FOR STATEMENT # OF GETPAGE OPERATIONS PERFORMED FOR STATEMENT. # OF ROWS EXAMINED FOR STATEMENT. # OF ROWS PROCESSED FOR STATEMENT - FOR EXAMPLE, THE # OF ROWS RETURNED FOR A SELECT, OR THE NUMBER OF ROWS AFFECTED BY AN INSERT, UPDATE, OR DELETE. # OF SORTS PERFORMED FOR STATEMENT. # OF INDEX SCANS PERFORMED FOR STATEMENT. # OF TABLESPACE SCANS PERFORMED FOR STATEMENT. * # OF PARALLEL GROUPS CREATED FOR STATEMENT. # OF SYNCHRONOUS BUFFER WRITE OPERATIONS PERFORMED FOR STATEMENT.

O Counters # OF TIMES THAT A RID LIST WAS NOT USED BECAUSE THE # OF RIDS EXCEEDED ONE OR MORE INTERNAL DB2 LIMITS, AND THE # OF RID BLOCKS EXCEEDED THE VALUE OF SUBSYSTEM PARAMETER MAXTEMPS_RID. # OF TIMES THAT A RID LIST WAS NOT USED BECAUSE NOT ENOUGH STORAGE WAS AVAILABLE TO HOLD THE RID LIST, OR WORK FILE STORAGE OR RESOURCES WERE NOT AVAILABLE. # OF TIMES THAT A RID LIST OVERFLOWED TO A WORK FILE BECAUSE NO RID POOL STORAGE WAS AVAILABLE TO HOLD THE LIST OF RIDS*. # OF TIMES A THAT RID LIST OVERFLOWED TO A WORK FILE BECAUSE THE NUMBER OF RIDS EXCEEDED ONE OR MORE INTERNAL LIMITS*. # OF TIMES THAT APPENDING TO A RID LIST FOR A HYBRID JOIN WAS INTERRUPTED BECAUSE NO RID POOL STORAGE WAS AVAILABLE TO HOLD THE LIST OF RIDS*. # OF TIMES THAT APPENDING TO A RID LIST FOR A HYBRID JOIN WAS INTERRUPTED BECAUSE THE NUMBER OF RIDS EXCEEDED ONE OR MORE INTERNAL LIMITS*. # OF TIMES THAT RID LIST RETRIEVAL FOR MULTIPLE INDEX ACCESS WAS NOT DONE BECAUSE DB2 COULD NOT DETERMINE THE OUTCOME OF INDEX ANDING OR ORING*.

TIMINGS ACCUMULATED CPU TIME. THIS VALUE INCLUDES CPU TIME THAT IS CONSUMED ON AN IBM SPECIALTY ENGINE. ACCUMULATED ELAPSED TIME USED FOR STATEMENT. ACCUMULATED WAIT TIME FOR LATCH REQUESTS*. ACCUMULATED WAIT TIME FOR PAGE LATCHES*. ACCUMULATED WAIT TIME FOR DRAIN LOCKS*. ACCUMULATED WAIT TIME FOR DRAINS DURING WAITS FOR CLAIMS TO BE RELEASED*. ACCUMULATED WAIT TIME FOR LOG WRITERS. ACCUMULATED WAIT TIME FOR SYNCHRONOUS I/O. ACCUMULATED WAIT TIME FOR LOCK REQUESTS. ACCUMULATED WAIT TIME FOR A SYNCHRONOUS EXECUTION UNIT SWITCH. ACCUMULATED WAIT TIME FOR GLOBAL LOCKS. ACCUMULATED WAIT TIME FOR READ ACTIVITY THAT IS DONE BY ANOTHER THREAD. ACCUMULATED WAIT TIME FOR WRITE ACTIVITY THAT IS DONE BY ANOTHER THREAD.

IDENTIFICATION DATA SHARING MEMBER THAT CACHED THE SQL STATEMENT*. PROGRAM NAME. PROGRAM NAME IS THE NAME OF THE PACKAGE OR DBRM THAT PERFORMED THE PREPARE/SQL. PRECOMPILER LINE NUMBER FOR THE PREPARE STATEMENT OR SQL STATEMENT. TRANSACTION NAME. THIS VALUE IS PROVIDED DURING RRS SIGNON OR RE-SIGNON. END USER ID*. THIS VALUE IS PROVIDED DURING RRS SIGNON OR RE-SIGNON. WORKSTATION NAME*. THIS VALUE IS PROVIDED DURING RRS SIGNON OR RE-SIGNON. USER ID. USER ID IS THE PRIMARY AUTH. ID OF THE USER WHO DID THE INITIAL PREPARE. USER GROUP. USER GROUP IS THE CURRENT SQLID OF THE USER WHO DID THE INITIAL PREPARE. USER-PROVIDED IDENTIFICATION STRING.

ENVIRONMENTAL REFERENCED TABLE NAME. FOR STATEMENTS THAT REFERENCE MORE THAN ONE TABLE, ONLY THE NAME OF THE FIRST TABLE THAT IS REFERENCED IS REPORTED. (ALL REFERENCED OBJECTS ARE STORED IN THE WLI DATA MODEL) LITERAL REPLACEMENT FLAG* CURRENT SCHEMA. QUALIFIER THAT IS USED FOR UNQUALIFIED TABLE NAMES. BIND OPTIONS: ISOLATION, CURRENTDATA, AND DYNAMICRULES. SPECIAL REGISTER VALUES: CURRENT DEGREE, CURRENT RULES, AND CURRENT PRECISION. WHETHER THE STATEMENT CURSOR IS A HELD CURSOR. TIMESTAMP WHEN STATISTICS COLLECTION BEGAN. DATA COLLECTION BEGINS WHEN A TRACE FOR IFCID 318 IS STARTED. DATE AND TIME WHEN THE STATEMENT WAS INSERTED INTO THE CACHE IN STORE CLOCK FORMAT. DATE AND TIME WHEN THE STATEMENT WAS UPDATED, IN STORE CLOCK FORMAT. DATE AND TIME WHEN THE STATEMENT WAS UPDATED, IN INTERNAL FORMAT.

SQL WorkloadExpert Use Cases run through

- Taste one „Use Case“ … and you can‘t resist.
- Appetite comes with eating. It‘s moreish.
- L‘appétite vient en mangeant!

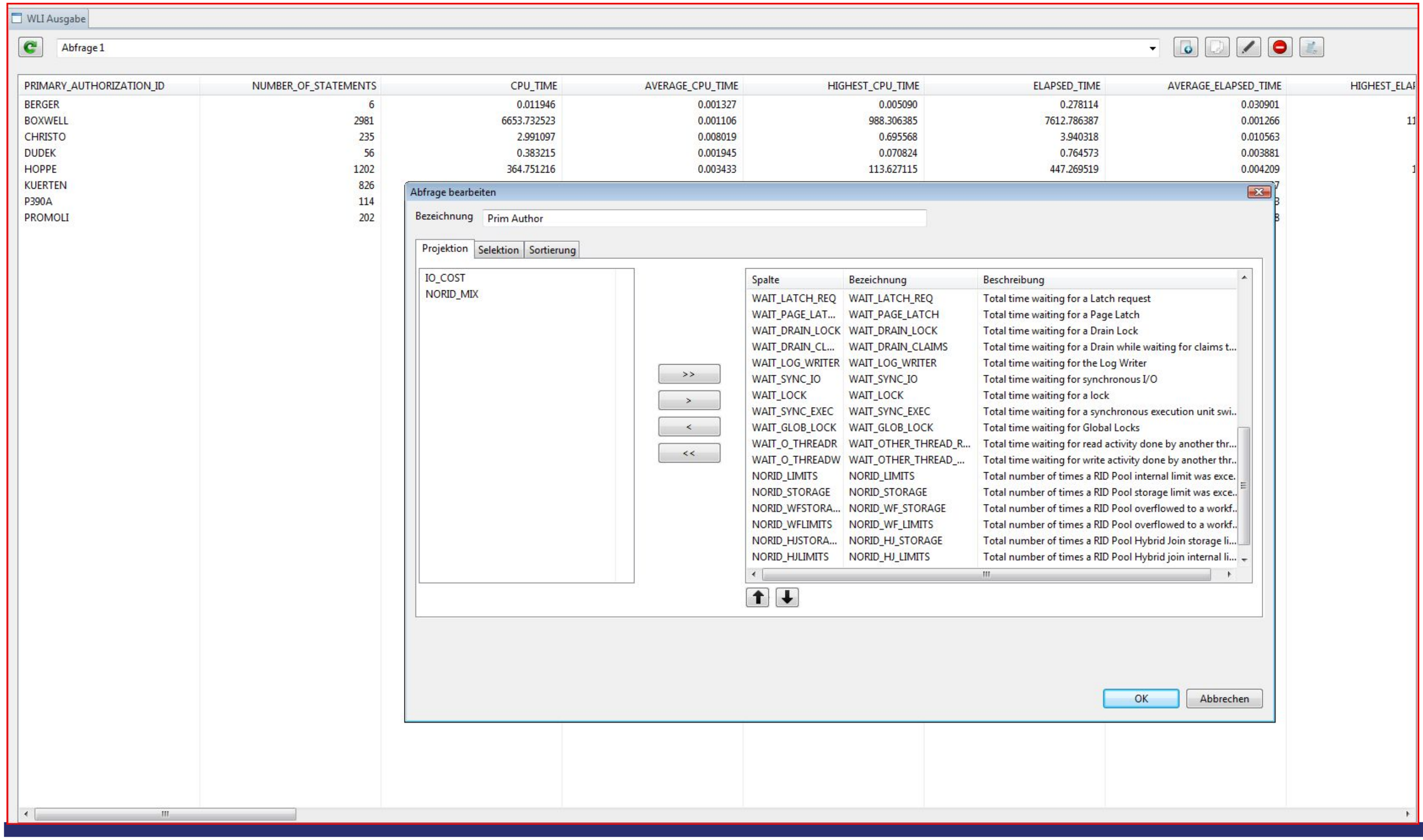
SQL WorkloadExpert for DB2 z/OS

Use case 1: Application Workload Analysis –

Which machine load is produced by a certain application?

SQL WorkloadExpert on Trial

Use case 1: Application workload analysis – which machine load is produced by a certain application



PRIMARY_AUTHORIZATION_ID	NUMBER_OF_STATEMENTS	CPU_TIME	AVERAGE_CPU_TIME	HIGHEST_CPU_TIME	ELAPSED_TIME	AVERAGE_ELAPSED_TIME	HIGHEST_ELAF
BERGER	6	0.011946	0.001327	0.005090	0.278114	0.030901	
BOXWELL	2981	6653.732523	0.001106	988.306385	7612.786387	0.001266	11
CHRISTO	235	2.991097	0.008019	0.695568	3.940318	0.010563	
DUDEK	56	0.383215	0.001945	0.070824	0.764573	0.003881	
HOPPE	1202	364.751216	0.003433	113.627115	447.269519	0.004209	1
KUERTEN	826					7	
P390A	114					3	
PROMOLI	202					3	

Abfrage bearbeiten

Bezeichnung: Prim Author

IO_COST
NORID_MIX

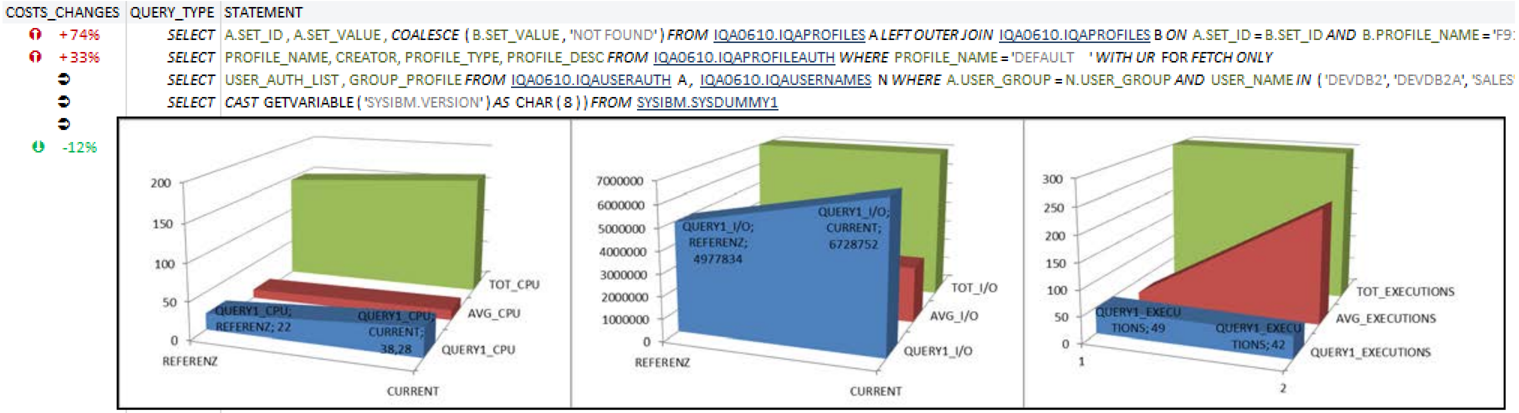
Spalte	Bezeichnung	Beschreibung
WAIT_LATCH_REQ	WAIT_LATCH_REQ	Total time waiting for a Latch request
WAIT_PAGE_LAT...	WAIT_PAGE_LATCH	Total time waiting for a Page Latch
WAIT_DRAIN_LOCK	WAIT_DRAIN_LOCK	Total time waiting for a Drain Lock
WAIT_DRAIN_CL...	WAIT_DRAIN_CLAIMS	Total time waiting for a Drain while waiting for claims t...
WAIT_LOG_WRITER	WAIT_LOG_WRITER	Total time waiting for the Log Writer
WAIT_SYNC_IO	WAIT_SYNC_IO	Total time waiting for synchronous I/O
WAIT_LOCK	WAIT_LOCK	Total time waiting for a lock
WAIT_SYNC_EXEC	WAIT_SYNC_EXEC	Total time waiting for a synchronous execution unit swi..
WAIT_GLOB_LOCK	WAIT_GLOB_LOCK	Total time waiting for Global Locks
WAIT_O_THREADR	WAIT_OTHER_THREAD_R...	Total time waiting for read activity done by another thr...
WAIT_O_THREADW	WAIT_OTHER_THREAD_...	Total time waiting for write activity done by another thr..
NORID_LIMITS	NORID_LIMITS	Total number of times a RID Pool internal limit was exce.
NORID_STORAGE	NORID_STORAGE	Total number of times a RID Pool storage limit was exce..
NORID_WFSTORA...	NORID_WF_STORAGE	Total number of times a RID Pool overflowed to a workf..
NORID_WFLIMITS	NORID_WF_LIMITS	Total number of times a RID Pool overflowed to a workf..
NORID_HJSTORA...	NORID_HJ_STORAGE	Total number of times a RID Pool Hybrid Join storage li...
NORID_HJLIMITS	NORID_HJ_LIMITS	Total number of times a RID Pool Hybrid join internal li...

SQL WorkloadExpert for DB2 z/OS

Use case 2 : Workload – Change, Problem Detection and Trending. Compare CPU, I/O, execution rates, current KPIs and deltas – Calculated and summarized to the application level

SQL WorkloadExpert on Trial

Use case 2: Trending of Applications – Compare of CPU, I/O , execution rates, current KPIs and deltas – calculated and summarized to application level



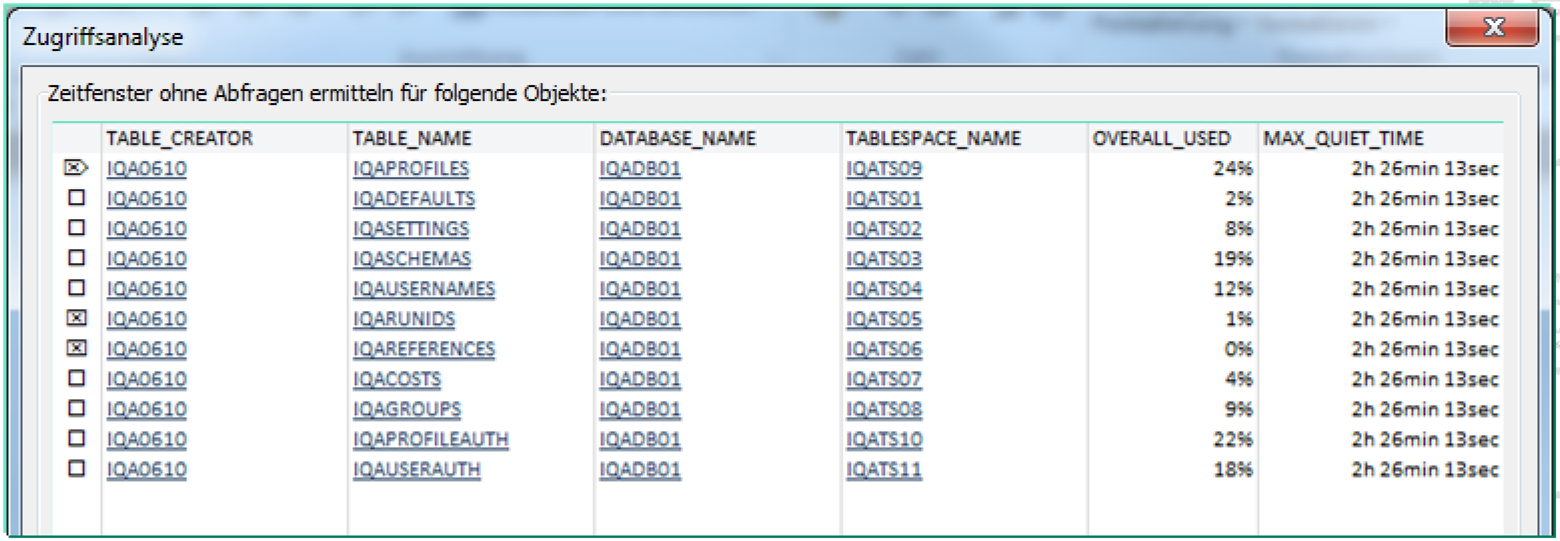
COSTS_CHANGES	QUERY_TYPE	STATEMENT
⬆ +74%	SELECT	A.SET_ID , A.SET_VALUE , COALESCE (B.SET_VALUE , 'NOT FOUND') FROM IQA0610.IQAPROFILES A LEFT OUTER JOIN IQA0610.IQAPROFILES B ON A.SET_ID = B.SET_ID AND B.PROFILE_NAME = 'F9:
⬆ +33%	SELECT	PROFILE_NAME, CREATOR, PROFILE_TYPE, PROFILE_DESC FROM IQA0610.IQAPROFILEAUTH WHERE PROFILE_NAME = 'DEFAULT ' WITH UR FOR FETCH ONLY
➔	SELECT	USER_AUTH_LIST , GROUP_PROFILE FROM IQA0610.IQAUSERAUTH A , IQA0610.IQAUSERNAMES N WHERE A.USER_GROUP = N.USER_GROUP AND USER_NAME IN ('DEVDB2', 'DEVDB2A', 'SALES'
➔	SELECT	CAST GETVARIABLE ('SYSIBM.VERSION') AS CHAR (8)) FROM SYSIBM.SYSDUMMY1
➔		
⬇ -12%		

SQL WorkloadExpert for DB2 z/OS

Use case 3 : Object Quiet Times for maintenance (REORG)

SQL WorkloadExpert on Trial

Use case 3: Object Quiet Times for maintenance (REORG)



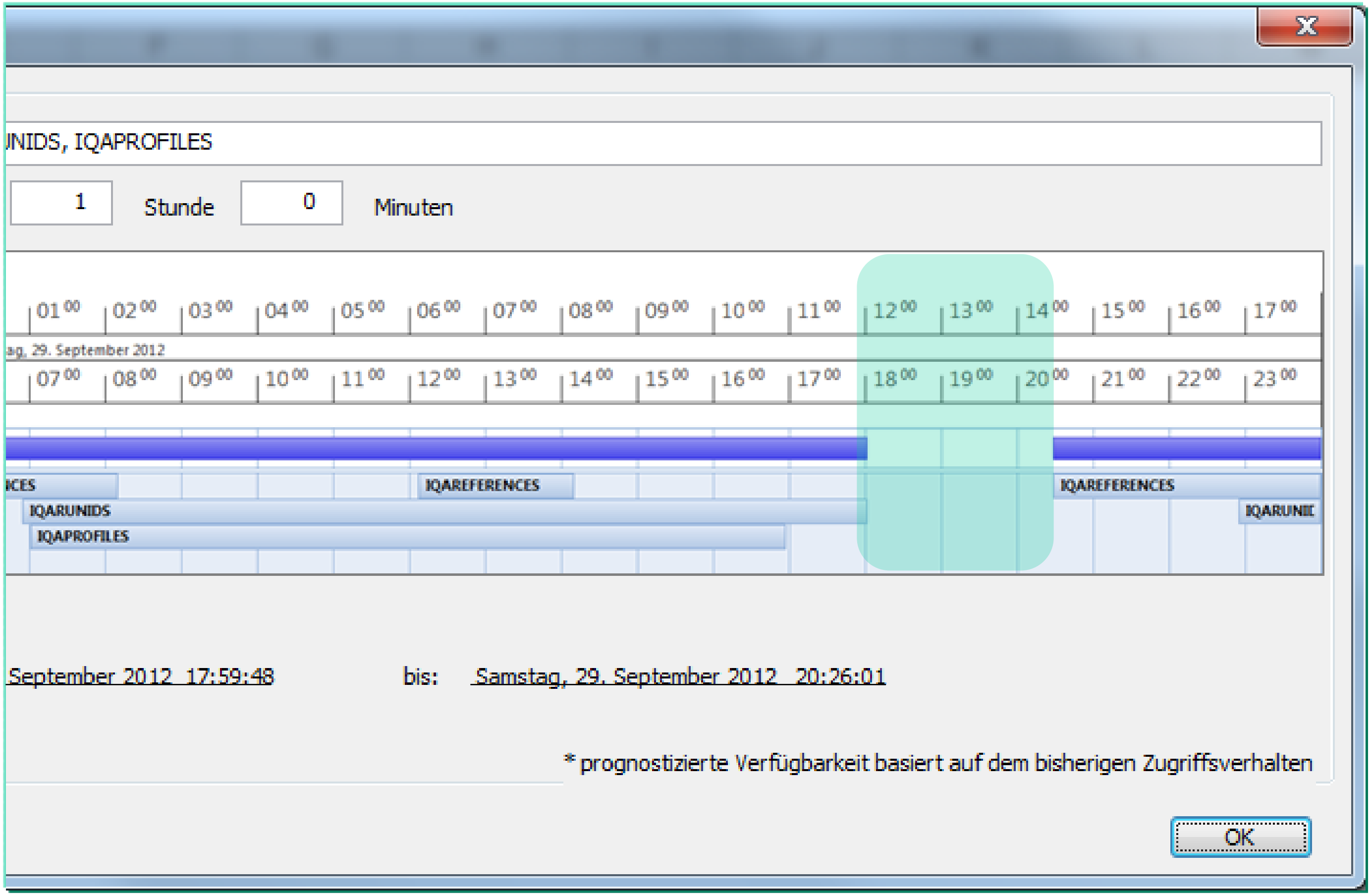
Zugriffsanalyse

Zeitfenster ohne Abfragen ermitteln für folgende Objekte:

	TABLE_CREATOR	TABLE_NAME	DATABASE_NAME	TABLESPACE_NAME	OVERALL_USED	MAX_QUIET_TIME
☒	IQA0610	IQAPROFILES	IQADB01	IQATS09	24%	2h 26min 13sec
☐	IQA0610	IQADEFAULTS	IQADB01	IQATS01	2%	2h 26min 13sec
☐	IQA0610	IQASETTINGS	IQADB01	IQATS02	8%	2h 26min 13sec
☐	IQA0610	IQASCHEMAS	IQADB01	IQATS03	19%	2h 26min 13sec
☐	IQA0610	IQAUSERNAMES	IQADB01	IQATS04	12%	2h 26min 13sec
☒	IQA0610	IQARUNIDS	IQADB01	IQATS05	1%	2h 26min 13sec
☒	IQA0610	IQAREFERENCES	IQADB01	IQATS06	0%	2h 26min 13sec
☐	IQA0610	IQACOSTS	IQADB01	IQATS07	4%	2h 26min 13sec
☐	IQA0610	IQAGROUPS	IQADB01	IQATS08	9%	2h 26min 13sec
☐	IQA0610	IQAPROFILEAUTH	IQADB01	IQATS10	22%	2h 26min 13sec
☐	IQA0610	IQAUSERAUTH	IQADB01	IQATS11	18%	2h 26min 13sec

SQL WorkloadExpert on Trial

Use case 3: Object Quiet Times for maintenance (REORG)



INIDS, IQAPROFILES

1 Stunde 0 Minuten

ag, 29. September 2012

September 2012 17:59:48 bis: Samstag, 29. September 2012 20:26:01

* prognostizierte Verfügbarkeit basiert auf dem bisherigen Zugriffsverhalten

SQL WorkloadExpert for DB2 z/OS

Use case 4 : AUDIT (Who did What, Where, When and How often?)

SQL WorkloadExpert on Trial

Use case 4: Audit

- User BOXWELL manages to run three SQLs at the same time from three separate servers in Düsseldorf, München and Sao Paulo.
- How is this possible? Has there been a password leak?
- Who has SELECTed from my Payroll table?

SQL WorkloadExpert for DB2 z/OS

Use case 5 : Never used Objects

e.g. Collections, Packages,Tables, MQTs, and Indexes

SQL WorkloadExpert on Trial

Use case 5: Never used objects (Collections, Packages, Tables, MQTs, and Indexes)

- With perhaps a years worth of data you can get a *very* good idea about which objects are being used. If the Catalog & RTS LASTUSED columns are also used then you can relatively safely STOP the object(s) and then DROP them. For packages verify that any needed LOAD modules and DBRMs are available for a new BIND, just in case, and then start FREEing them all!

- Remember that UNIQUE indexes are not marked as used if they are *only* used to get a programmatic SQLCODE -803

SQL WorkloadExpert for DB2 z/OS

Use case 6 : Never executed (static) SQL

SQL WorkloadExpert on Trial

Use case 6: Never executed (static) SQL

- All DBAs have seen „bad“ SQL in pre-production and have then spent time „correcting“ it or even creating/altering indexes to tune it up, however, in reality this SQL is never actually executed.

- These dead SQLs cause problems because they show up in the dependency checks of packages which then causes the package to be needlessly verified by **BIX** or even actually REBINDed with all the risks that that entails!

SQL WorkloadExpert for DB2 z/OS

Use case 7 : Forecasting of possible performance improvements in dynamic SQL by exchanging literals with parameter markers.

SQL WorkloadExpert on Trial

Use case 7: Forecasting of possible performance improvements in dynamic SQL by exchanging literals with parameter markers.

- The idea here is to check *all* the filtered dynamic SQL for usage of literals. If found then the literal(s) will be exchanged for CASTed Parameter Marker(s). Then the SQL will be reEXPLAINed and the „new“ access path checked against the „old“ using our **BIX** technology thus enabling the DBA to know whether or not parameter markers will help or heed these SQLs.

- Remember though that Filter Factors (literal usage) can be good and bad for the *same* SQL statement!

SQL WorkloadExpert for DB2 z/OS

Use case 8 : Disc problem detection – I/O rates

SQL WorkloadExpert on Trial

Use case 8: Disc problem detection – I/O rates

- As WLX has all the data about how long it takes for synchronous IOs it can calculate your IO speeds and thus warn when:
  - Any wait time per synchronous IO is over two milliseconds
  - For OLTP any application that has more than one synchronous IO per statement

SQL WorkloadExpert for DB2 z/OS

Use case 9 : Bufferpool Analysis

Hit ratios, VPSEQT-Tuning…

SQL WorkloadExpert on Trial

Use case 9 : Bufferpool analysis – hit ratios ,VPSEQT-Tuning (Virtual Pool Sequential Threshold). Is the buffer pool primarily random or sequential?

- The following values are then calculated:
  - System and Application Hit ratios
  - Three Residency times (System, Random Page and Sequential page)
  - Two Bufferpool write efficiency measures (Page updates per page written and pages written per write IO)
  - VPSEQT
  - Bufferpool intensity. Which is a good pointer to use PAGEFIX=YES

SQL WorkloadExpert for DB2 z/OS

Use case 10: Multi-row fetch candidate detection

Use case 10: Multi-row fetch candidate detection

- In DB2 V8 Multi-row FETCH (as well as insert & update, however we will only deal with SELECT as that is a „low hanging fruit“) was introduced. The problem is no-one changed the existing programs to actually use it (never change a running program) but it is shown by benchmarking to save about 50% of FETCH related CPU which is pretty good!
- So how do you find the top 10 CURSORs that would give the biggest return?
- Find all non-ROWSET defined cursors and calculate the No. Fetch‘s / No. of Execution‘s if this number is > 100 BINGO! You have found a candidate!

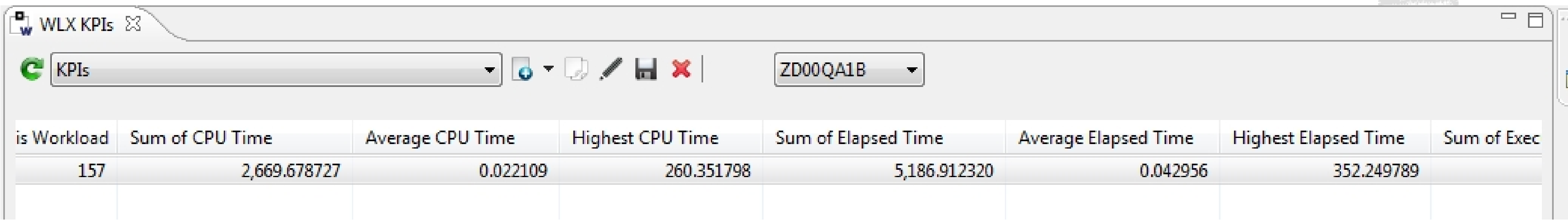
SQL WorkloadExpert for DB2 z/OS

Use case 11 : SQL KPIs

Background noise and exceptions

SQL WorkloadExpert on Trial

Use case 11: SQL WLX KPIs - Background noise and exceptions



is Workload	Sum of CPU Time	Average CPU Time	Highest CPU Time	Sum of Elapsed Time	Average Elapsed Time	Highest Elapsed Time	Sum of Exec
157	2,669.678727	0.022109	260.351798	5,186.912320	0.042956	352.249789	

Here you can easily see that for all „numbers of interest“ we output the Sum, the Average, and the Highest – Thus enabling you to drill down and find the outliers…

SQL WorkloadExpert for DB2 z/OS

Use case 12 : SELECT only table detection

SQL WorkloadExpert on Trial

Use case 12: SELECT only table detection

Here it is desired to find any objects that *only* have SELECT SQLs running against them. The idea here is to check if these, typically look-up or xref style tables, should be moved to their „own“ Bufferpool etc.

Use case 13 : Long Delay Detection

SQL WorkloadExpert on Trial

Use case 13: Long Delay detection

- Normally a transaction (SQL) takes 0.2 seconds to complete but every now and again it spikes up to say 30 seconds. No DEADLOCK as it finally completes but the user is puzzled or upset!

- Here WLX allows you to “scroll back” to the time of the delay and then see any of the inflight SQL that references any of the same objects in the current SQL.

- This enables the finding of the bad guy who is blocking!

SQL WorkloadExpert for DB2 z/OS

Use case 14 : Deadlock, Lock escalation, Index page splits, and BIF usage

SQL WorkloadExpert on Trial

Use case 14: Deadlock, Lock escalation, Index page splits, and BIF usage

- Using four other IFCIDS it is possible to trap any or all of the above occurrences and then enable full detection of:
  - Why the DEADLOCK occurred (You get all holders not just the first one).
  - For lock escalation you can see who „is being naughty“
  - For index page splits you get a chance to redesign the index or change the definition and then REORG to stop the splits causing problems
  - Use of changed BIFs in DB2 9

SQL WorkloadExpert for DB2 z/OS

Use case 15 : Multi-snap

SQL WorkloadExpert on Trial

Use case 15: Multi-snap

- The point here is to get a very granular view of the SQL running on the system. In this mode it is proposed to not EXPLAIN (As that could take too long) but to simply snap all the SQL as quickly as you can.

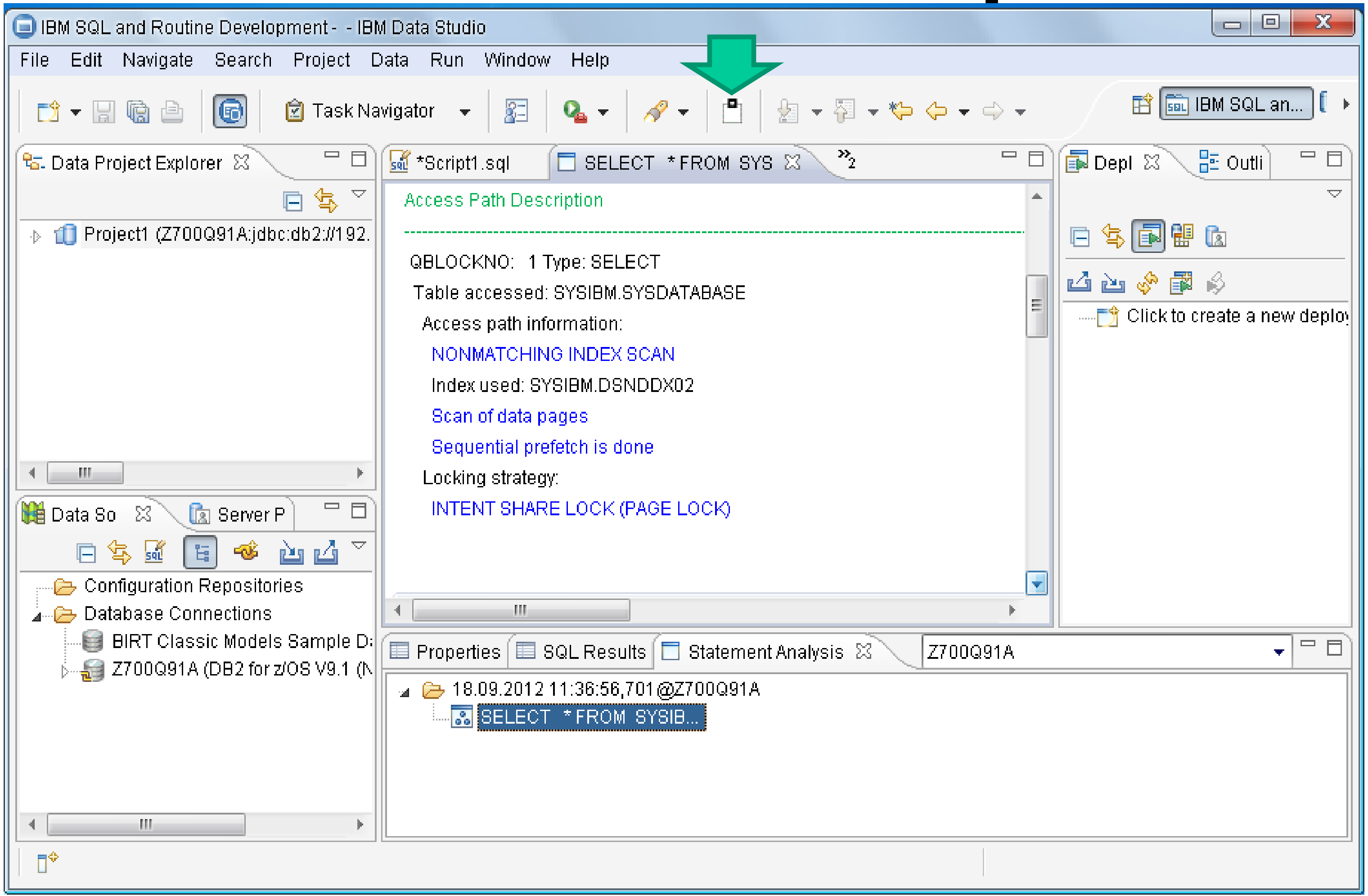
- Doing this at „known problem times“ e.g. from 20:00 until 21:00 then enables a thorough investigation of which SQL caused the peak in CPU at 20:15 which, when averaged out over 30 minutes, would not seem „that bad“

SQL WorkloadExpert for DB2 z/OS

Use case 16 : SPX (**SQL PerformanceExpert**) link (IBM DataStudio)

SQL WorkloadExpert on Trial

Use case 16: SPX (SQLPerformanceExpert) link



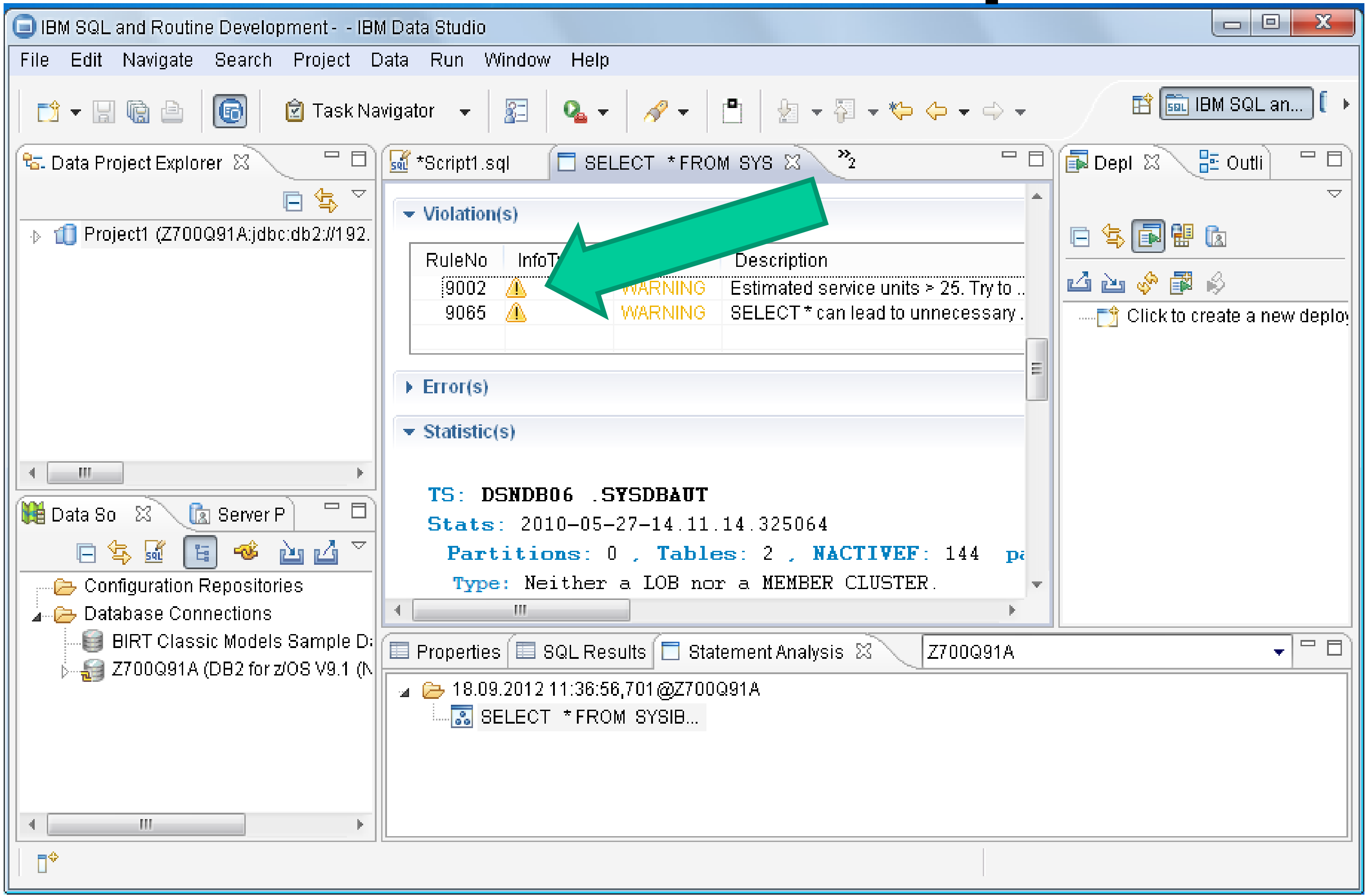
IBM SQL and Routine Development - - IBM Data Studio
File Edit Navigate Search Project Data Run Window Help
Task Navigator
IBM SQL an...
Data Project Explorer
Project1 (Z700Q91A:jdbc:db2://192.
*Script1.sql
SELECT * FROM SYS
Depl
Outli
Click to create a new deploy
Data So
Server P
Configuration Repositories
Database Connections
BIRT Classic Models Sample D
Z700Q91A (DB2 for z/OS V9.1 (N
Properties
SQL Results
Statement Analysis
Z700Q91A

```
Access Path Description
--------------------------------------------------------------------------------
QBLOCKNO:   1 Type: SELECT
 Table accessed: SYSIBM.SYSDATABASE
  Access path information:
    NONMATCHING INDEX SCAN
    Index used: SYSIBM.DSNDDX02
    Scan of data pages
    Sequential prefetch is done
  Locking strategy:
    INTENT SHARE LOCK (PAGE LOCK)
```

```
18.09.2012 11:36:56,701@Z700Q91A
  SELECT * FROM SYSIB...
```

SQL WorkloadExpert on Trial

Use case 16: SPX (SQLPerformanceExpert) link


IBM SQL and Routine Development - - IBM Data Studio
File Edit Navigate Search Project Data Run Window Help
Task Navigator
IBM SQL an...
Data Project Explorer
Project1 (Z700Q91A:jdbc:db2://192.
*Script1.sql
SELECT * FROM SYS
Depl
Outli
Click to create a new deplo
Violation(s)
Error(s)
Statistic(s)
Data So
Server P
Configuration Repositories
Database Connections
BIRT Classic Models Sample D
Z700Q91A (DB2 for z/OS V9.1 (N
Properties
SQL Results
Statement Analysis
Z700Q91A
18.09.2012 11:36:56,701@Z700Q91A
SELECT * FROM SYSIB...

RuleNo	InfoT		Description
9002	⚠	WARNING	Estimated service units > 25. Try to ..
9065	⚠	WARNING	SELECT * can lead to unnecessary .

```
TS: DSNDB06 .SYSDBAUT
Stats: 2010-05-27-14.11.14.325064
  Partitions: 0 , Tables: 2 , NACTIVEF: 144  pa
  Type: Neither a LOB nor a MEMBER CLUSTER.
```

SQL WorkloadExpert for DB2 z/OS

Use case 17 : REORG Detector and Suppressor

SQL WorkloadExpert on Trial

Use case 17: REORG Detector & Suppressor

- As we „know“ the usage of objects and thanks to the RTS we know when the last reorgs were done we can see if a reorg actually helped reduce the IO or CPU. If it made no or marginal difference then probably better to forget it! If the access is purely Random then also forget it.
- If, however, we see a candidate then insert it into the **RTDX** API to do an OnDemand REORG.

SQL WorkloadExpert for DB2 z/OS

Use case 18 : Eager vs. Lazy Loader detection (JPA-Java Persistence API)

SQL WorkloadExpert on Trial

Use case 18: Eager vs Lazy Loader detection (JPA – Java Persistence API)

- Various JPAs are used and some have Eager Loading or Lazy Loading as an option. Eager instantiates everything for everywhere which can be overkill! Lazy delays the instantiation until used.
- As we have all of the SQL and all of the Columns for a given application and/or user and connection (Java etc.) we can make a recommendation of whether or not eager is better than lazy. As this is actually an Application Compare it needs a code change in Java of course!

SQL WorkloadExpert for DB2 z/OS

Use case 19 : Object Usage of Application including service naming by object (enhanced RECOVER support)

SQL WorkloadExpert on Trial

Use case 19: Object Usage by Application including service naming by object (Enhanced RECOVER support)

- Problem here is a RECOVER scenario with various tables and the decision „Which order should be used?“ T1, T2, T3, T4 or T4, T2, T1, T3 etc

- With WLX you see the KPIs for all of the tables and further there is a user defined „son segment“ for *every* object where the customer can write free form text about „What is this object? Which service uses this object? How important is this object?“ when all this data is then available the correct order of RECOVER is clear!

SQL WorkloadExpert for DB2 z/OS

Use case 20 : Offline Performance Database

SQL WorkloadExpert on Trial

Use case 20: Offline Performance Database

- Use DB2 LUW or DB2 Express-C to hold an unloaded version of the z/OS WLX Performance Database for offline data mining etc.

- This feature enables years of data to be kept on cheap PC disks while the z/OS holds the “current” workload metrics.

SQL WorkloadExpert for DB2 z/OS

Use case 21 : Up and down scaling SQL Workloads

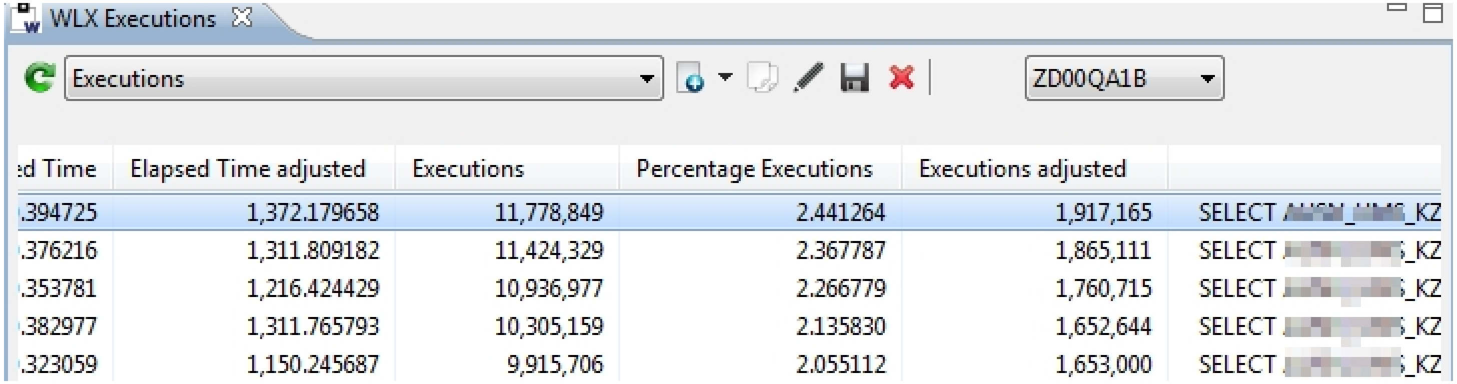
Use case 21: Up and Down scaling

- One of the many problems we face is finding out the background level of work and where should we concentrate our efforts at tuning or problem discovery.

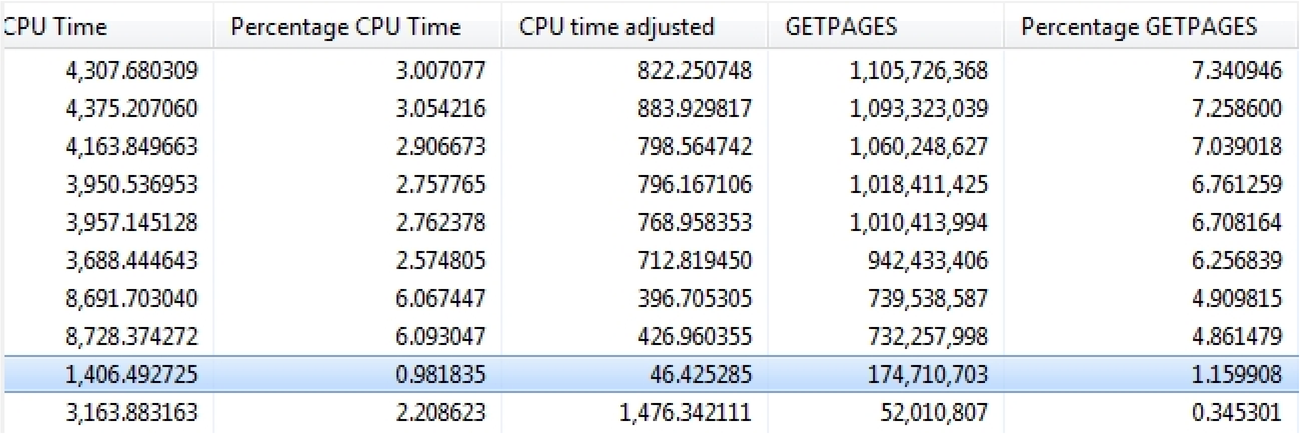
- This Use Case takes the workload and scales or adjusts the figures of all the workload to a single factor (Default is 60 minutes). Thus SQLs that run for days are adjusted down to appear as if they have only run for one hour and SQLs that have executed at least twice and have been in the cache for at least ten minutes are adjusted up so it appears they have also been run for one hour thus enabling a true comparison.

SQL WorkloadExpert on Trial

Use case 21: Here you can see some examples of Up and Down scaling:



d Time	Elapsed Time adjusted	Executions	Percentage Executions	Executions adjusted	
.394725	1,372.179658	11,778,849	2.441264	1,917,165	SELECT [redacted]_KZ
.376216	1,311.809182	11,424,329	2.367787	1,865,111	SELECT [redacted]_KZ
.353781	1,216.424429	10,936,977	2.266779	1,760,715	SELECT [redacted]_KZ
.382977	1,311.765793	10,305,159	2.135830	1,652,644	SELECT [redacted]_KZ
.323059	1,150.245687	9,915,706	2.055112	1,653,000	SELECT [redacted]_KZ



CPU Time	Percentage CPU Time	CPU time adjusted	GETPAGES	Percentage GETPAGES
4,307.680309	3.007077	822.250748	1,105,726,368	7.340946
4,375.207060	3.054216	883.929817	1,093,323,039	7.258600
4,163.849663	2.906673	798.564742	1,060,248,627	7.039018
3,950.536953	2.757765	796.167106	1,018,411,425	6.761259
3,957.145128	2.762378	768.958353	1,010,413,994	6.708164
3,688.444643	2.574805	712.819450	942,433,406	6.256839
8,691.703040	6.067447	396.705305	739,538,587	4.909815
8,728.374272	6.093047	426.960355	732,257,998	4.861479
1,406.492725	0.981835	46.425285	174,710,703	1.159908
3,163.883163	2.208623	1,476.342111	52,010,807	0.345301

SQL WorkloadExpert for DB2 z/OS

Use case 22 : Same SQL with Multiple Schemas

SQL WorkloadExpert on Trial

Use case 22: Same SQL with Multiple Schemas

A lot of shops these days have multiple creators (schemas) but the tables are the same! All other SQL tools do not take cognizance of this fact and so aggregate the data “incorrectly”.

With this Case you can view the true Aggregate SQLs.

SQL WorkloadExpert for DB2 z/OS

Use case 23 : DSC Flush Analysis

SQL WorkloadExpert on Trial

Use case 23: DSC/SSC Flush analysis

- With all of the data that WLX collects the so called Flush Rate is one of the interesting by products. This shows you how many statements per hour are being flushed and is a major pointer to either:
  - Increase the size of the DSC/EDMPOOL
  - Rewrite the SQL for parameter marker usage
  - Change the BIND parameters to ***not*** use the DSC for certain packages or SQLs – Not recommended of course!

SQL WorkloadExpert for DB2 z/OS

Use case 24 : SQL Text Analysis

SQL WorkloadExpert on Trial

Use case 24: SQL text analysis

- Sometime it is very interesting to search the SQL for use (or non-use) of certain keywords or text fragments. E.g. “ * “ or “.*” or “CHAR(“, DISTINCT, “FOR%FETCH%ONLY” etc.
- This case enables a simple text search for any text with/without wildcards of any/all the SQL in WLX.

SQL WorkloadExpert for DB2 z/OS

Use case 25 : Application Object Cross-reference builder

Use case 25: Application Object Cross-reference builder

- In Use case 19 we introduced the concept of linking an object to an Application Name and with this Use Case we go a step further. The idea here is that the user defines a list of one or more Primary Auth Ids (For DSC) and Package/CollIds (For SSC), both with wild cards of course, and then an Application Name. When WLX is running it will use this table to automatically insert the matching Application name(s) in the Case 19 table. This Use Case thus saves a lot of time and effort and it nearly automates the application detection definition.

SQL WorkloadExpert for DB2 z/OS

Use case 26: CLUSTER index detection

Use case 26: CLUSTER index detection

This Use Case will check which Indexes are used, how they are used (non-matching, with Prefetch etc.) and how often. From this data it can be deduced which index could/should be the CLUSTER index.

SQL WorkloadExpert for DB2 z/OS

Use case 27: Object view

Use case 27: Object view

This Use case is simply a view “up” the data model starting principally from a list of Objects and/or Applications.

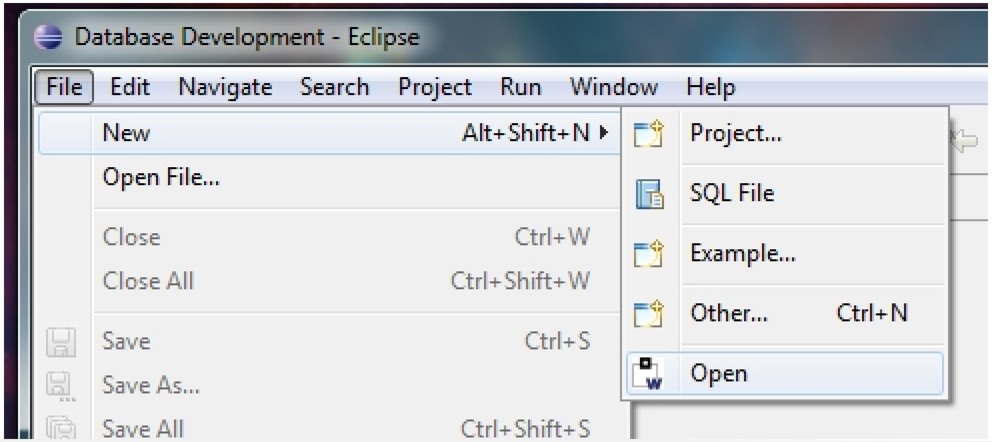
SQL WorkloadExpert on Trial

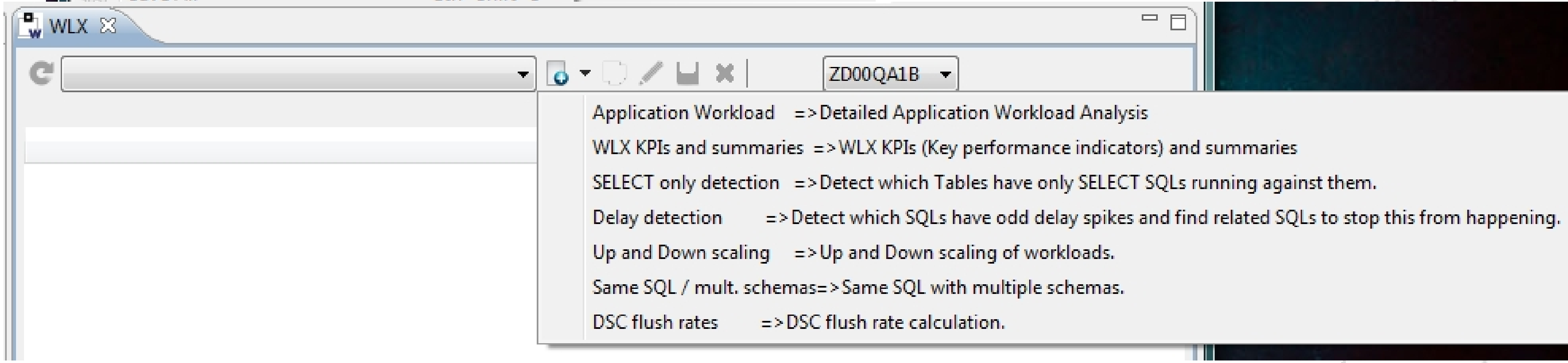
After more than 20 Use cases…

The proof of the pudding is in the eating

SQL WorkloadExpert on Trial

Use case Examples:





Application Workload =>Detailed Application Workload Analysis

WLX KPIs and summaries =>WLX KPIs (Key performance indicators) and summaries

SELECT only detection =>Detect which Tables have only SELECT SQLs running against them.

Delay detection =>Detect which SQLs have odd delay spikes and find related SQLs to stop this from happening.

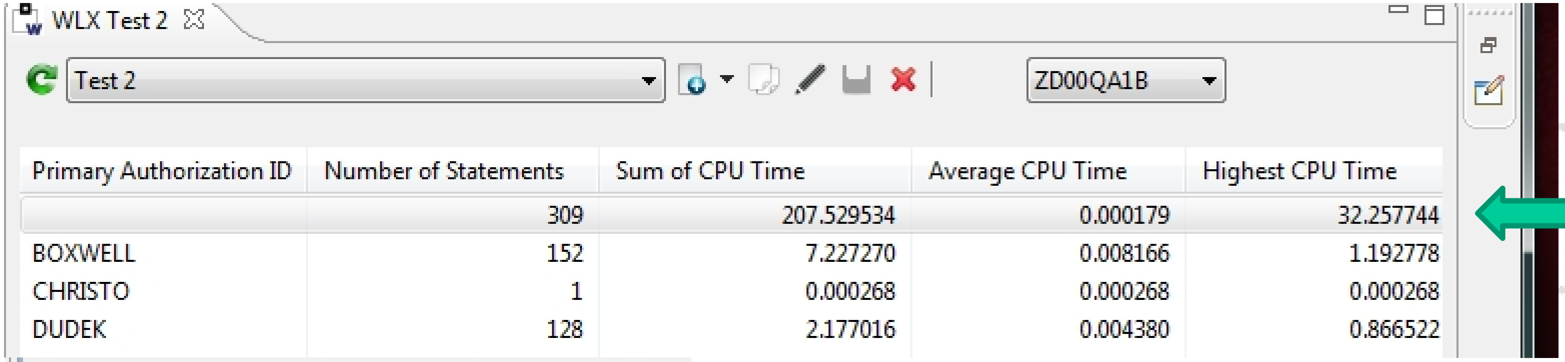
Up and Down scaling =>Up and Down scaling of workloads.

Same SQL / mult. schemas=>Same SQL with multiple schemas.

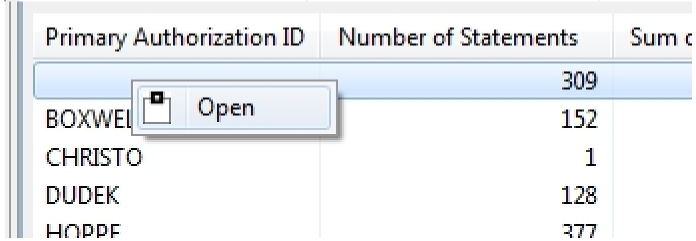
DSC flush rates =>DSC flush rate calculation.

SQL WorkloadExpert on Trial

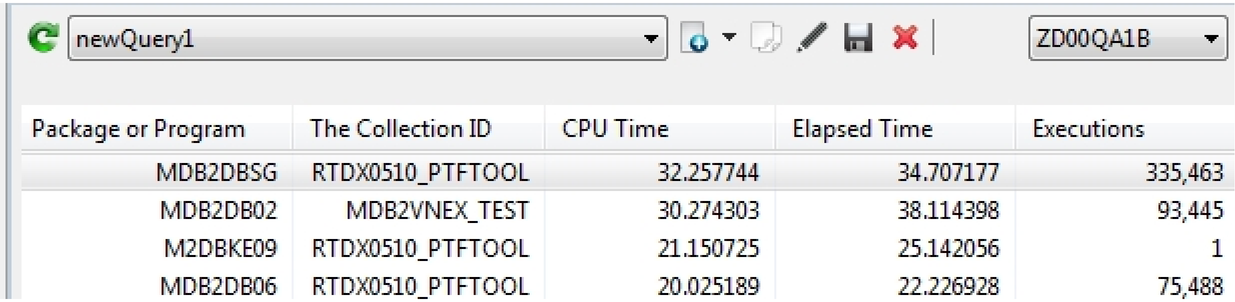
Use case Examples:



Primary Authorization ID	Number of Statements	Sum of CPU Time	Average CPU Time	Highest CPU Time
	309	207.529534	0.000179	32.257744
BOXWELL	152	7.227270	0.008166	1.192778
CHRISTO	1	0.000268	0.000268	0.000268
DUDEK	128	2.177016	0.004380	0.866522



Primary Authorization ID	Number of Statements	Sum c
	309	
BOXWEL	152	
CHRISTO	1	
DUDEK	128	
HOPPE	377	



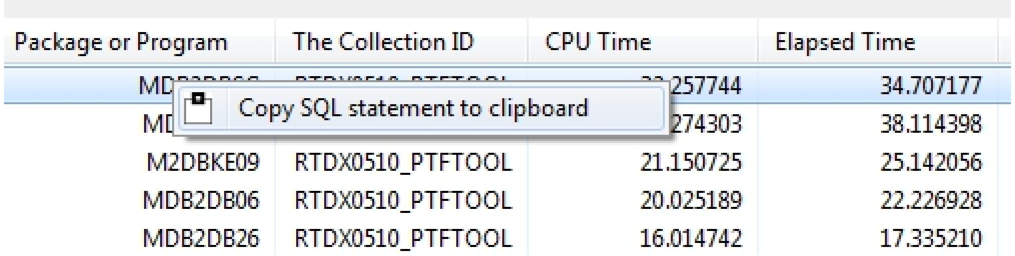
Package or Program	The Collection ID	CPU Time	Elapsed Time	Executions
MDB2DBSG	RTDX0510_PTFTOOL	32.257744	34.707177	335,463
MDB2DB02	MDB2VNEX_TEST	30.274303	38.114398	93,445
M2DBKE09	RTDX0510_PTFTOOL	21.150725	25.142056	1
MDB2DB06	RTDX0510_PTFTOOL	20.025189	22.226928	75,488

Here we have found our own bad guy! STOGROUP SQL

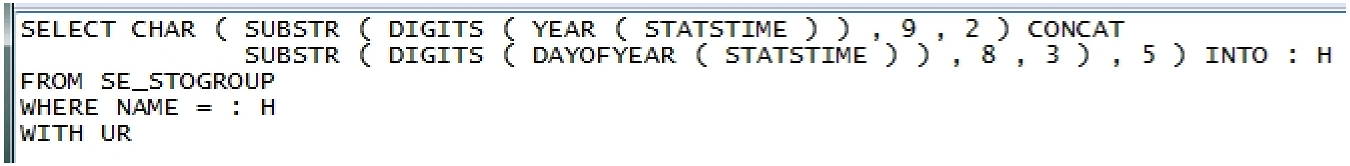
SQL WorkloadExpert on Trial

Use case Examples:

Now we need to see what it is doing…



Package or Program	The Collection ID	CPU Time	Elapsed Time
MD		257744	34.707177
MD		274303	38.114398
M2DBKE09	RTDX0510_PTFTOOL	21.150725	25.142056
MDB2DB06	RTDX0510_PTFTOOL	20.025189	22.226928
MDB2DB26	RTDX0510_PTFTOOL	16.014742	17.335210



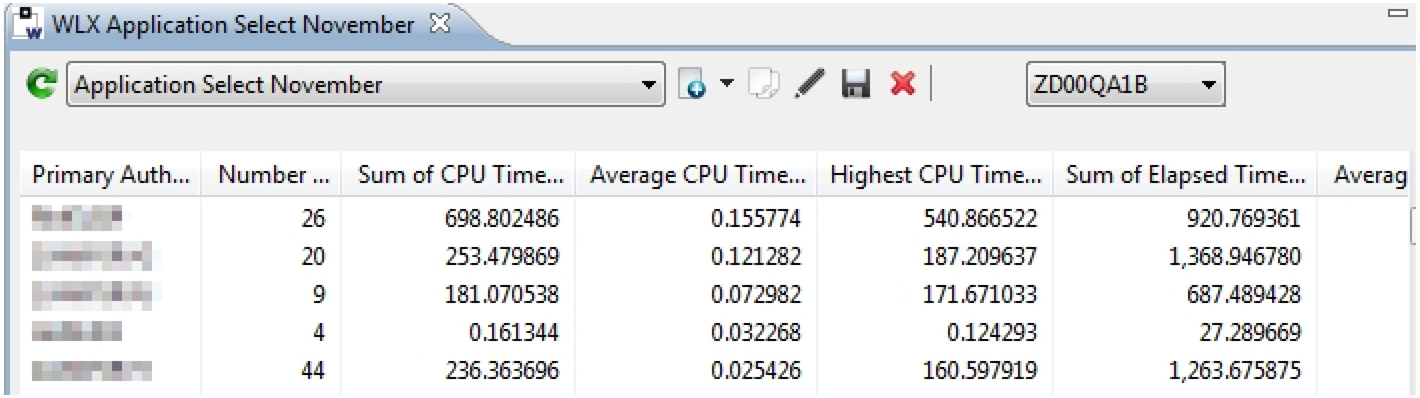
```
SELECT CHAR ( SUBSTR ( DIGITS ( YEAR ( STATSTIME ) ) , 9 , 2 ) CONCAT
              SUBSTR ( DIGITS ( DAYOFYEAR ( STATSTIME ) ) , 8 , 3 ) , 5 ) INTO : H
FROM SE_STOGROUP
WHERE NAME = : H
WITH UR
```

Aha! This looks like a great candidate for LEFT OUTER JOIN processing (Already in our next RTDX PTF by the way!)

SQL WorkloadExpert on Trial

Use case Examples:

Application Usage figures



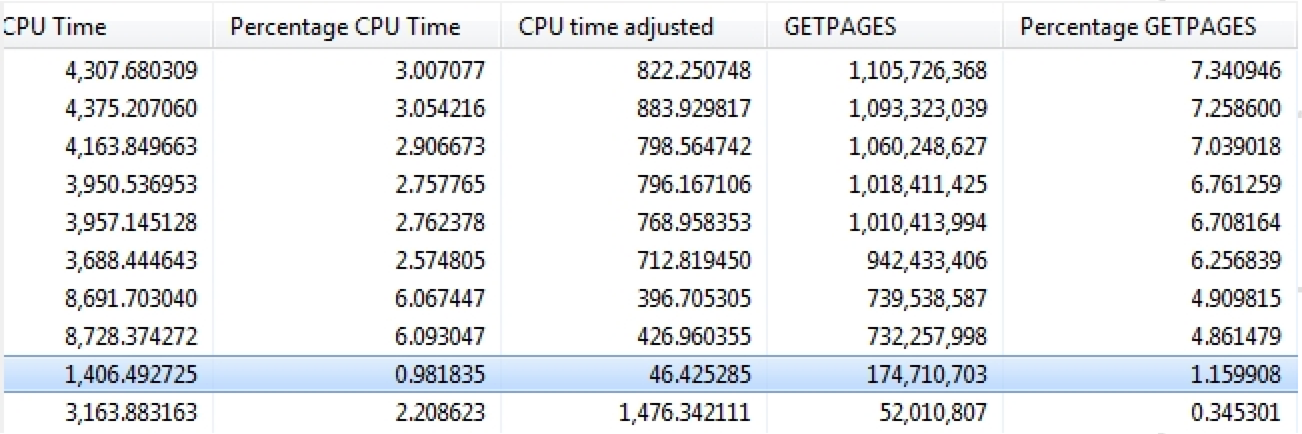
WLX Application Select November

Application Select November

ZD00QA1B

Primary Auth...	Number ...	Sum of CPU Time...	Average CPU Time...	Highest CPU Time...	Sum of Elapsed Time...	Averag
[redacted]	26	698.802486	0.155774	540.866522	920.769361	
[redacted]	20	253.479869	0.121282	187.209637	1,368.946780	
[redacted]	9	181.070538	0.072982	171.671033	687.489428	
[redacted]	4	0.161344	0.032268	0.124293	27.289669	
[redacted]	44	236.363696	0.025426	160.597919	1,263.675875	

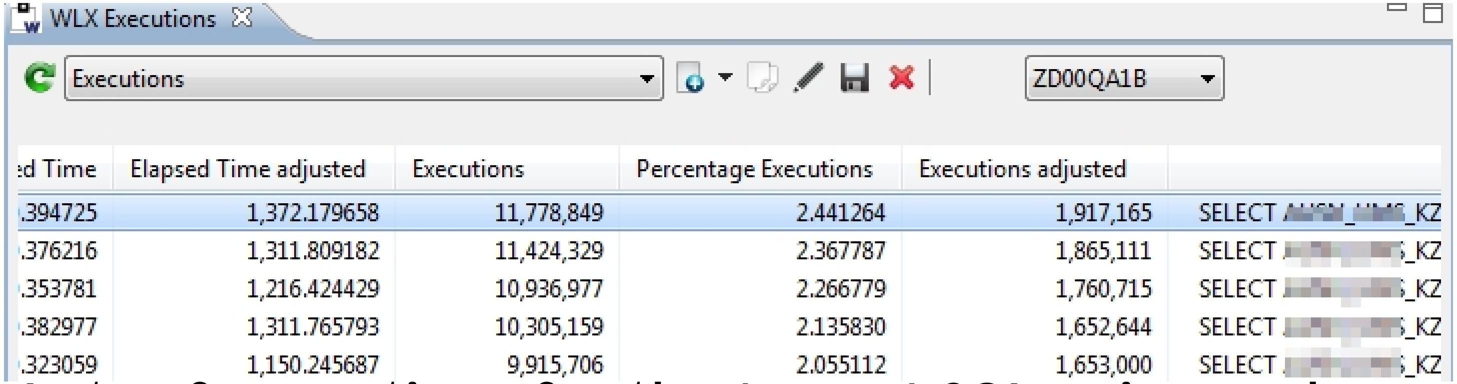
Adjusted data



CPU Time	Percentage CPU Time	CPU time adjusted	GETPAGES	Percentage GETPAGES
4,307.680309	3.007077	822.250748	1,105,726,368	7.340946
4,375.207060	3.054216	883.929817	1,093,323,039	7.258600
4,163.849663	2.906673	798.564742	1,060,248,627	7.039018
3,950.536953	2.757765	796.167106	1,018,411,425	6.761259
3,957.145128	2.762378	768.958353	1,010,413,994	6.708164
3,688.444643	2.574805	712.819450	942,433,406	6.256839
8,691.703040	6.067447	396.705305	739,538,587	4.909815
8,728.374272	6.093047	426.960355	732,257,998	4.861479
1,406.492725	0.981835	46.425285	174,710,703	1.159908
3,163.883163	2.208623	1,476.342111	52,010,807	0.345301

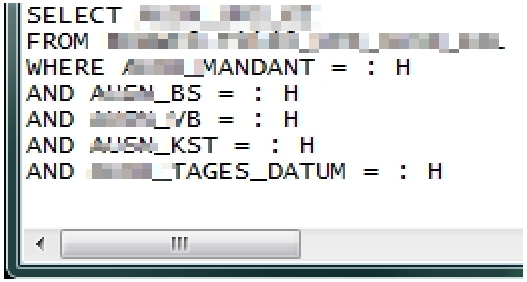
SQL WorkloadExpert on Trial

Use case Examples:



ed Time	Elapsed Time adjusted	Executions	Percentage Executions	Executions adjusted	
.394725	1,372.179658	11,778,849	2.441264	1,917,165	SELECT [redacted]_KZ
.376216	1,311.809182	11,424,329	2.367787	1,865,111	SELECT [redacted]_KZ
.353781	1,216.424429	10,936,977	2.266779	1,760,715	SELECT [redacted]_KZ
.382977	1,311.765793	10,305,159	2.135830	1,652,644	SELECT [redacted]_KZ
.323059	1,150.245687	9,915,706	2.055112	1,653,000	SELECT [redacted]_KZ

Lots of executions for the *same* SQL going on here…

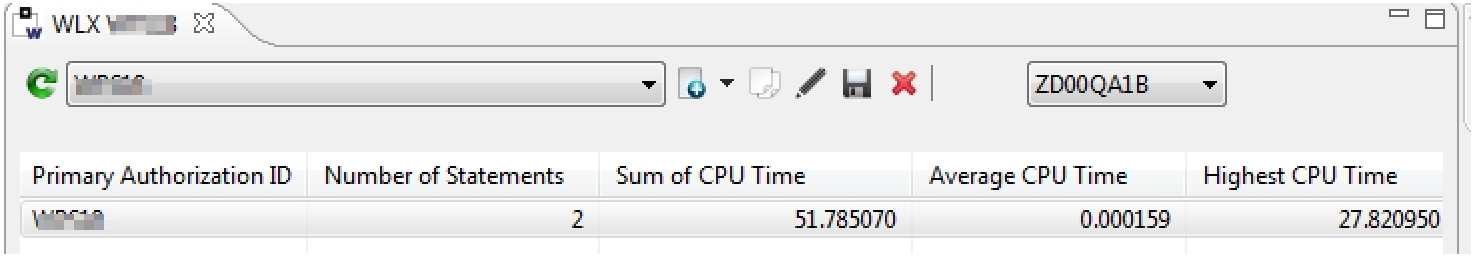


```
SELECT [redacted]
FROM [redacted]
WHERE [redacted]_MANDANT = : H
AND [redacted]_BS = : H
AND [redacted]_VB = : H
AND [redacted]_KST = : H
AND [redacted]_TAGES_DATUM = : H
```

Why so often? Discussed with developement and find it is a „design“ problem… The query could be run earlier and then only a few times a day instead of millions!

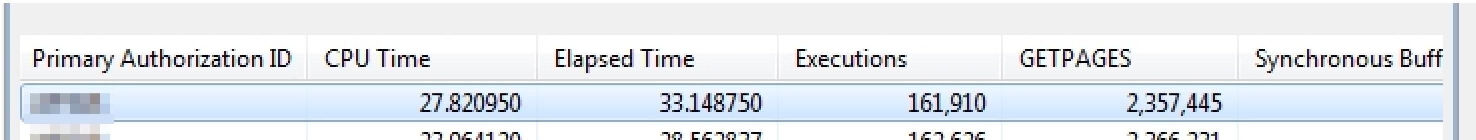
SQL WorkloadExpert on Trial

Use case Examples: Often run BAD SQL



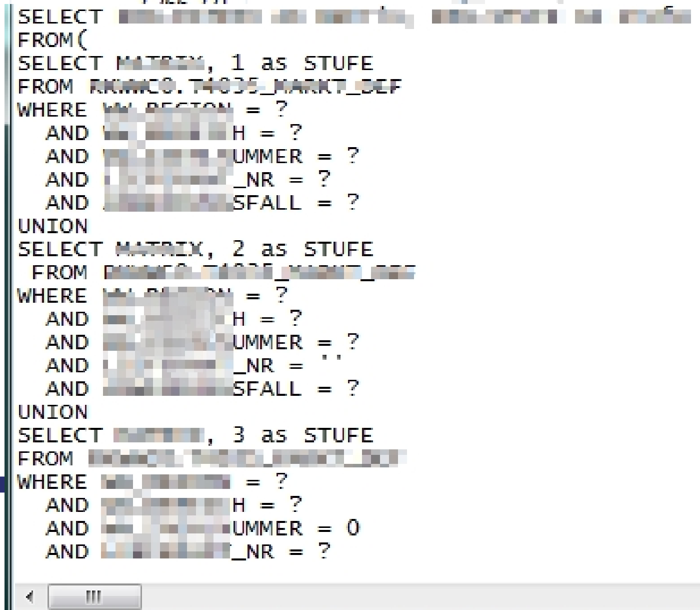
Primary Authorization ID	Number of Statements	Sum of CPU Time	Average CPU Time	Highest CPU Time
[redacted]	2	51.785070	0.000159	27.820950

This workload splits into two SQLs:



Primary Authorization ID	CPU Time	Elapsed Time	Executions	GETPAGES	Synchronous Buff
[redacted]	27.820950	33.148750	161,910	2,357,445	
[redacted]	23.964120	28.562837	162,626	2,366,221	

Which have this SQL: Six UNIONs…
DBA rewrote down to one SELECT
and IN usage.



```
SELECT [redacted]
FROM(
SELECT [redacted], 1 as STUFE
FROM [redacted]
WHERE [redacted] = ?
  AND [redacted]H = ?
  AND [redacted]UMMER = ?
  AND [redacted]_NR = ?
  AND [redacted]SFALL = ?
UNION
SELECT [redacted], 2 as STUFE
 FROM [redacted]
WHERE [redacted] = ?
  AND [redacted]H = ?
  AND [redacted]UMMER = ?
  AND [redacted]_NR = ''
  AND [redacted]SFALL = ?
UNION
SELECT [redacted], 3 as STUFE
FROM [redacted]
WHERE [redacted] = ?
  AND [redacted]H = ?
  AND [redacted]UMMER = 0
  AND [redacted]_NR = ?
```

Appendix

- Problem list and sample customer timings:
  - UK70891 – Reset stats when STOP/START MONITOR TRACE
  - UK72630 – Incorrect when executing same statement from different threads
  - UK73219 – Strange counters - HIPER
  - UK73903 – Storage leak leading to abend
  - UK78414 – Storage overlay leading to abend - HIPER
  - UK81878 – Storage leak
  - UK90073 – Difference between IFCID 401 and IFCID 58
  - UK93065 – SOS - HIPER
- Timings:
  - Externalizing every 5 mins for 24 hours cost 300 secs cpu (No EXPLAIN of course in this scenario!)

SQL WorkloadExpert

Podcast from IOD (IBM Information On Demand) about tracing and its cost:

Dan Luksetich with Florence Dubois and John Campbell

http://www.db2expert.com/podcasts/
db2_Cocktail_Hour_s1p6_perf.mp3

Performance Warehouse and Trending are No. 1!

It is voting time!!!

Please use the Chat box and list your top three Use cases (numerically please!)

E.g.. 21 , 1 , 9

From all the votes we will then calculate two or three winners to be „deep dived“ in the next (Part 3) Webinar!

Start your voting now ! ! ! ! !