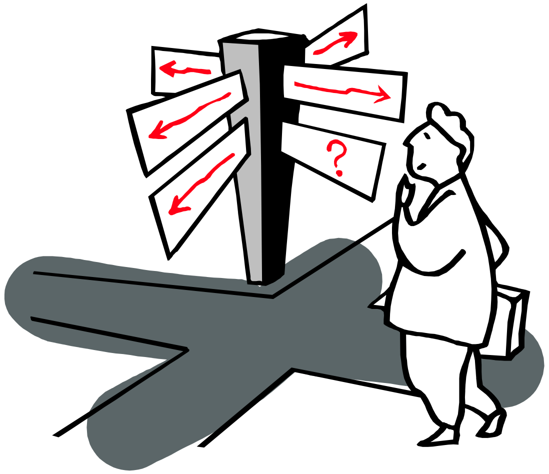
DB2 Nightshow
Facilitate WLM Functions and Tools to Improve your Monitoring

Michael Tiefenbacher

ids-System GmbH

Agenda

- **Introduction**
  - WLM Intro
  - WLM Licensing
  - WLM Architecture
- WLM Event Monitors
  - Activity event monitor
  - Statistics event monitor
- WLM Table Functions



WorkLoad Management

Why not Leverage it for Monitoring?

Workload Management Overview

- WLM is available since DB2 9.5
  - It is directly integrated in (every) DB2’s engine

- Part of Advanced Editions (AESE and AWSE)
  - But parts of the functionality can be used in all editions!

- Usage
  - Limitation, prioritisation and control of workloads and activities

WLM Licensing

Licensing requirements (Knowledge Center – WLM FAQ)

- ❖ The following workload management functions are not restricted by license:
  - **Using or altering the default service classes and workloads; this includes all monitoring capabilities**
  - Creating, altering, or dropping histogram templates
  - Using the DB2 workload management table functions or stored procedures
  - Creating, activating, stopping, or dropping workload management event monitors
  - Granting, altering, or revoking workload privileges

WLM Idea

- ❖ Identification
  - Only things you can identify can be controlled
- ❖ Management
  - Management means here to set priorities or limit resources
- ❖ Monitoring
  - Supervise your environment

⇒ Just to get the idea

⇒ Challenge: collect & understand the company requirements

⇒ We will focus on the monitoring part and how it can be used for DB2 monitoring

WLM Terms and Definitions

❖ Activity

- DML, DDL and Loads

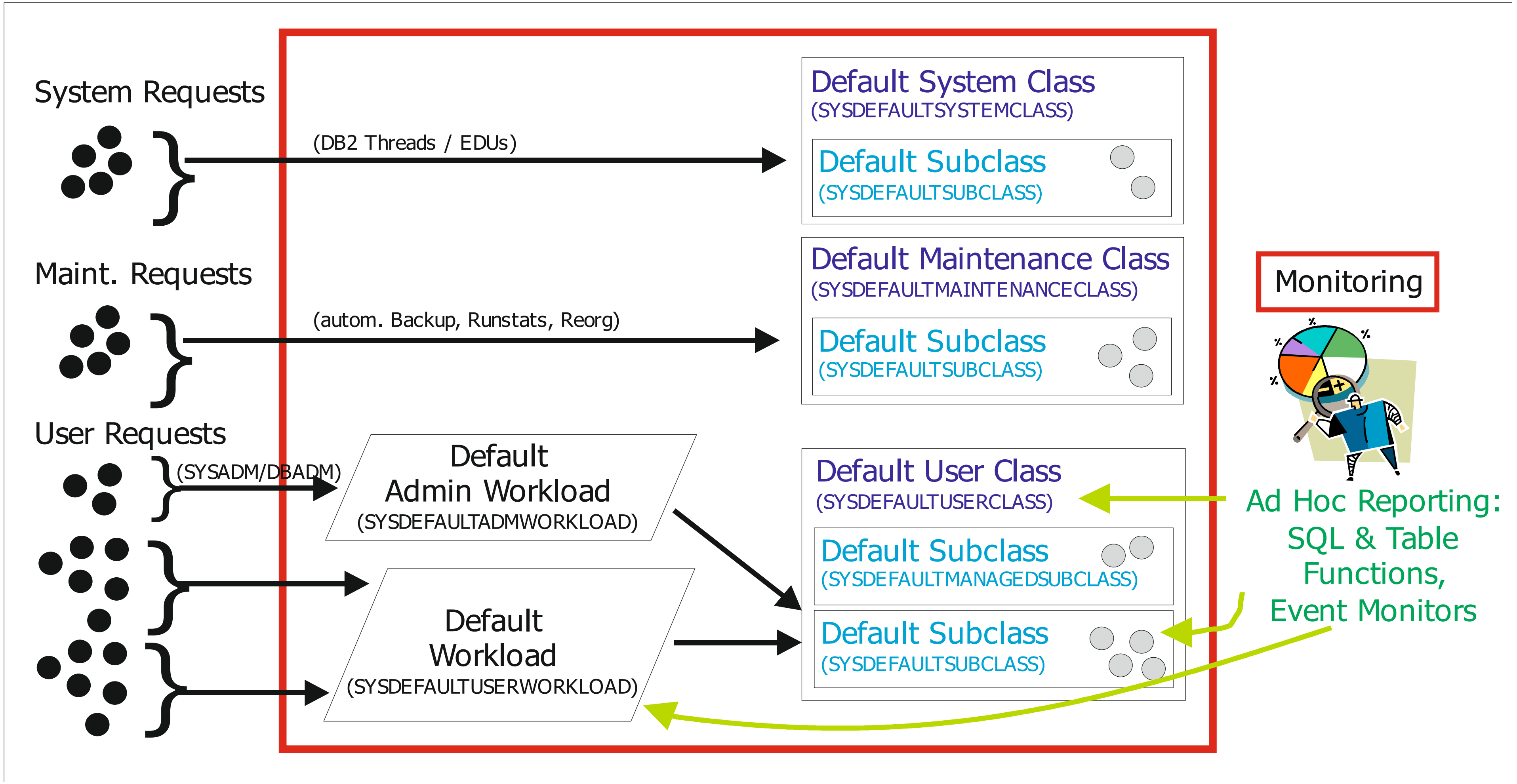
❖ Workload

- Every connection is assigned to a workload based on its connection attributes like
  - Application name, system authorization ID
- Evaluation of the defined workload in list order
- Always available: SYSDEFAULTUSERWORKLOAD

❖ Service Class

- Primary point of resource management
- It is a hierarchy with a superclass and subclass
  - superclass **1 : n** subclass
- Always available: SYSDEFAULTUSERCLASS

WLM Overview



System Requests

(DB2 Threads / EDUs)

Default System Class
(SYSDEFAULTSYSTEMCLASS)

Default Subclass
(SYSDEFAULTSUBCLASS)

Maint. Requests

(autom. Backup, Runstats, Reorg)

Default Maintenance Class
(SYSDEFAULTMAINTENANCECLASS)

Default Subclass
(SYSDEFAULTSUBCLASS)

Monitoring

User Requests

(SYSADM/DBADM)

Default
Admin Workload
(SYSDEFAULTADMWORKLOAD)

Default
Workload
(SYSDEFAULTUSERWORKLOAD)

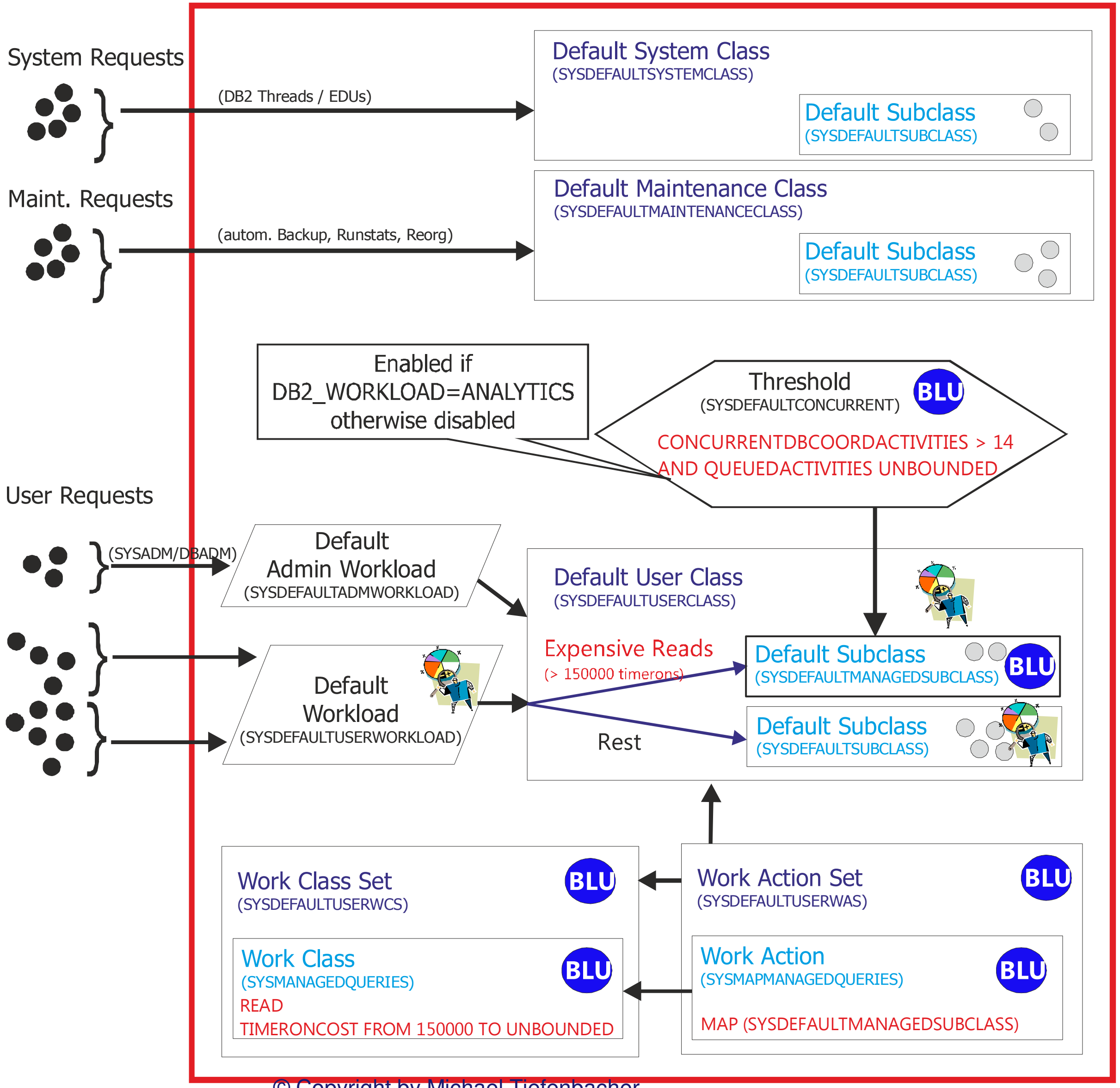
Default User Class
(SYSDEFAULTUSERCLASS)

Default Subclass
(SYSDEFAULTMANAGEDSUBCLASS)

Default Subclass
(SYSDEFAULTSUBCLASS)

Ad Hoc Reporting:
SQL & Table
Functions,
Event Monitors

WLM Overview



System Requests

(DB2 Threads / EDUs)

Default System Class
(SYSDEFAULTSYSTEMCLASS)

Default Subclass
(SYSDEFAULTSUBCLASS)

Maint. Requests

(autom. Backup, Runstats, Reorg)

Default Maintenance Class
(SYSDEFAULTMAINTENANCECLASS)

Default Subclass
(SYSDEFAULTSUBCLASS)

Enabled if
DB2_WORKLOAD=ANALYTICS
otherwise disabled

Threshold
(SYSDEFAULTCONCURRENT)

BLU

CONCURRENTDBCOORDACTIVITIES > 14
AND QUEUEDACTIVITIES UNBOUNDED

User Requests

(SYSADM/DBADM)

Default
Admin Workload
(SYSDEFAULTADMWORKLOAD)

Default
Workload
(SYSDEFAULTUSERWORKLOAD)

Default User Class
(SYSDEFAULTUSERCLASS)

Expensive Reads
(> 150000 timerons)

Default Subclass
(SYSDEFAULTMANAGEDSUBCLASS)

BLU

Rest

Default Subclass
(SYSDEFAULTSUBCLASS)

Work Class Set
(SYSDEFAULTUSERWCS)

BLU

Work Class
(SYSMANAGEDQUERIES)
READ
TIMERONCOST FROM 150000 TO UNBOUNDED

BLU

Work Action Set
(SYSDEFAULTUSERWAS)

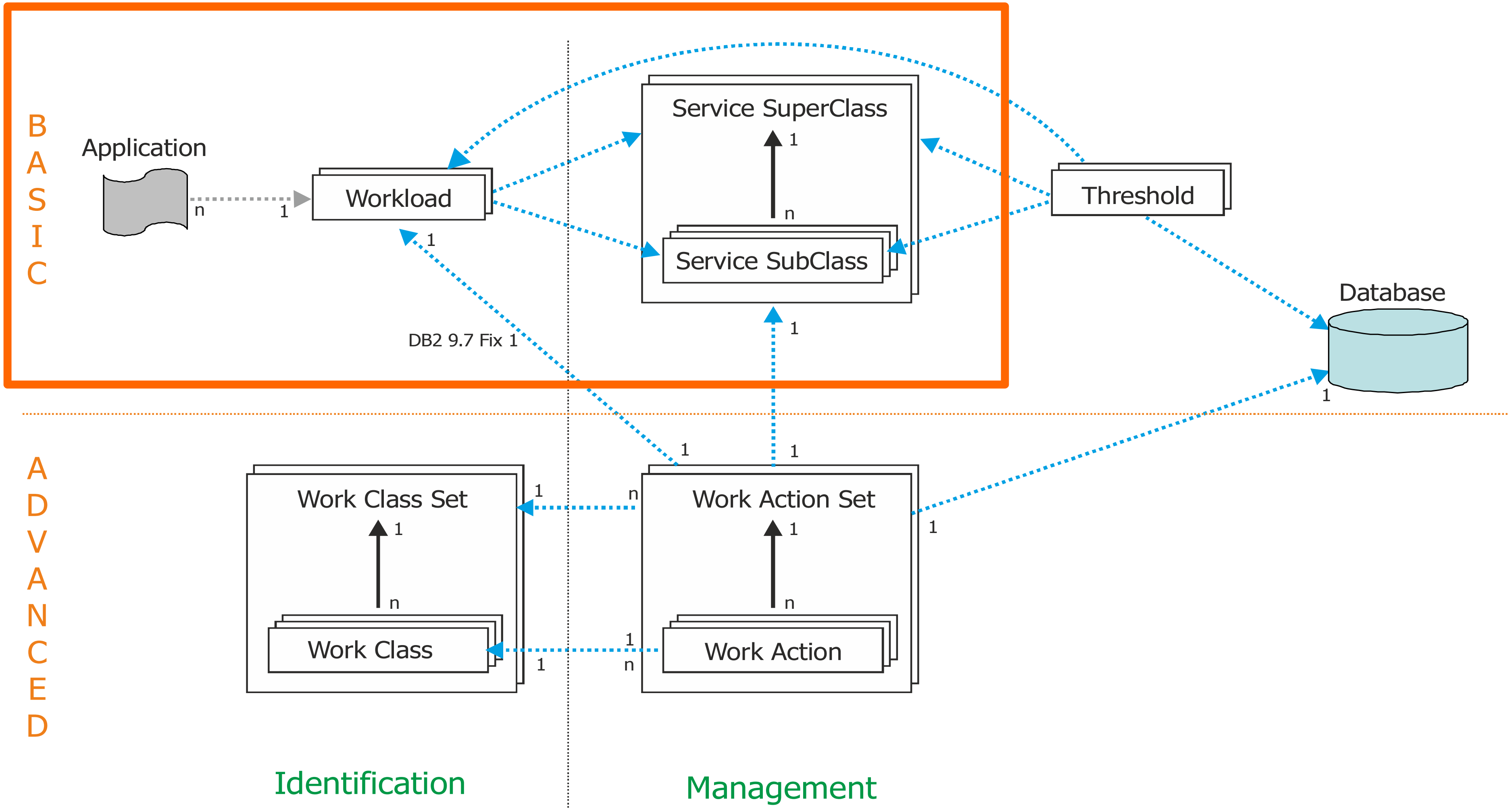
BLU

Work Action
(SYSMAPMANAGEDQUERIES)

MAP (SYSDEFAULTMANAGEDSUBCLASS)

BLU

WLM Architecture



© Copyright by Michael Tiefenbacher

Agenda

- Introduction
  - WLM Intro
  - WLM Licensing
  - WLM Architecture
- WLM Event Monitors
  - Activity event monitor
  - Statistics event monitor
- WLM Table Functions

WLM Event Monitoring

- Monitoring over a period in time (historical monitoring)
  - Typically use for event monitoring

- Available Event Monitors for WLM
  - **Activity** event monitor
    - Can capture down to SQL statement level
  - **Statistics** event monitor
    - Summarized information
  - **Threshold Violation** event monitor
    - Thresholds are WLM rules
    - Not covered here – not useful **without** a WLM license

WLM Activity Event Monitor

- SQL statement monitoring
- Only executed statements are captured
  - Less than SQL statement event monitor
- Dynamic and static SQL statements are shown

Activity Event Monitoring – Step by Step

❖ Create and start a WLM activity event monitor

No overhead yet

```
create event monitor DB2ACTIVITIES
   FOR ACTIVITIES write to table;

set event monitor DB2ACTIVITIES state 1;
```

HINT:
see script: sqllib/misc/wlmevmon.ddl

Activity Event Monitoring – Step by Step

- Alter workload to activate monitoring

```
alter workload SYSDEFAULTUSERWORKLOAD
COLLECT ACTIVITY DATA on coordinator
   WITH DETAILS AND VALUES;
```

Monitoring of activity information in this workload

WITH DETAILS: statement information gets collected
AND VALUES: incl. content of host variables – yes CONTENT!

Attention:
In case of the default workload the monitoring will be for the whole database

Improvement with WLM License

- Create additional Workloads

```
CREATE WORKLOAD Cognos
       SESSION_USER GROUP ('Deptmgr')
       APPLNAME('Cognos')
```

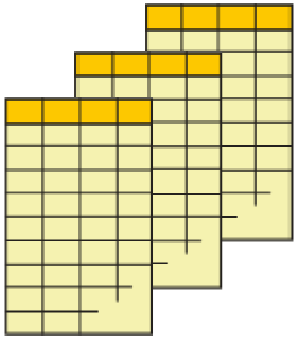
- With additional Workloads
  - No overhead
  - Better focus on specific applications or users
  - Easy to do – no complete WLM architecture necessary

WLM Event Monitor Tables

- **ACTIVITY** monitor tables:
  - CONTROL_DB2ACTIVITIES
  - ACTIVITY_DB2ACTIVITIES
  - ACTIVITYSTMT_DB2ACTIVITIES
  - ACTIVITYVALS_DB2ACTIVITIES
  - ACTIVITYMETRICS_DB2ACTIVITIES

WITH DETAILS option

WITH DETAILS
AND VALUES option

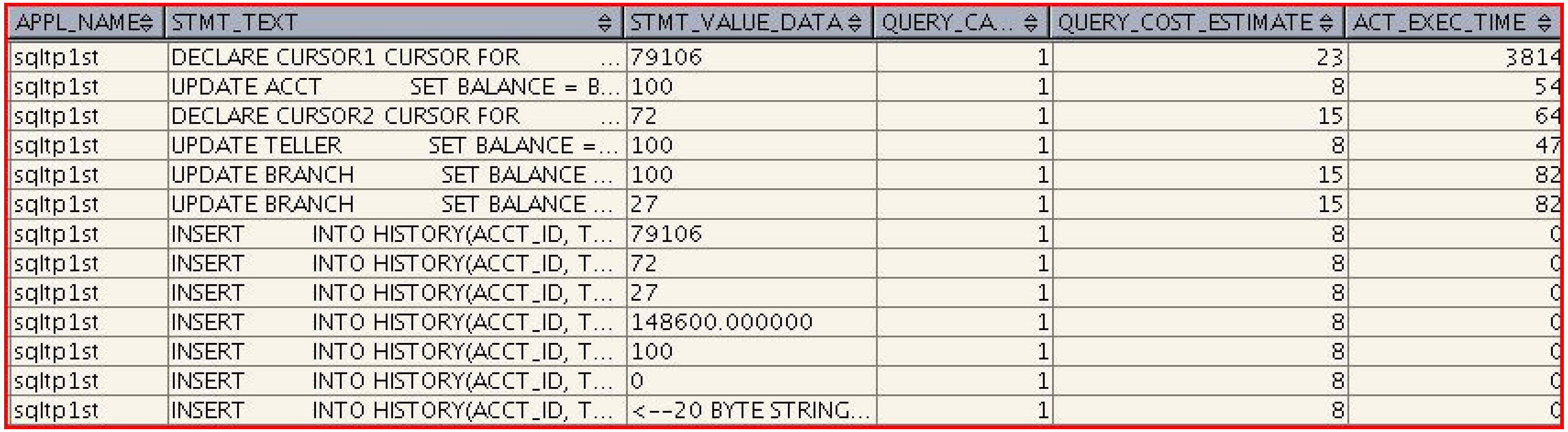


WLM Activity Event Monitor Evaluation

```
select a.AGENT_ID, a.APPL_ID, a.ACTIVITY_TYPE,
       s.STMT_TEXT, v.STMT_VALUE_DATA,
       a.QUERY_CARD_ESTIMATE, a.QUERY_COST_ESTIMATE,
       a.ACT_EXEC_TIME, a.ROWS_RETURNED
  from ACTIVITY_DB2ACTIVITIES a
 inner join ACTIVITYSTMT_DB2ACTIVITIES s
         on a.ACTIVITY_ID = s.ACTIVITY_ID
            and a.APPL_ID = s.APPL_ID and a.UOW_ID = s.UOW_ID
  left join ACTIVITYVALS_DB2ACTIVITIES v
         on a.ACTIVITY_ID = v.ACTIVITY_ID
            and a.APPL_ID = v.APPL_ID and a.UOW_ID = v.UOW_ID
```

SQL statement information

Host variable values



APPL_NAME	STMT_TEXT	STMT_VALUE_DATA	QUERY_CA...	QUERY_COST_ESTIMATE	ACT_EXEC_TIME
sqltp1st	DECLARE CURSOR1 CURSOR FOR ...	79106	1	23	3814
sqltp1st	UPDATE ACCT SET BALANCE = B...	100	1	8	54
sqltp1st	DECLARE CURSOR2 CURSOR FOR ...	72	1	15	64
sqltp1st	UPDATE TELLER SET BALANCE =...	100	1	8	47
sqltp1st	UPDATE BRANCH SET BALANCE ...	100	1	15	82
sqltp1st	UPDATE BRANCH SET BALANCE ...	27	1	15	82
sqltp1st	INSERT INTO HISTORY(ACCT_ID, T...	79106	1	8	0
sqltp1st	INSERT INTO HISTORY(ACCT_ID, T...	72	1	8	0
sqltp1st	INSERT INTO HISTORY(ACCT_ID, T...	27	1	8	0
sqltp1st	INSERT INTO HISTORY(ACCT_ID, T...	148600.000000	1	8	0
sqltp1st	INSERT INTO HISTORY(ACCT_ID, T...	100	1	8	0
sqltp1st	INSERT INTO HISTORY(ACCT_ID, T...	0	1	8	0
sqltp1st	INSERT INTO HISTORY(ACCT_ID, T...	<--20 BYTE STRING...	1	8	0

Monitor Comparison

	Statement Monitor	Activity Monitor
Created tables	CONTROL_EVMON_STATEMENTS CONNHEADER_EVMON_STATEMENTS STMT_EVMON_STATEMENTS	CONTROL_EVMON_ACTIVITES ACTIVITYMETRICS_EVMON_ACTIVITIES ACTIVITY_EVMON_ACTIVITIES ACTIVITYSTMT_EVMON_ACTIVITIES ACTIVITYVALS_EVMON_ACTIVITIES
SELECT recorded	3 statements (prepare, open, close)	one statement (select)
Note	results need to be interpreted with sqlmon.h	
SQL	static SQL is not shown (stmt_type = 1) but referenced	static and dynamic are shown
Host variable content	Not collected – only “?“ are shown	can be collected – captured in ACTIVITYVALS

Statistics Event Monitoring – Step by Step

- Create and start a WLM statistics event monitor

No overhead yet

```
create event monitor DB2STATISTICS
   FOR STATISTICS write to table;

set event monitor DB2STATISTICS state 1;
```

HINT:

see script: sqllib/misc/wlmevmon.ddl

Statistics Event Monitoring – Step by Step

- Adjust workload or service class
  - In case of the default service class the monitoring will be for the whole database

```
alter service class SYSDEFAULTSUBCLASS
under SYSDEFAULTUSERCLASS
COLLECT AGGREGATE ACTIVITY DATA BASE
```

- COLLECT ACTIVITY DATA
  collects activity information in Service Class
- COLLECT AGGREGATE ACTIVITY DATA
  aggregates information of a Subclass + histogram
- COLLECT AGGREGATE REQUEST DATA
  aggregated requests of a Subclass + histogram
- COLLECT AGGREGATE UNIT OF WORK DATA
  only collects UowLifetime histogram

Collected data in **BASE** level:

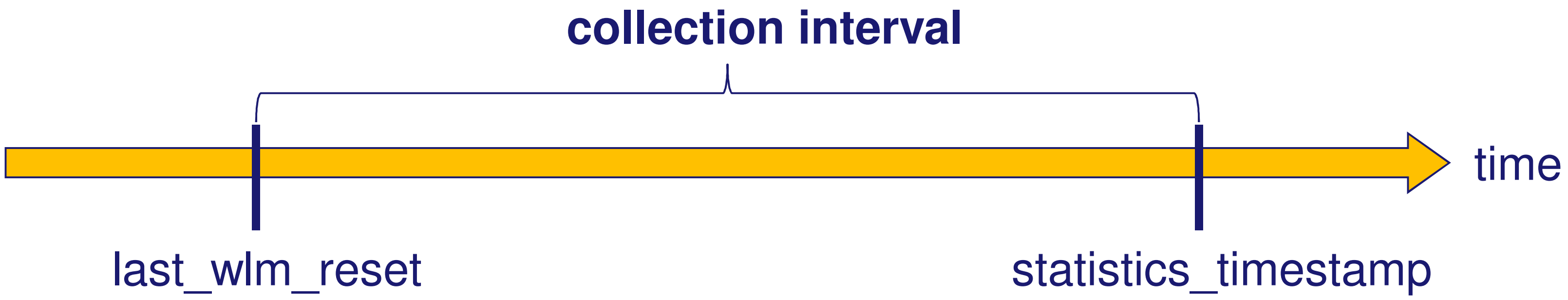
- Estimated activity cost high watermark
- Rows returned high watermark
- Temporary table space usage high watermark
- Activity life time, queue time, execution time histogram

additionally things with **EXTENDED** option

- Activity DML estimated cost histogram
- Activity DML inter-arrival time histogram

Statistics Event Monitoring – Step by Step

- Send monitor data to event monitor
  - Manually: CALL SYSPROC.WLM_COLLECT_STATS()
    CALL SYSPROC.MON_COLLECT_STATS() - in 10.5
  - Automatically: WLM_COLLECT_INT (DB CFG Parameter)
    - Default: 0 (no collection)
    - specified in minutes

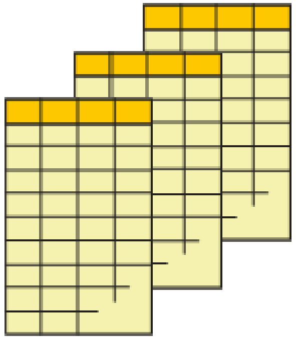


- Additionally the values are reset
  - Deactivating a database will reset them as well

Statistics Event Monitor Tables

- Tables for **STATISTICS** monitoring:
  - CONTROL
  - **SCSTATS (Service Class Stats)** (one row per subclass per interval)
  - WLSTATS (Workload Stats) (one row per workload and interval)
  - HISTOGRAMBIN (41 rows per histogram (3) per interval)
  - OSMETRICS (new in 10.5) (one row per interval)
  - SCMETRICS (one row per subclass per interval)
  - WLMMETRICS (one row per workload and interval)
  - **Less important (in our case)**
    - WCSTATS (Work Class Stats)
    - QSTATS (Queue Stats) (one row per interval)

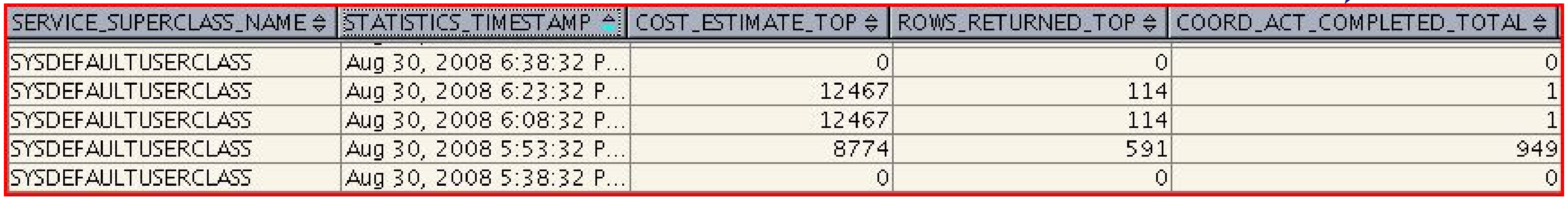
Only prefixes:
<prefix>_<monname>



Statistics Event Monitoring – Evaluation

```
select   *
   from SCSTATS_DB2STATISTICS
 where SERVICE_SUPERCLASS_NAME = ‘SYSDEFAULTUSERCLASS’
 order by STATISTICS_TIMESTAMP desc
```

Completed Activities



SERVICE_SUPERCLASS_NAME	STATISTICS_TIMESTAMP	COST_ESTIMATE_TOP	ROWS_RETURNED_TOP	COORD_ACT_COMPLETED_TOTAL
SYSDEFAULTUSERCLASS	Aug 30, 2008 6:38:32 P...	0	0	0
SYSDEFAULTUSERCLASS	Aug 30, 2008 6:23:32 P...	12467	114	1
SYSDEFAULTUSERCLASS	Aug 30, 2008 6:08:32 P...	12467	114	1
SYSDEFAULTUSERCLASS	Aug 30, 2008 5:53:32 P...	8774	591	949
SYSDEFAULTUSERCLASS	Aug 30, 2008 5:38:32 P...	0	0	0

(only subset)

Statistics Event Monitor

- Workload Statistics via WLSTATS_DB2STATISTICS
  - Many TOP-values (ACT_ROWS_READ_TOP etc.)
  - COORD_ACT_COMPLETED_TOTAL
  - COORD_ACT_EST_COST_AVG
  - COORD_ACT_EXEC_TIME_AVG
  - COORD_ACT_LIFETIME_AVG

- Service Class Statistics via SCSTATS_DB2STATISTICS
  - Many results overlap with WLSTATS
  - CONCURRENT_ACT_TOP
  - CONCURRENT_CONNECTION_TOP
  - COORD_ACT_COMPLETED_TOTAL

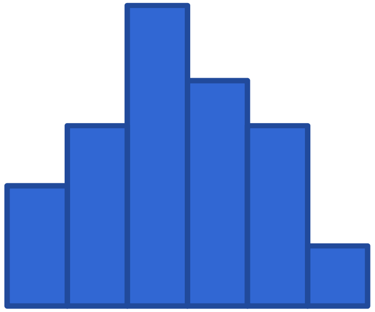
Histograms

Histogram Definition

- A **histogram** is a graphical representation of the distribution of numerical data. To construct a histogram, the first step is to split the range of values into bins —that is, divide the entire range of values into a series of small intervals—and then count how many values fall into each interval. (Wikipedia)

Histograms in DB2

- 41 Bins/Intervals (40 regular intervals+ 1 infinite Bin)
- bin ranges follow a logarithmic scale
- Units (depends on histogram): here milliseconds
- Default upper limit is 21600000 (= 6 hours)
- Can be individually created:
  CREATE HISTOGRAM Template

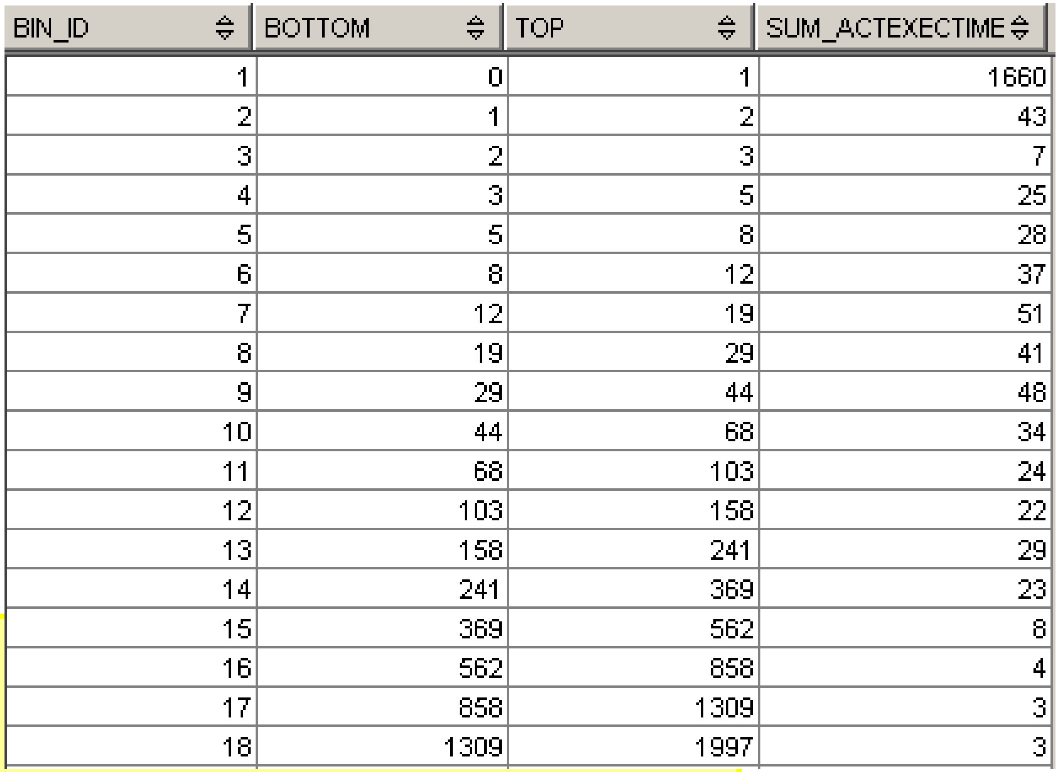


Histograms via HISTOGRAMBIN

Histogram	Histogram_type	Prerequisite
ACTIVITY LIFETIME	CoordActLifetime	COLLECT AGGREGATE ACTIVITY DATA base
ACTIVITY QUEUETIME	CoordActQueueTime	COLLECT AGGREGATE ACTIVITY DATA base
ACTIVITY EXECUTETIME	CoordActExecTime	COLLECT AGGREGATE ACTIVITY DATA base
REQUEST EXECUTETIME	ReqExecTime	COLLECT AGGREGATE REQUEST DATA base
ACTIVITY ESTIMATEDCOST	CoordActEstCost	COLLECT AGGREGATE ACTIVITY DATA extended
ACTIVITY INTERARRIVALTIME	CoordActInterArrivalTime	COLLECT AGGREGATE ACTIVITY DATA extended
UOW LIFETIME	UowLifetime	COLLECT AGGREGATE UNIT OF WORK DATA base

Histograms

- Example



BIN_ID	BOTTOM	TOP	SUM_ACTEXECTIME
1	0	1	1660
2	1	2	43
3	2	3	7
4	3	5	25
5	5	8	28
6	8	12	37
7	12	19	51
8	19	29	41
9	29	44	48
10	44	68	34
11	68	103	24
12	103	158	22
13	158	241	29
14	241	369	23
15	369	562	8
16	562	858	4
17	858	1309	3
18	1309	1997	3

```
SELECT bin_id,
       bottom,
       top,
       sum(number_in_bin) as Sum_ActExecTime
  FROM DB2ADMIN."HISTOGRAMBIN_DB2STATISTICS"
 WHERE histogram_type = 'CoordActExecTime'
 GROUP BY bin_id, bottom, top
```

Usage

- Differentiate load between
  - Service classes
  - Times

- Prove your service level
- Show how many statements are successfully processed
  - to show you did a great job

Monitor Comparison

	Event Monitor	WLM
Evaluation via SQL	✓	✓
Filtering	**Auth. ID,** **Appl. ID,** **Appl. Name**	**Workload** **(Connection attribute),** **Service Class** **(Statement)**
Content of host variables	✗	✓
Monitor switches necessary	✓	✓
Performance details for SQL	++	++
Generated system load	**medium to big**	**medium**
Static SQL available?	**indirect**	✓
No extra cost	✓	**partially**

	Event Monitor	WLM
Bufferpool Monitoring	✓	✓
Rows Read / Written / Returned	✓	✓
Sort	✓	✓
Runstats / Statistics Info Fabricated Stats	✓	✓
System CPU Time	✓	✓
Prepare Time etc.	✗	✓
Execution Time	✓	✓
Number of Executions	**indirect**	**indirect**

WLM Clauses – Event Monitor Mapping

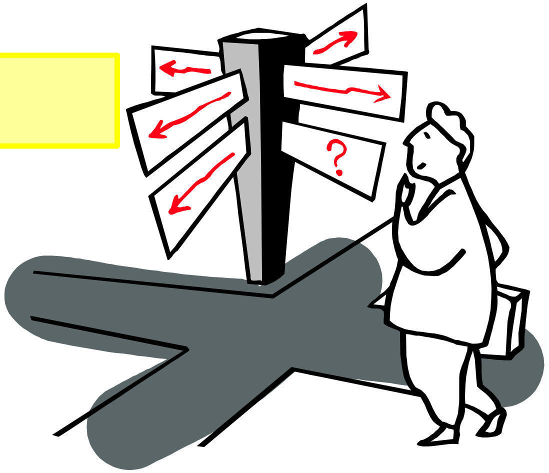
Object	Monitor clause	Event Monitor
Workload	COLLECT ACTIVITY DATA	ACTIVITY
	COLLECT ACTIVITY METRICS	-
	COLLECT AGGREGATE ACTIVITY DATA	STATISTICS
	COLLECT AGGREGATE UNIT OF WORK DATA	
	COLLECT LOCK TIMEOUT DATA	LOCKING
	COLLECT DEADLOCK DATA	
	COLLECT LOCK WAIT DATA	
	COLLECT UNIT OR WORK DATA	UNIT OF WORK

WLM Clauses – Event Monitor Mapping

Object	Monitor clause	Event Monitor
Service **Sub**class	COLLECT ACTIVITY DATA	ACTIVITY
	COLLECT AGGREGATE ACTIVITY DATA	STATISTICS
	COLLECT AGGREGATE REQUEST DATA	
	COLLECT AGGREGATE UNIT OF WORK DATA	
Service **Super**class	COLLECT REQUEST METRICS	
		UNIT OF WORK

Agenda

- Introduction
  - WLM Intro
  - WLM Licensing
  - WLM Architecture
- WLM Event Monitors
  - Activity event monitor
  - Statistics event monitor
- WLM Table Functions



WLM Monitoring – Table Functions

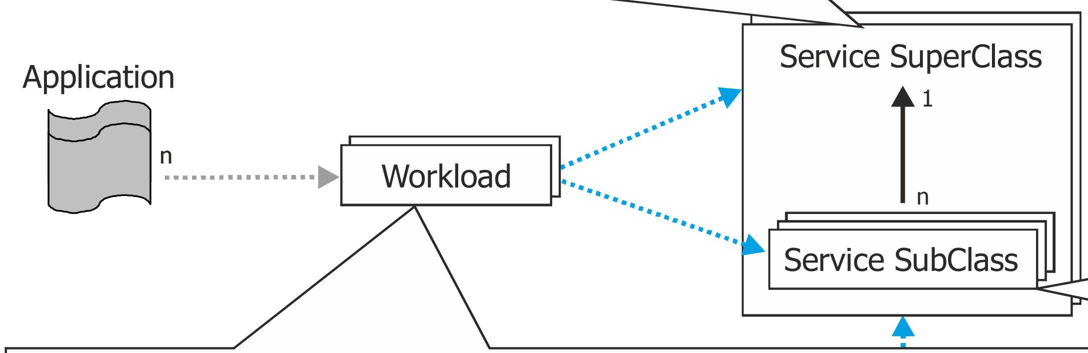
- Lots of table functions for real time monitoring
- The returned metrics are similar to the ones of the event monitor
  - Additional metrics with STDDEV
    - i.e. COORD_ACT_LIFETIME_STDDEV
  - Less in the area of locking or tempspace usage
    - i.e. TEMP_TABLESPACE_TOP
- Related DB CFG parameter
  - mon_act_metrics
  - mon_req_metrics

WLM Table Functions – Common Points

- *_STATS table functions always return a LAST_RESET column
  - since last WLM_COLLECT_STATS
- *_DETAILS returns metrics as XML in a BLOB column
- Last option is always “Member”
  - -1: current member (also set implicitly)
  - -2: all active database members

WLM Table Functions – Overview

MON_GET_SERVICE_SUPERCLASS_STATS(", -2) -- basic statistics for one or more Service Super Classes
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES(", ", -2) -- lists all running workload occurrences of a Service Class
MON_GET_AGENT(", ", NULL, -2) -- lists agents and processes (db2fmp, prefetcher, cleaner) running in the Service Class
MON_SAMPLE_SERVICE_CLASS_METRICS(", ", ", ", 30, -2) -- collects Service class data of multiple DBs and shows the differences per intervall (30s)



MON_GET_SERVICE_SUBCLASS_STATS(", ", -2)
-- Statistics for one or more Service Subclasses
MON_GET_SERVICE_SUBCLASS(", ", -2)
-- Metrics for one or more Service Subclasses
MON_GET_SERVICE_SUBCLASS_DETAILS(", ", -2)
-- **XML** metrics for one or more Service Subclasses

MON_GET_WORKLOAD_STATS(", -2) -- Statistics of one or more Workloads
MON_GET_WORKLOAD(", -2) -- Metrics for one or more Workloads
MON_GET_WORKLOAD_DETAILS(", -2) -- **XML** metrics for one or more Workloads
MON_GET_ACTIVITY(NULL, -2) -- lists Activities of an application
└→ MON_GET_ACTIVITY_DETAILS(<apphandl>, <uow_id>, <activity_id>, -2)
-- **XML** metrics of an activity (incl. statements)
└→ MON_CAPTURE_ACTIVITY_IN_PROGRESS(<apphandl>, <uow_id>, <activity_id>)
-- sends activity data to an activity event monitor
└→ WLM_CANCEL_ACTIVITY(<apphandl>, <uow_id>, <activity_id>)
-- Cancels an Activity
MON_SAMPLE_WORKLOAD_METRICS(", ", ", 30, -2) -- Workload metrics diffs in intervall

RED: table function
BLUE: stored procedure

Further Information I

- WLM FAQ
  - http://www-01.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.admin.wlm.doc/doc/c0052604.html?lang=en
  - http://goo.gl/B5JZKA

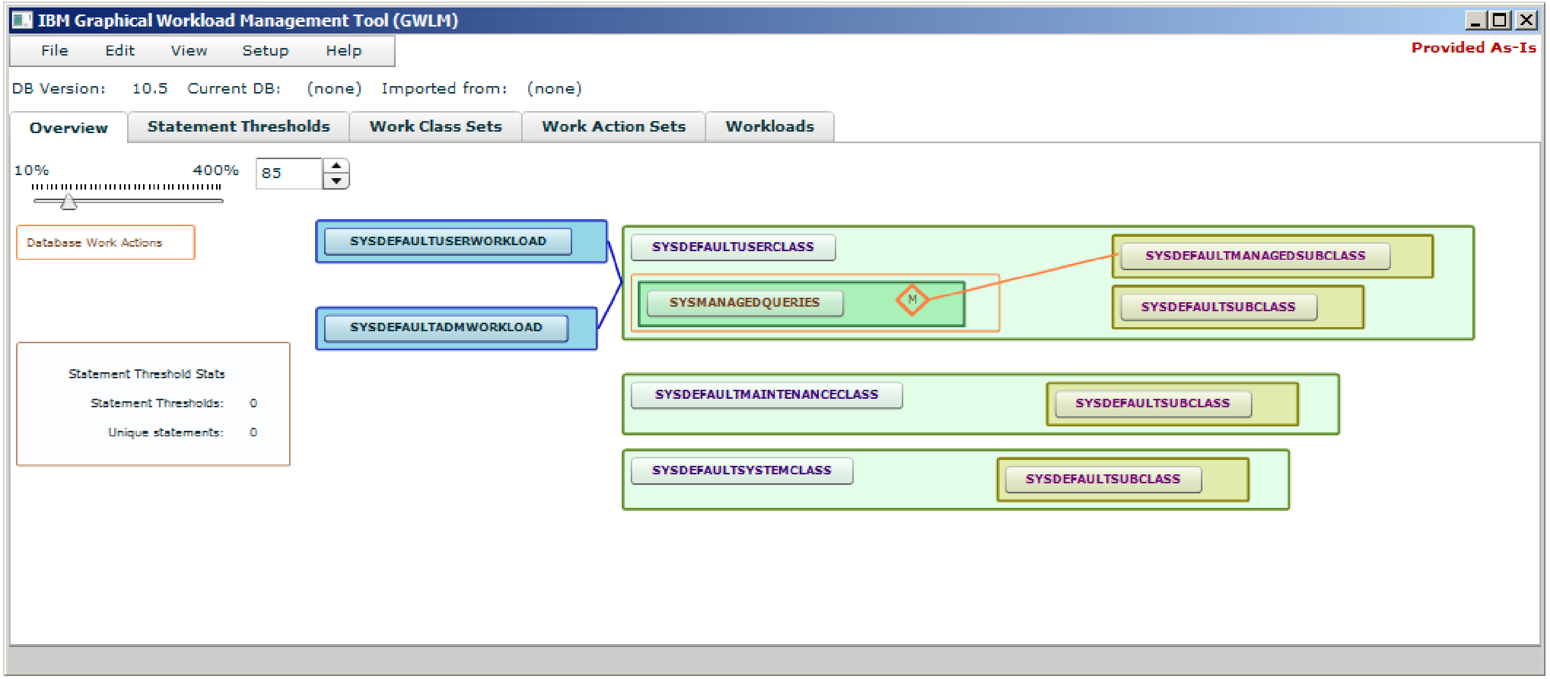
- WLM Best Practices
  - https://www.ibm.com/developerworks/community/blogs/SusanVisser/entry/new_db2_wlm_best_practice_paper1?lang=en_us
  - https://goo.gl/60k8JT

Further Information II

- Articles: DB2 workload management histograms,
  - Part 1: A gentle introduction to histograms http://download.boulder.ibm.com/ibmdl/pub/software/dw/dm/db2/dm-0810mcdonald/dm-0810mcdonald-pdf.pdf
  - Part 2: Understanding the six histograms of DB2 workload management http://download.boulder.ibm.com/ibmdl/pub/software/dw/dm/db2/dm-0810mcdonald2/dm-0810mcdonald2-pdf.pdf
  - Part 3: Visualizing and deriving statistics from DB2 histograms using SQL http://download.boulder.ibm.com/ibmdl/pub/software/dw/dm/db2/dm-0810mcdonald3/dm-0810mcdonald3-pdf.pdf

Further Information III

- New Tool GWLM on DeveloperWorks
  - Authors: Kevin Beck and Paul Bird
  - Needs Java and Adobe Air
  - https://www.ibm.com/developerworks/community/groups/service/html/communityview?communityUuid=87992700-9b53-4137-83a5-1ed837e04858
    or https://goo.gl/d30Rb7



DB Version: 10.5 Current DB: (none) Imported from: (none)

Statement Threshold Stats
Statement Thresholds: 0
Unique statements: 0

WorkLoad Management

Why not Leverage it for Monitoring?

We Love Monitoring

Michael Tiefenbacher

ids-System GmbH

m.tiefenbacher@ids-system.de

Twitter: @globomike