

Performance analysis of Oracle databases with Panorama



Otto Group Solution Provider (OSP) GmbH



Founded:

March 1991

Parent company:

OTTO Group

Locations:

Dresden, Hamburg, Burgkunstadt, Madrid, Taipeh, Bangkok

Number of employees:

Ca. 300

Managing directors:

Dr. Stefan Borsutzky, Norbert Gödicke, Jens Gruhl

Website:

<https://www.osp.de>

About me



Peter Ramm

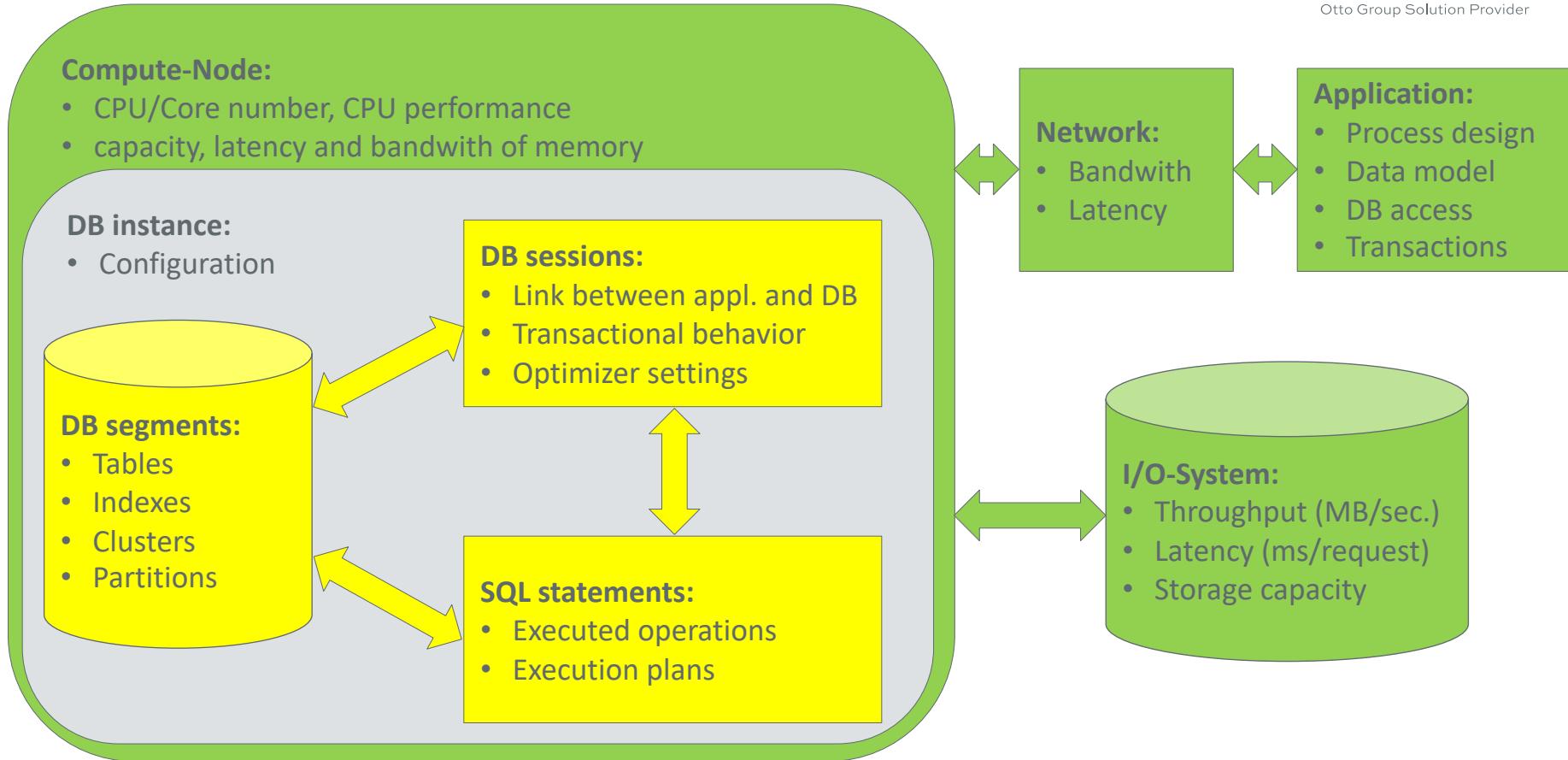
Team lead for strategic-technical consulting at OSP Dresden

> 30 years of history in IT projects

Main focus:

- Development of OLTP systems based on Oracle databases
- From architecture consulting up to trouble shooting
- Performance optimization of existing systems

Factors influencing performance in DB usage



Agenda

- Presentation of the used tool "Panorama"
- Top/down analysis of session activities
- Executed SQL statements with detail analysis
- Segment statistics: Recording of characteristic values for objects (tables, indexes ...)
- Capacity and utilization of CPU, memory and I/O system, configuration of the instance
- Dragnet investigation: Systematic scan of the system for performance antipatterns

Agenda

- Presentation of the used tool "Panorama"
- Top/down analysis of session activities
- Executed SQL statements with detail analysis
- Segment statistics: Recording of characteristic values for objects (tables, indexes ...)
- Capacity and utilization of CPU, memory and I/O system, configuration of the instance
- Dragnet investigation: Systematic scan of the system for performance antipatterns

Panorama for Oracle databases



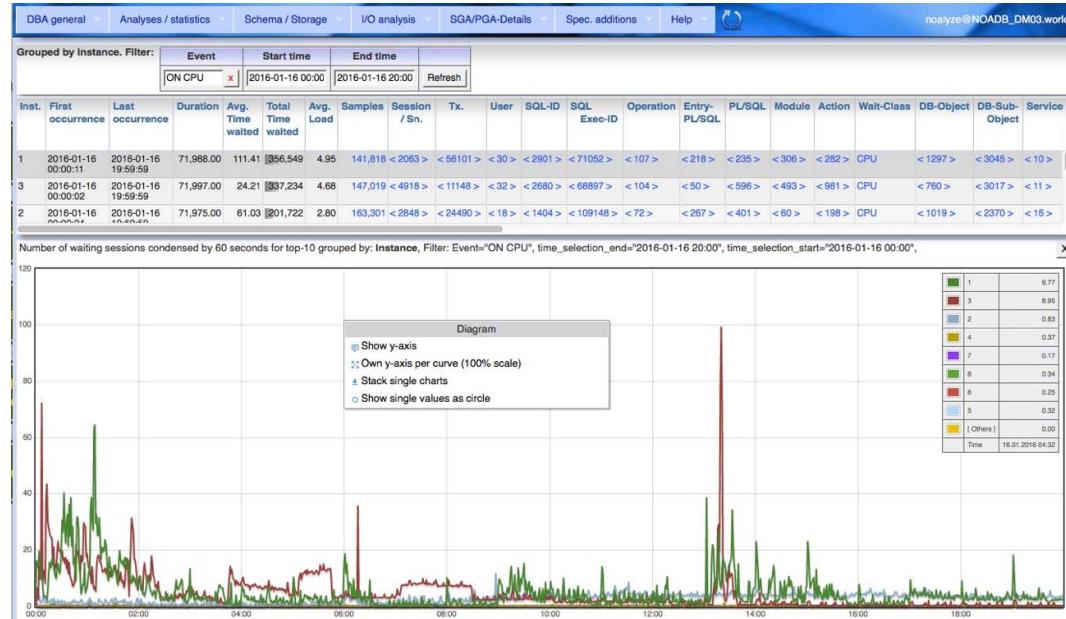
Free available tool for performance analysis of Oracle DB

Based on AWR data or Panorama's own sampling (also with Standard Edition or without Diagnostics Pack)

Docker image or self-booting war file

Panorama accesses your DB on a read-only basis and does not install any PL/SQL objects of its own. So you can test the functions without any risk.

Description of Panorama incl. download link :
Oracle performance analysis blog :
Various slides on the use of Panorama :

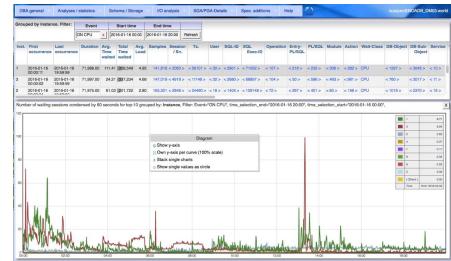


<https://rammpeter.github.io/panorama.html>
<https://rammpeter.blogspot.com>
<https://www.slideshare.net/PeterRamm1>

Panorama for Oracle: Motivation

Focus on:

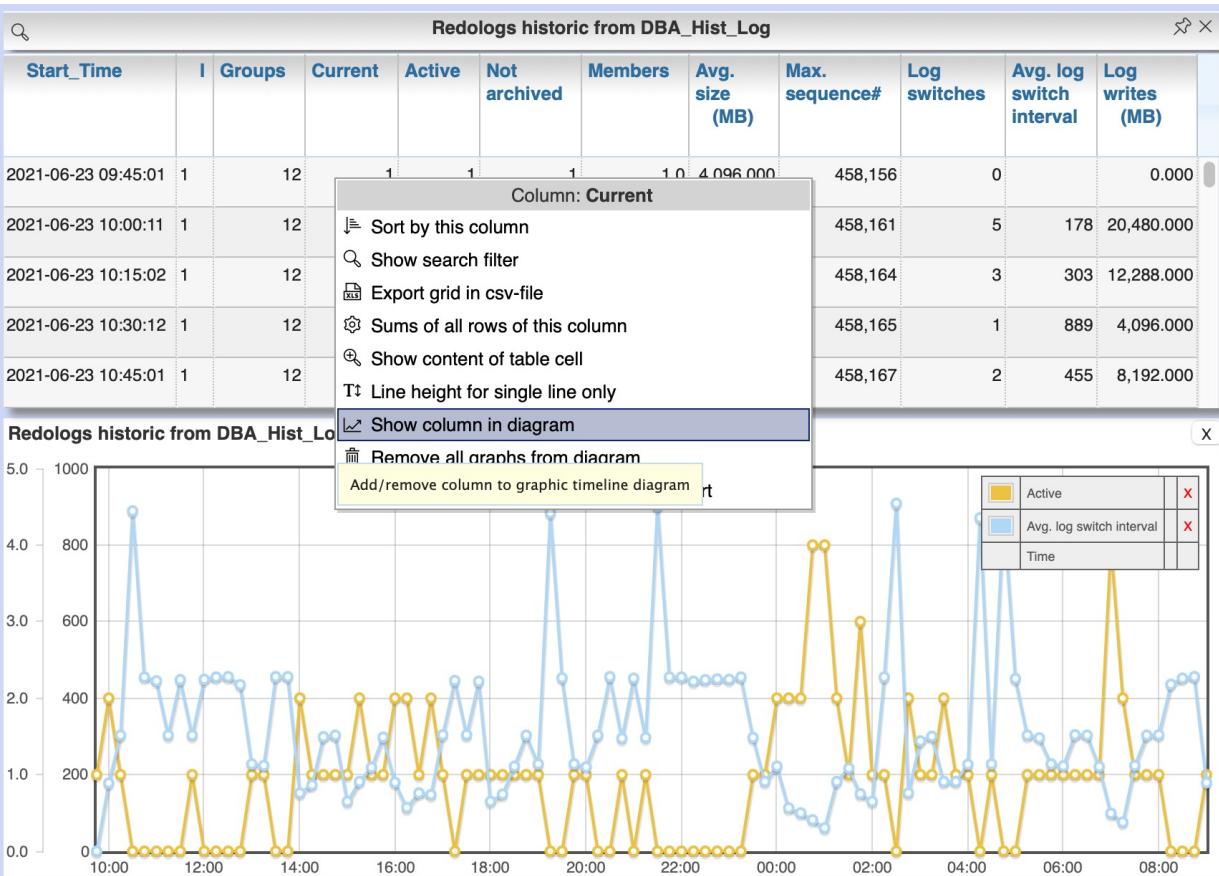
- Preparation of complex interrelationships of DB without deep insider knowledge
- Support of an analysis workflow by linking the individual steps on web GUI as an alternative to loving collection of individual SQL scripts
- Drilldown in root cause identification starting from concrete problem points
- Offline analysis with time distance to the problem under investigation
- Lowering the barriers to actually getting to the bottom of problems in detail



Differentiation from other established monitoring tools :

- Panorama does not claim to cover all facets of monitoring and visualization of Oracle DB internals
- Functions usually found inclusion in Panorama when they are:
 - not or only insufficiently offered by the established tools
 - Existing tools are not accessible for normal users (e.g. because of costs)
- A useful application is not instead of, but in combination with e.g. EM Cloud Control etc.

Forms of presentation



Workflow in browser page
continuously downwards

Results usually as tables

Time-related values can generally
also be displayed as graphs by
showing/hiding columns of the
tables

Visualization in graphs via context
menu (right mouse button)

Environment variable
PANORAMA_LOG_LEVEL=debug
shows executed SQLs on console

Agenda

- Presentation of the used tool "Panorama"
- Top/down analysis of session activities
- Executed SQL statements with detail analysis
- Segment statistics: Recording of characteristic values for objects (tables, indexes ...)
- Capacity and utilization of CPU, memory and I/O system, configuration of the instance
- Dragnet investigation: Systematic scan of the system for performance antipatterns

Sessions currently connected to DB

Menu: „DBA general“ / „sessions“

Show sessions

Active only Parallel Query User only DB links only Inst. Filter Show sessions

I	SID/SN	Status	SQL_ID	Wait event	Wait time	User	Proc	Machine	C-User	C-Proc	Program	Client-Info	Client ID	Module	Action	Service	Logon Time	Last Active Start	I/O-Index
2	5649, 25284	ACTIVE		ON CPU	0.0	NOA	96382	coreprod025	core	1234	JDBC Thin Client			OFMSGType_	process OFMessages	NOADB_LOG	2021-06-22 09:29:48	2021-06-24 09:34:24	556,476
2	5676, 52716	ACTIVE	7xbj30xa9v971	SQL*Net message from client	0.0	NOA	319141	coreprod016	core	1234	JDBC Thin Client	AppExecID = 11058136 WsMethodID = 236		ID_Application = 128	ID_WSMetho = 236	NOADB_ONLI	2021-06-20 00:45:51	2021-06-24 09:34:24	776,876

Details for session SID=5649, Serial#=25284, Instance=2, 2021-06-24 09:34:55

Status	Client-Info	Module	Action	Username	Audit-SID	Sh. Proc.	PID	Machine	OSUser	Process	Program	Logon-time	Last active start	Tx. ID	Auth. type	Client char set	Client connection	Client OCI lib	Client version		
ACTIVE		OFMSGType_	process OFMessage	NOA			327340	96382	3143	coreprod02	core	1234	JDBC Thin Client	2021-06-22 09:29:41	2021-06-24 09:34:54	6E00020012EAFC06	DATABASE	Unknown	Heterogeneous	Unknown	12.1.0.2.0

SQL-ID		C.	SQL exec start		SQL exec ID		SQL-Text	
Aktuelles SQL-Statement	gp109ra3nwca1		1	2021-06-24 09:34:53			42,927,715 BEGIN AUFRAG.AU_SE_FKT_NOA_FAKTURA_IW_JL.fakturiere(:1 ,	
Vorheriges SQL-Statement	9anb1gf84m4tm		44	2021-06-24 09:34:53			47,947,422 INSERT /*+ APPEND "UT_MO_BUSINESS_LOG.schreibe_log" */	

Process Memory (incl. PQ-Server) from GVSProcess_Memory

Category	Allocated	Used	Max Allocated
Freeable	2,162,688	0	
PL/SQL	9,350,864	737,008	23,969,056
Other	19,920,268		19,920,268
SQL	3,528,936	38,960	7,541,472

Wait-Status Locks Temp-Usage 612 open cursor Objects accessed Active Session History Session-Statistics Audit Trail Optimizer Env. SQL-Monitor (77)

Listing of DB sessions with optional filter and drilldown into session and SQL details

Filter and sort sessions by several criterias

Retrospective analysis of the workload

- Evaluating and analyzing the workload of interest of a DB is rarely possible live
 - Usually an escalation occurs more or less delayed to the triggering event
 - The past cannot be inferred from the current load situation of a DB
-
- Some information about executed SQLs can be found out from the SGA (v\$SQL ...) as well as e.g. via StatsPack

Oracle's builtin solution: "Active Workload Repository" (AWR) and "Active Session History" (ASH)

- Historization of various operating states out of the box
- AWR: recording at hourly intervals with retention for 7 days (customizable)
- ASH: Recording of active sessions every second with condensation to 10 seconds
- Available only for Enterprise Edition with additional licensing of the "Oracle Diagnostics Pack"
- Caution: Also accessible without license but access means license violation

And what about Standard Edition or without Diagnostics Pack?

Panorama provides its own recording of the workload (Panorama-Sampler):

- Structurally identical to Oracle's AWR and ASH tables
- Usable without EE / Diagnostics Pack also for Standard Edition or Express Edition
- Recording in tables in local schema of DB
- Panorama transparently uses either AWR data or its own workload recording
- Other sampler functions not included in AWR :
 - Cache usage history, history of object sizes, blocking locks, long-term trend of usage

- Integrated in Panorama
- Activate recording via GUI
- Unlimited number of observed DB instances

Database connect info	AWR and ASH	Object size history	DB-cache history	Blocking locks history	Long-term trend
Sample AWR and ASH	<input checked="" type="checkbox"/>				
Snapshot cycle (minutes)	60				
Snapshot retention (days)	30				
Limits for SQL-history	Min. executions	20	Min. runtime (ms)	200	
<input type="button" value="Test connection"/> <input type="button" value="Save"/> <input type="button" value="Cancel"/>					

More details about Panorama-Sampler: https://rammpeter.github.io/panorama_sampler.html

Range of functions of the Panorama-Sampler

For the following AWR views with history recording there is an equivalent in the Panorama-Sampler :

gv\$Active_Session_History
DBA_Hist_Active_Sess_History
DBA_Hist_Cache_Advice
DBA_Hist_Datafile
DBA_Hist_Enqueue_Stat
DBA_Hist_FileStatXS
DBA_Hist_IOStat_Detail
DBA_Hist_IOStat_Filetype
DBA_Hist_Log
DBA_Hist_Memory_Resize_Ops
DBA_Hist_OSStat

DBA_Hist_OSStat_Name
DBA_Hist_Parameter
DBA_Hist_PGAStat
DBA_Hist_Process_Mem_Summary
DBA_Hist_Resource_Limit
DBA_Hist_Seg_Stat
DBA_Hist_Service_Name
DBA_Hist_Snapshot
DBA_Hist_SQL_Bind
DBA_Hist_SQL_Plan
DBA_Hist_SQLStat
DBA_Hist_SQLText

DBA_Hist_StatName
DBA_Hist_Sysmetric_History
DBA_Hist_Sysmetric_Summary
DBA_Hist_System_Event
DBA_Hist_SysStat
DBA_Hist_Tablespace
DBA_Hist_Tempfile
DBA_Hist_TempStatXS
DBA_Hist_TopLevelCall_Name
DBA_Hist_UndoStat
DBA_Hist_WR_Control

Why leave traces for analysis

The escalation of technical problems usually takes place with a certain latency.
Standard query: Process XY ran last night 3 x as long as usual! Why?

Example: The application works with Java JEE application server and session pooling:

- For each transaction, a DB session is again randomly fetched from the pool
- With regard to the OLTP character of the system, transactions are very short
- A process scales in parallel using a variable number of DB sessions

How should the traces in the database history be assigned to the business process here?

Extremely helpful for retrospective analysis is the tagging of DB sessions with business context information about the process at the beginning of a transaction.

Setting the context info is done by executing the PL/SQL function :

DBMS_Application_Info.Set_Module(<Module>, <Action>)

This info is recorded with in various tracks of the DB and allows later the assignment of the DB activities to the triggering process via module and action.

Active Session History

Menu "Analyses/statistics" / „Session Waits“ / „Historic“

- For me the biggest step in the analysis functions of the Oracle DB
- Introduced with Oracle 10g, strongly extended in 11g, still somewhat in 12c
- Historization of the data of all active sessions of the DB from V\$Session ++
- Storage per second in SGA memory, query via view V\$Active_Session_History
- Persistence of every 10th second in the cycle of AWR snapshots in AWR table, query via view DBA_Hist_Active_Sess_History
- Stored in AWR table analogous to other AWR data, default 7 days
- This database can be used to reconstruct the operating conditions of the DB very precisely, even after a long period of time.

Usage requires Enterprise Edition plus licensing of 'Diagnostics Pack' option

Active Session History

Menu "Analyses/statistics" / „Session Waits“ / „Historic“

Session-Statistics from DBA_Hist_Active_Sess_History and gv\$Active_Session_History

Start 2021-06-24 09:02 End 2021-06-24 09:34 Idle-Waits Inst. Grouping Event Filter Show waits

Wait-Event	Total Time waited	Avg. Load	First occurrence	Last occurrence	Avg. Time waited	Samples	Wait-Class	Inst.	Session / Sn.	Start time		End time		User	SQL-ID	SQL Exec-ID	Operat	Module	Act
										2021-06-24 09:02	2021-06-24 09:34	Refresh							
ON CPU	40,517	21.11	2021-06-24	2021-06-24	0.00	40,517	CPU	< 4 >	< 1607 >	< 2 >	< 11916 >	< 36 >	< 1495 >	< 28309 >	< 77 >	< 230 >	< 2 >		
cell single block physical read	15,473	8.06	2021-06-24	2021-06-24	0.68	15,473	User I/O	< 4 >	< 532 >	< 2 >	< 3740 >	< 25 >	< 658 >	< 12062 >	< 24 >	< 126 >	< 1 >		
SQL*Net message from dblink	3,654	1.90	2021-06-24	2021-06-24	125.97	3,654	Network	< 3 >	< 181 >	F	< 2993 >	< 4 >	< 161 >	< 2897 >	< 3 >	< 21 >	< 1 >		
buffer busy waits	1,574	0.83	2021-06-24	2021-06-24	8.65	1,574	Concurrency	< 3 >	< 103 >	F	< 1558 >	< 3 >	< 19 >	< 1573 >	< 12 >	< 13 >	< 9 >		
log file parallel write	683	0.36	2021-06-24	2021-06-24	0.40	683	System I/O	< 4 >	< 5 >	B	SYS								
gc cr grant 2-way	654	0.34	2021-06-24	2021-06-24	0.63	654	Cluster	< 4 >	< 201 >	F	< 244 >	< 5 >	< 131 >	< 639 >	< 10 >	< 33 >	< 3 >		
enq: TX - index contention	619	0.39	2021-06-24	2021-06-24	31.55	619	Concurrency	< 1 >	< 81 >	F	< 619 >	N/A	< 4 >	< 619 >	< 2 >	< 5 >	< 4 >		

Number of waiting sessions condensed by 10 seconds for top-10 grouped by: Event, Filter: time_selection_start="2021-06-24 09:02", time_selection_end="2021-06-24 09:34":

Legend:

- ON CPU
- cell single block physical read
- SQL*Net message from dblink
- buffer busy waits
- log file parallel write
- gc cr grant 2-way
- enq: TX - index contention
- gc cr block busy
- gc cr disk read
- [Others]
- Time

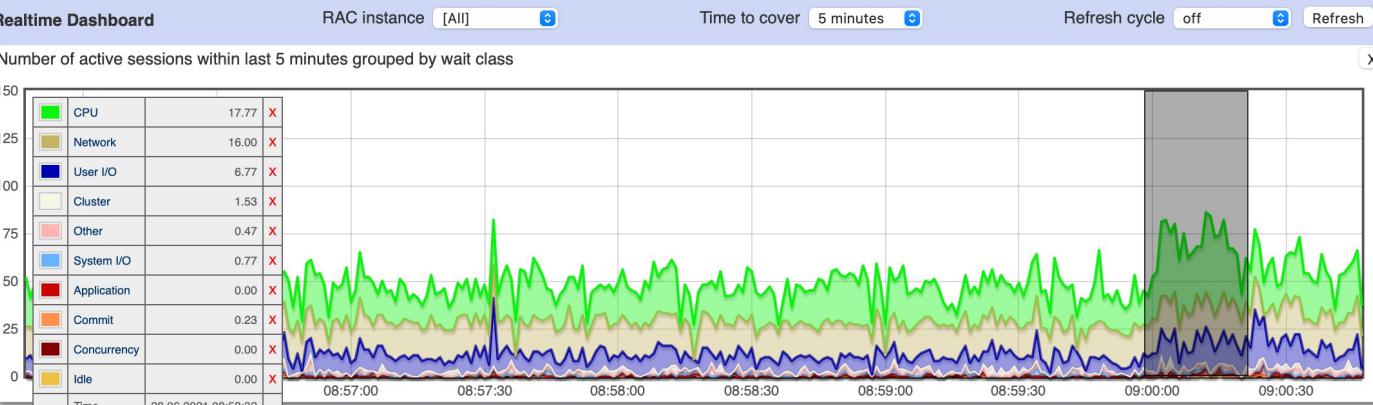
- Select period and entry grouping
- Successive drilldown according to various criteria
- „< xxx>“ shows number of different values of the category column in current selection
- Click on „< xxx>“ groups the values of the current row according to this category in another table
- Drill down to individual ASH sample records

ASH: Real-time dashboard

Menu “DBA general” / „Dashboard“

Realtime Dashboard RAC instance [All] Time to cover 5 minutes Refresh cycle off Refresh

Number of active sessions within last 5 minutes grouped by wait class



Time: 28.06.2021 08:58:32

Top sessions and SQLs between 2021-06-28 08:59:58 and 2021-06-28 09:00:21

SID / SN	Wait time (sec.)	PQ	User	Machine	OS user	SQL	SQL_ID	Wait time (sec.)	Sess.	PQ	SQL text
336,	23		HALDEN	kr1suedprod.int-	halden	b160	b160n08hfbc3c	23	336,		Select * from (with gebinde_id AS (SELECT id from a1ubsw77q0bz
481,	20		SYS	kr1suedprod	root	a1ub	a1ubsw77q0bz	20	481,		WITH LIST AS (/* verhindern, das C@TS-Meldung
623,	20		SYS	kr1suedprod	root	b7qs	b7qs8g3wjvqz	20	623,		SELECT /*+ ORDERED */ ko_lagertyp, ko_raeum_st
1055,	17		AUFRAGSPUFFER_READ_ONLY	ctmproda.ov.otto.de	cmagent	asp8	asp8mtqrma9t	17	1055,		SELECT COUNT(INFO.LOGMSG) AS ANZAHL,
733,	16		HALDEN	kr1suedprod		71yv	71yv24f2v70ht	16	733,		UPDATE rt SET rt_wap_nr = (SELECT wa_wap_nr
875,	13		HALDEN	kr1suedprod.int-	halden	0zf8	b7a3mk9dbu5fs	10	896,		WITH params AS (SELECT /* Report
896,	10		HALDEN	kr1suedprod.int-		b7a3	ggzd4m24uk66z	9	182,		SELECT DISTINCT "A1"."SPRINT_EXPORT_ID"
182,	9		SPRINTSST_KR1	dm13db03		g9zd	0zf88jcmqas1u	7	875,		SELECT /* ZwangsQuittierungService */
1048,	8		HALDEN	kr1suedprod		4zcra	4zzcrfz7xkzd	7	1048,		UPDATE zo SET zo_status_code = 2 WHERE
747,	5		HALDEN	kr1suedprod.int-	halden	a0hx	40vrhsspr80zu	4	< 2 >		select /*+ RULE */ FA_lfd_tag ,FA_bat_nr ,FA_sen_nr

- Real-time monitoring of active sessions by wait class
- Automatic refresh in desired cycle
- Top SQLs and Top sessions for chosen period
- Drilldown by wait class (click in legend), session or SQL

ASH: Retrospective analysis of blocking locks

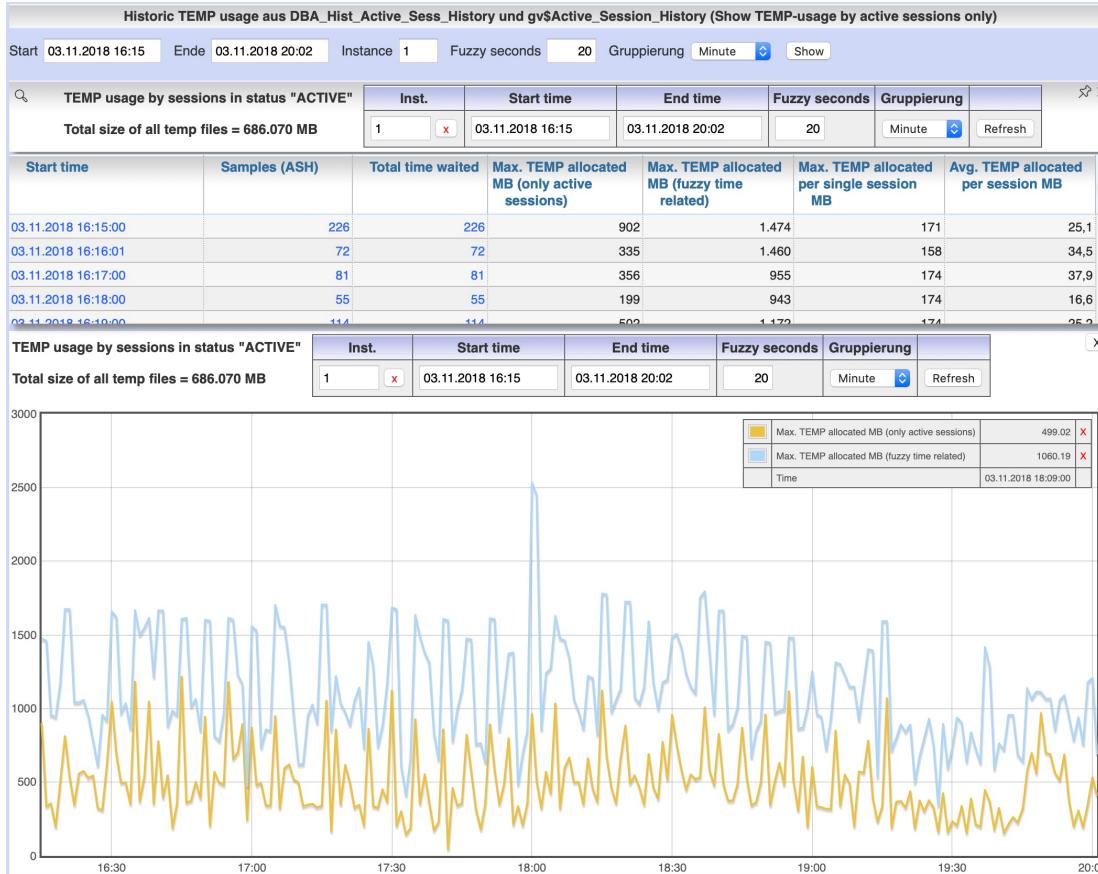
Menu “DBA general“ / „DB-Locks“ / „Blocking locks historic from ASH“

Blocking Locks from DBA_Hist_Active_Sess_History																	
Start 2021-06-24 09:02		End 2021-06-24 09:34		Blocking locks session dependency tree				Instances		Blocking locks event dependency							
Blocking locks between 2021-06-24 09:02 and 2021-06-24 09:34 hierarchical grouped beginning with root-blockers																	
First occ.	Last occ.	Dead lock	Samples direct	B. I.	B.SID	B.User	B.SQL-ID	B.Event	Blocking Module / Action / Program	Direct Blocked	Total Blocked	Max. depth	Total Wait (sec.)	Waiting (sec.)	Blocking Object	W.I.	W.User
2021-06-24 09:01:59	2021-06-24 09:34:01			3,299	GLOBAL				INACTIVE			< 481 >	< 482 >	2	6.4	3,321 < 20 > < 303 >	< 4 > < 22 >
2021-06-24 09:02:06	2021-06-24 09:34:00			491	2	6242	SYS		< 3 >	oracle@dm13db02 (LGWR)		< 105 >	< 105 >	1	0.1	491 < 5 > < 8 >	2 < 3 >
2021-06-24 09:02:08	2021-06-24 09:33:43			219	1	6242	SYS		< 3 >	oracle@dm13db01 (LGWR)		< 59 >	< 60 >	2	0.0	220 < 6 > < 10 >	1 < 2 >
2021-06-24 09:09:55	2021-06-24 09:11:08	Y		50	1	1457	NOA	634vh689g5 < 2 >	User name of blocking session	ID_Application = 1582 JDBC Thin Client		< 36 >	< 70 >	5	0.7	162 INDEX SYSP.IND_FHDATAREC	1 NOA
2021-06-24 09:03:50	2021-06-24 09:16:46			47	1	1136	NOA	< 3 >	< 3 >	< 3 > process OFMessages JDBC Thin Client		< 32 >	< 68 >	6	1.2	151 INDEX < 2 > < 2 >	1 < 2 >
2021-06-24 09:09:58	2021-06-24 09:16:46	Y		39	1	812	NOA	634vh689g5 < 2 >	< 2 > JDBC Thin Client		< 31 >	< 63 >	5	0.6	141 INDEX SYSP.IND_FHDATAREC	1 NOA	

- Shows one line per blocking cascade triggering session (root blocker)
- Sorted by waiting time of all directly or indirectly blocked sessions
- Drilldown in:
 - Blocked sessions
 - ASH of blocking session
 - ASH of blocked sessions
 - Blocking objekt down to the primary key value of the record

ASH: Retrospective analysis of TEMP users

Menu "Schema / Storage" / „TEMP-usage“ / „Historic from ASH“



Detection of the users of TEMP tablespace including the cause of a possible "unable to extend temp segment"

- Display of TEMP usage over time in diagram
- Determine the relevant point in time
- Change to ASH analysis to determine specific sessions by TEMP consumption at that time

See also: [Blog post on the topic](#)

Agenda

- Presentation of the used tool "Panorama"
- Top/down analysis of session activities
- Executed SQL statements with detail analysis
- Segment statistics: Recording of characteristic values for objects (tables, indexes ...)
- Capacity and utilization of CPU, memory and I/O system, configuration of the instance
- Dragnet investigation: Systematic scan of the system for performance antipatterns

SQL History

Menü „SGA/PGA Details“ / „SQL-Area“ / ..

SQL of current SGA from GV\$SQLArea, grouped by SQL-ID

I	SQL-ID	SQL-Text	C	V	P	Last active	User	Parse	Execs	Elapsed	Ela./Ex.	CPU	Disk Reads	Disk/Ex.	Buffer Gets	Buffer/Ex.	Rows proc
1	6wq9h22z83d0y	SELECT ID, ID_DISCRIMINATOR,	1	2	1	2018-11- EKS	EKS		24,458,911	155,832	0.0064	134,901	437,947	0	117,746,970	5	10,51
1	9tpcpq6wqc6s7	SELECT S.SID FROM V\$SESSION S	1	3	1	2018-11- DBAITEWKD	DBAITEWKD		63,563,724	150,815	0.0024	131,043	71	0	2,215	0	63,56
1	1tyjz7ucuuq5	BEGIN :1 :=	1	1	1	2018-11- EKS	EKS		246,232	123,718	0.5024	107,186	205,302	1	1,117,368,294	4,538	241

Statement details of current SGA from GV\$SQL: Instance = 1, SQL-ID = '6wq9h22z83d0y', Child-No. = 0

```
/* single line SQL-text formatted by Panorama */
SELECT ID, ID_DISCRIMINATOR, ID_EDITORIALNODE, TSUPDATE, ID_LOGICALPRODUCT, OBJECTID
FROM BUYING.EDITORIALLINK
WHERE ((OBJECTID = :1) AND (ID_DISCRIMINATOR = :2 ))
```

Parsing schema name	EKS	
Plan-Hash-Value	2489798738	
Child-address	0000003ED664198	
Optimizer Env Hash-Value	2384849091	
Parsing module	JDBC Thin Client	
Parsing action		
Object status	VALID	
PL/SQL program / line	..:0	
First Load Time	2018-10-14 14:00:20	
Last Load Time	2018-11-09 05:00:25	
Last Active Time	2018-11-09 11:57:05	
Buffer cache hit ratio	99.63	
Metric-Name	Total	Per Exec.
Executions	24,459,973	-
Parse calls	2,543,252	0.10
Fetches	24,453,828	1.00
Sorts	0	0.00
Rows processed	10,515,132	0.43
Buffer gets	118,494,428	4.84
Disk reads	437,957	0.02
Elapsed Time (sec), incl. parallel Query Slaves	155,837.21	0.0064
CPU-Time (sec)	134,906.57	0.0055
Application Wait Time (sec)	0.00	0.0000
Concurrency Wait Time (sec)	16.66	0.0000
Cluster Wait Time (sec)	0.00	0.0000
User-IO Wait Time (sec)	1,031.38	0.0000
PL/SQL Exec Time (sec)	0.00	0.0000

Execution-Plan (1) Bind variables (2) Objects (2) Full history Cursor Sharing (1 versions) Active Session History Open Cursor (81) DBMS_XPLAN SQL-Monitor (0) SQL-Patch

Evaluation of the SQLs currently in SGA as well as the history from AWR

Listing of SQLs sorted by various criteria

Details per SQL incl.

- Execution plan
- Bind variables
- Child cursors
- Full history of all AWR snapshots of SQL
- Reasons for multiple cursors
- SQL monitor recordings
- Create baseline / SQL patch

Explain Plan of Child=0 parsed at 2018-11-09 05:00:25, Optimizer-Mode=ALL_ROWS, Executions=24,462,287, first ASH-Sample in SGA from 2018-11-09 07:54:19

ID	R.	Object-name	Rows	MB	Cost	Card.	Parallel	Access	Filter	Temp est.	Temp max.	DB time	CPU	Waits	I/O	IC	Dist	PGA max.	Proj.	Starts	Rows	Cost
0	3									3	0.9	0.9	1.0	1.0					23			
1	2	BUYING.EDITORIALLINK	3,959,392	183	1,308	1				3	0.2	0.1	36.4	0.2	0.2				23 "ID"			
2	1	BUYING.IX_EDTLINK_LOGPK	3,959,392	155	1,307	1		"ID DISCRIMINATOR" ("OBJECTID" =:1 AND		3	98.9	99.0	63.6	98.9	98.9				25 "EDITORIALLIN			

SQL statement details: Execution plan



SQL statement details : AWR history

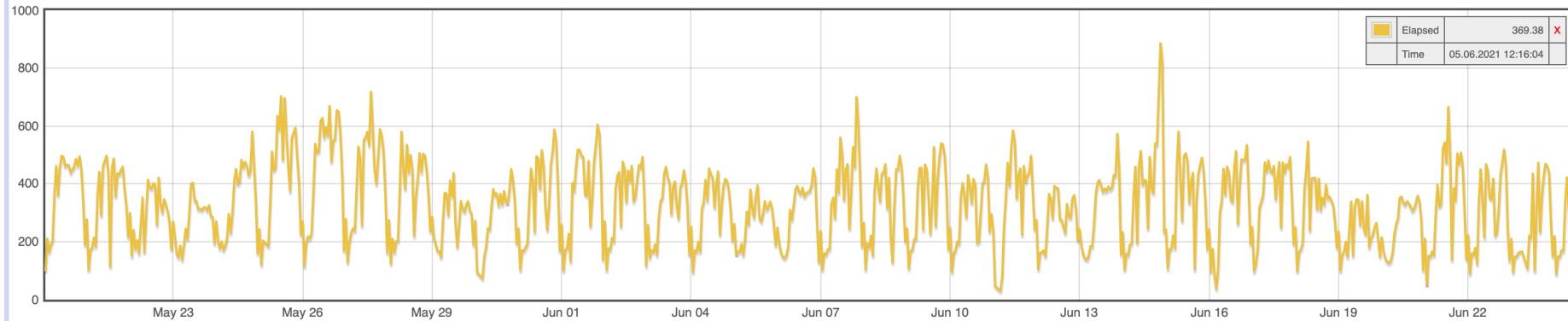


Historic snapshots from DBA_Hist_SQLStat for SQL-ID = 'apq66m3gj4uwg' Instance = 2

Start	First occurrence	Last occurrence	Plans	Plan hash value	Env.	Opt. env hash value	Execs	Elapsed	Ela. / Ex.	Ela. / Row	CPU	App.	Con.	Clu.	I/O	PL/SQL	Disk Reads	Disk / Ex.	ms / Disk Read	Hit Ratio	Buffer Gets	Buffer / Ex.	Buffer / Row	Rows proc.	Rows / Ex.	Parses
2021-06-24 10:00:00	2021-06-24 10:00:07	2021-06-24 10:15:03	1	2238303688	1	1683692210	421,968	100	0.0002	0	50	0	1	0	23	24	60,196	0.143	0.37	97.14	2,104,214	4.99	5.06	416,068	0.99	0
2021-06-24 09:00:00	2021-06-24 09:45:31	2021-06-24 10:00:07	1	2238303688	1	1683692210	410,333	93	0.0002	0	49	0	1	0	18	23	58,468	0.142	0.31	97.14	2,046,823	4.99	5.05	405,085	0.99	1
2021-06-24 08:00:00	2021-06-24 08:00:19	2021-06-24 09:00:16	1	2238303688	1	1683692210	1,279,603	315	0.0002	0	166	0	2	0	70	75	210,311	0.164	0.33	96.71	6,386,473	4.99	5.04	1,267,592	0.99	15
2021-06-24 07:00:00	2021-06-24 07:00:10	2021-06-24 08:00:19	1	2238303688	1	1683692210	1,775,193	422	0.0002	0	207	0	3	0	103	97	279,368	0.157	0.37	96.85	8,868,559	5.00	5.02	1,767,650	1.00	1

Historic snapshots from DBA_Hist_SQLStat for SQL-ID = 'apq66m3gj4uwg' Instance = 2

Start	Schema	NOA	End	Grouping	Hour	Refresh
2021-05-20 01:15			2021-06-24 10:15			



SQL statement details : SQL-Monitor

Enterprise Manager Database Active Report

Real-time SQL Monitoring: Ou8czymn2gpar Status ✓

Time Zone: Browser (GMT+02:00) ▾

Overview

General

SQL Text: `SELECT PCSELEKT.ART.BEST_WG_FT_KZ FT, PCSELEKT...`
Execution Started: 06.05.2021 16:58:25 GMT+02:00
Last Refresh Time: 06.05.2021 17:01:07 GMT+02:00
Execution Id: 16777216
User Name: THEINS@EKR30PS
Fetch Calls: 1

Time & Wait

Metric	Value
Duration	2,7 m
Database Time	2,97 m
PL/SQL & Java	0 s
Activity	100 %

I/O

Metric	Value
Buffer Gets	1.009K
I/O Requests	334K
I/O Bytes	2,6 GB

Details

Plan Statistics Plan Notes

Plan Hash Value: 1020049330 Plan Size: 20 Lines

Operation	Object	Informati	Line ID	Timeline	Execut	Est. Rows	Rows	Mem (Max)	Temp (Max)	I/O Requests	Activity
SELECT STATEMENT			0		1						
SORT ORDER BY			1		1	18K					
HASH GROUP BY			2		1	18K					
FILTER			3		1						
HASH JOIN OUTER			4		1	18K	1.552	1,8 MB			5,03 %
FILTER			5		1	1.552					
HASH JOIN OUTER			6		1	18K	1.552	1,8 MB			3,77 %
NESTED LOOPS			7		1	18K	1.552				
NESTED LOOPS			8		1	55K	1.552				
HASH JOIN			9		1	55K	1.552	2,6 MB			0,63 %
TABLE ACCESS BY INDEX ROWID BATCHED	ART		10		1	35K	1.008		3		
INDEX RANGE SCAN	IND_ART_6		11		1	20K	2.000		12		
INLIST ITERATOR			12		1	29M					

SQL statement details : Plan stability

Menu "SGA/PGA Details" / „SQL plan management“

SQL plan management directives										
SQL profiles from DBA_SQL_Profiles										
Name	Category	Created	Last modified	Description	Type	Status	Force matching	Usages in SGA	Usages in history	SQL
coe_7g7nxtsc978z3_1180579734	DEFAULT	28.02.2020 16:25:54	28.02.2020 16:25:54	coe 7g7nxtsc978z3 1180579734 1684505828464809677, 1684505828464809677,	MANUAL	ENABLED	NO	1	1	SELECT ITEMcmp_id, ITEMcmptype
SYS_SQLPROF_0171e44aea53	DEFAULT	05.05.2020 12:04:07	05.05.2020 12:04:07		MANUAL	ENABLED	NO			INSERT INTO temp_ottoint WITH
SYS_SQLPROF_0171bb184c610	DEFAULT	27.04.2020 12:04:24	27.04.2020 12:04:24		MANUAL	ENABLED	NO			SELECT ABEST_WG_FT_KZ FT,
SYS_SQLPROF_014932983b35	DEFAULT	18.02.2020 15:02:05	18.02.2020 15:02:05		MANUAL	ENABLED	NO			SELECT TL.ID ,TL.LOGTYPE ,GETI

Existence per SQL and
as overview :

- SQL-Profiles
- SQL Plan Baselines
- Stored Outlines
- SQL-Translations
- SQL-Patches

For executed SQLs generation of PL/SQL snippets for creation of directives :

- SQL plan baselines: Pinning based on already existing plans via plan hash value
- SQL patches: Injecting optimizer hints into unmodified SQL statements
- SQL translations: Complete replacement of SQL text for statements unchanged in application

Agenda

- Presentation of the used tool "Panorama"
- Top/down analysis of session activities
- Executed SQL statements with detail analysis
- Segment statistics: Recording of characteristic values for objects (tables, indexes ...)
- Capacity and utilization of CPU, memory and I/O system, configuration of the instance
- Dragnet investigation: Systematic scan of the system for performance antipatterns

Structure info of tables, indexes, packages etc.

Menu: „Schema/Storage“ / „Describe object“

Describe database object

Object-Owner **cust** Object-Name **Customer** Object-Type [Alle]

Description of TABLE CUST.CUSTOMER

Company dependent customer
Komprimierung reduziert Footprint auf 60%

Columns of TABLE CUST.CUSTOMER

Col.Name	Type	Prec.	Sc.	N.	Def.	Distinct	Nulls	Avg. Len.	Density	Buckets	Histogram	Comments	LOB segment	EQ	EQJ	NEQ
ID	NUMBER	9	0	N		68.305.085	0	7	0,0000	1	NONE	Primary Key		24.729	84.737	0
ID_COMPANY	NUMBER	4	0	N		59	0	3	0,0000	59	FREQUENCY	Company to		48.698	37.917	0
ID_PERSON	NUMBER	9	0	N		68.305.085	0	7	0,0000	1	NONE	Person that		9.015	26.073	0
CREATIONDATE	DATE			N		43.819.008	0	8	0,0000	254	HYBRID	Date of		5.626	192	78
ID_LOCALE	NUMBER	4	0	N		13	0	3	0,0000	13	FREQUENCY	Locale		7.318	4.971	0
CUSTNO	VARCHAR2	10	Chars	N		39.424.000	0	10	0,0000	254	HYBRID	Company		12.343	1.714	0
CANCELLATIONDATE	DATE			Y		96.240	64.013.335	2	0,0000	254	HYBRID	Date when		7.322	0	0
ORDERCOUNTER	NUMBER	6	0	Y		2.533	12.500.973	3	0,0000	254	TOP-	Comments		119	0	0
LASTONLINEUPD	DATE			Y		46.915.584	7.174.628	8	0,0000	1	NONE	Letzte				
ID_SHOPDOMAIN	NUMBER	4	0	Y		88	44.031.316	3	0,0114	1	NONE	ShopDomain,				

Attributes of TABLE CUST.CUSTOMER

TS	Pct Free	Init. Trans	Initial extent (KB)	Rows	Size (MB) Table	Size (MB) Total	Extents	Blocks	Empty	Avg. Space	Chained	RowLen	Dg.	Cache	Part.	Sub-Part.	Created	Last Update
DATA02	10	8		68.305.085	4.659,75	18.598,94	598	596.448	0	0	0	56	1	N		2	08.03.2015 00:42:45	05.08.2015 08:08:08

6 Indexes 1 Primary Key 2 Check Constraints 4 References from 85 References to 12 Triggers 129 Dependencies 49 Grants DBMS_METADATA DB-Cache

Sessions accessing SQLs Active Session History Segment statistics Size evolution

Detailed info on DB objects with drilldown into structures, accesses, dependencies, etc.

Storage usage overview

Menu: „Schema/Storage“ / „Disk storage summary“

Storage-objects for database NOADB_PROCESS 2021-06-24 10:53

Sums total				
Typ	MB Total	MB Used	MB Free	% used
PERMANENT	11,168,763	8,596,727	2,572,036	77.0
TEMPORARY	956,121	873,918	82,203	91.4
UNDO	917,504	789,474	128,030	86.0
Redo-Logs outside FRA	196,608	196,608	0	100.0
Fast Recovery Area	9,216,000	5,122,253	4,093,747	55.6
TOTAL	22,454,996	15,578,980	6,876,016	69.4

Net sums in TS by segment types				
Segment-Type	MBytes			
TABLE	4,637,933			
INDEX	3,677,001			
TYPE2 UNDO	789,465			
LOBSEGMENT	0.00	50.00	20.81	
SYSTEM STATISTICS	= 3,590.8 Gigabytes			
CLUSTER	= 3.5 Terabytes			
	= 39.18 % of column sum 9386024			

Schema-Usage							
Schema	Table	Index	Index IOT-PKey	TYPE2 UNDO	SYSTEM STATISTICS	ROLLBACK	Total MB
AUFTRAG	1,984,192	2,078,643	26,900				4,089,736
CUST	297,175	634,812	605,292				1,537,279
SYS	66,152	35,559	24	789,465	8,447	0	899,646
LOGISTIK	204,000	242,100	0				496,100

Tablespace-usage per schema														
Schema	Tablespace			Used ~ (MB)	Quota (MB)	Charged (MB)								
AUFTRAG	DATASP01			1,979,282										
CUST	DATA02			558,223										
CUST	INDEX01			329,599										
CUST	DATA01			317,831										
CUST	INDEX02			278,581										

Tablespace-usage total from DBA_Tablespace																					
Tablespace	Contents	Block size	MB ~ Total	MB Used	MB Free	% used	Auto ext.	MB max.	% used max.	Files	Status	Logging	Force log.	Extent mgmt.	Allocation Type	Pl. in	Segment Space Mgmt.	Def. table compress	Big file	Encry	Def. in memory
DATASP01	PERMANENT	8,192	2,867,200	2,521,547	345,654	87.9	NO			1	ONLINE	LOGGING	NO	LOCAL	SYSTEM	NO	AUTO	DISABLED	YES	NO	DISAB
INDEXSP01	PERMANENT	8,192	2,867,200	2,497,308	369,892	87.1	NO			1	ONLINE	LOGGING	NO	LOCAL	SYSTEM	NO	AUTO	DISABLED	YES	NO	DISAB
DATA02	PERMANENT	8,192	1,730,560	1,157,123	573,437	66.9	NO			1	ONLINE	LOGGING	NO	LOCAL	SYSTEM	NO	AUTO	DISABLED	YES	NO	DISAB
DATA01	PERMANENT	8,192	1,228,800	962,742	266,058	78.3	NO			1	ONLINE	LOGGING	NO	LOCAL	SYSTEM	NO	AUTO	DISABLED	YES	NO	DISAB
INDEXX01	PERMANENT	8,192	921,600	666,775	254,825	72.3	NO			1	ONLINE	LOGGING	NO	LOCAL	SYSTEM	NO	AUTO	DISABLED	YES	NO	DISAB

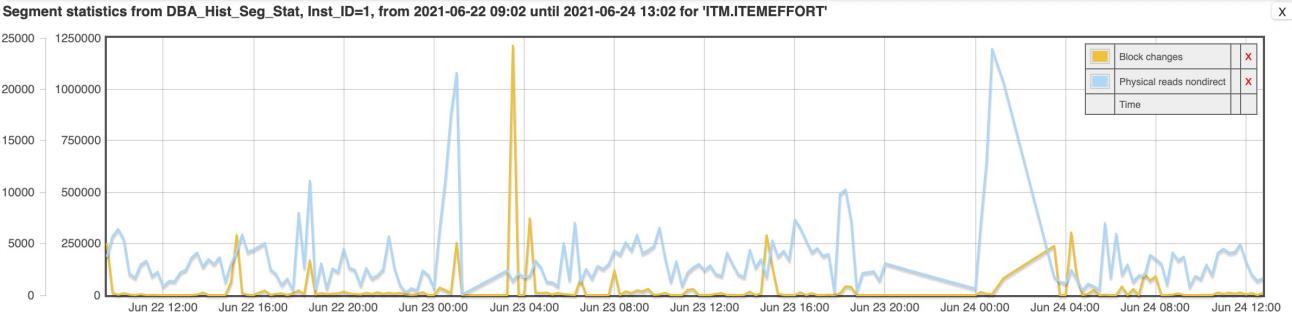
- Storage usage by types, schemas, tablespaces
- Drill down to objects and partitions
- Calculation of the space to be released by reorganization
- Representation of the real available storage under consideration of fragmentation

Segment statistiks

Menu „Analyses / statistics“ / „Segment statistics“

Historic segment statistics from DBA_Hist_Seg_Stat																						
Start	2021-06-22 09:02	End	2021-06-24 13:02	Inst.		Part.		Objekt-Name		Show												
I	Object	SQLs	Num Rows	Object size (MB)	Time waited	Avg. Time waited	AWR snaps.	Logical reads	Buffer busy waits	Block changes	Physical reads nondirect	Physical reads direct	Physical writes nondirect	Physical writes direct	ITL	Row lock waits	GC buffer busy waits					
1	ITM . ITEMEFFORT	5,670	47,546,070	7,202.44	105,830	69.22	174	665,658,976	66	103,136	31,143,264	77,469,497	84,582	0	0	0	21					
1	AUFTRAG . AU_ABSTIMMZAHL	76	10,355,730	868.00	90,570	0.55	12	56,060,672	0	0	755,641	0	0	0	0	0	0					
2	AUFTRAG . AU_FORDERUNG_OP	333	2,629,259,559	469,960.19	64,110	16.78	188	23,628,943,488	2,310	11,699,376	177,878,302	0	1,496,337	0	0	0	8					
1	MOMAIN . CUSTORDER	972	15,014,599	3,107.94	61,760	5.09	208	10,963,614,400	19,955	33,929,533	11,226,208	7,534,762,6C	414,706	0	0	0	41					

Segment statistics from DBA_Hist_Seg_Stat, Inst_ID=1, from 2021-06-22 09:02 until 2021-06-24 13:02 for 'ITM.ITEMEFFORT'																
Time	Logical reads	Buffer busy waits	Block changes	Physical reads nondirect	Physical reads direct	Physical writes nondirect	Physical writes direct	ITL	Row lock waits	Global cache buffer busy waits	GC CR blocks received	GC CU blocks received	Space used (MB)	Space allocated (MB)	Full table scans	
2021-06-22 09:30:08	1,262,816	1	5,040	185,821	0	3,577	0	0	0	2,367	214,573	111,598	0.00	0.00	0	
2021-06-22 09:45:18	2,979,728	0	224	286,916	662,030	109	0	0	0	723	422,481	238,813	0.00	0.00	106	
2021-06-22 10:00:09	3,125,232	0	32	321,033	662,158	28	0	0	0	679	484,729	263,542	0.00	0.00	106	



Characteristic values per table/index for period

Sortable according to individual characteristic values

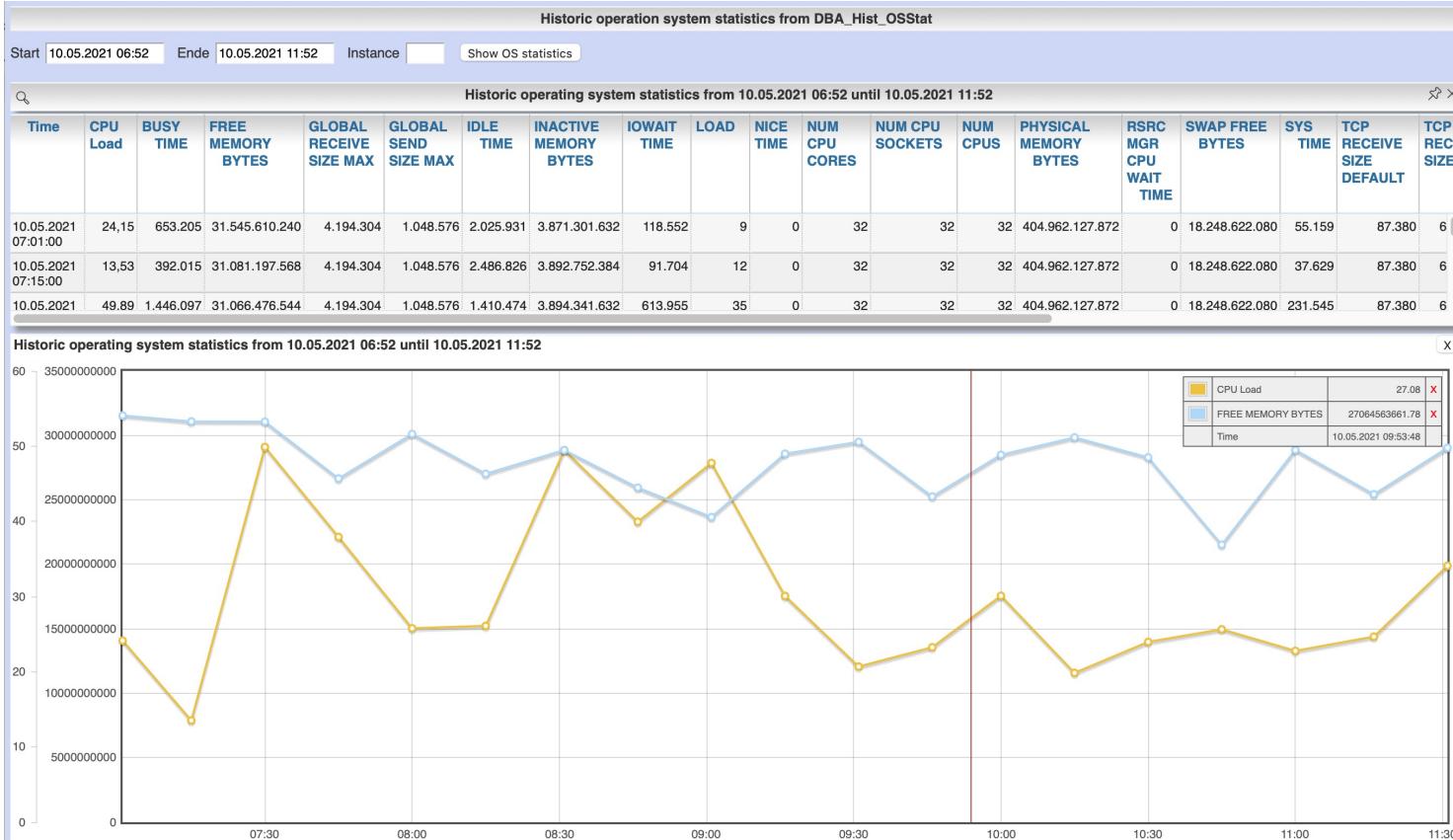
Evolution over time within the period

Agenda

- Presentation of the used tool "Panorama"
- Top/down analysis of session activities
- Executed SQL statements with detail analysis
- Segment statistics: Recording of characteristic values for objects (tables, indexes ...)
- Capacity and utilization of CPU, memory and I/O system, configuration of the instance
- Dragnet investigation: Systematic scan of the system for performance antipatterns

OS statistics for CPU and memory

Menu: „Analyses/statistics“ / „OS statistics“



- Number of CPUs/Cores
- Total CPU load of the machine
- Total physical memory
- Available physicle memory
- Available swap space
- History in resolution of the AWR cycle

I/O-statistics

Menu: „I/O-Analysis“ / „I/O history by files“

I/O History from DBA_Hist_FileStatxs (Data and Temp files, without Redo-Logs)

Start 10.05.2021 06:52 Ende 10.05.2021 12:00 Instance Gruppierung FileType Show I/O history

Verdichtung nach FileType. Filter: Start time End time
10.05.2021 06:52 Refresh 10.05.2021 12:00 Refresh

FileType	Erstes Auftreten	Letztes Auftreten	Dauer	DB	Inst.	Tablespace	Filename	Samples	Phys. reads	ms / phys. read	Blocks / phys. read	Phys. writes	ms / phys. write	Blocks / phys. write	Single block reads	ms / single block read	Blocks / multi block read	Total I/O per second	Read I/O per second	Write I/O per second
DATA	10.05.2021 07:00:01	10.05.2021 12:00:12	18.011	COR01PX	< 4 >	< 20 >	< 24 >	20	-3.696.468.899	-0,05	-0,21	56.528.705	1.300,52	1,85	591.784.930	0,26	0,0	-202.095	-205.234	3.139
TEMP	10.05.2021 07:00:01	10.05.2021 12:00:12	18.011	COR01PX	< 4 >	< 2 >	< 23 >	20	5.777.391	9,04	15,55	12.451.900	43.784,75	15,55	1.372.537	2,85	20,1	1.012	321	691

Avg. total I/O-operations per second within sample time

Sample-Records. Filter: FileType Start time End time
DATA 10.05.2021 06:52 Refresh 10.05.2021 12:00 Refresh

Sample-Time	Phys. reads	ms / phys. read	Blocks / phys. read	Phys. writes	ms / phys. write	Blocks / phys. write	Single block reads	ms / single block read	Blocks / multi block read	Total I/O per second	Read I/O per second	Write I/O per second
10.05.2021 07:00:00	42.229.513	0,27	1,43	3.905.480	162,34	2,68	41.783.676	0,26	41,3	50.642	46.355	4.287
10.05.2021 07:15:00	46.722.063	0,28	1,32	6.188.583	179,33	2,36	46.289.290	0,27	35,6	57.449	50.730	6.720
10.05.2021 07:30:00	32.909.889	0,27	1,51	6.187.230	902,84	1,97	32.407.982	0,25	34,7	42.870	36.085	6.784

Sample-Records. Filter: FileType Start time End time
DATA 10.05.2021 06:52 Refresh 10.05.2021 12:00 Refresh

The chart shows three data series over a four-hour period. The 'Single block reads' series starts at approximately 3.5 million, peaks at about 4.5 million around 07:45, and then fluctuates between 2.0 and 3.0 million. The 'ms / single block read' series starts at 2.5 million, peaks at 3.5 million around 08:00, and then generally declines to around 2.5 million. The 'Time' series, represented by crosses, starts at 2.5 million, dips to 2.0 million around 07:45, and then rises to 2.8 million by 11:30.

- Three sources for I/O history from AWR
- Recording of various characteristic values
- different grouping by file types up to file names

- The third source DBA_Hist_Filestatxs is no longer filled by AWR as of 18c

Memory configuration

Menu: „SGA/PGA details“ / „SGA memory“ / „SGA components“

Show memory consumption in SGA

Inst. 1 Show memory components

Filter on specific RAC instance (Optional)

Memory-related init-parameter		Value	Display-Value	Update comment	Default	Container ID
1	memory_max_target	Max size for Memory Target	0	0	TRUE	0
1	memory_target	Target size of Oracle SGA and PGA memory	0	0	TRUE	0
1	pga_aggregate_target	Target size for the aggregate PGA memory consumed by the instance	51539607552	48G	FALSE	0

Summary by pools from gv\$SGAStat for instance = 1

I	Pool	Bytes	MBytes	GBytes	Parameter	Resize ops.
1	buffer cache	60,397,977,600	57,600.00	56.25	db_block_buffers = 0, db_cache_size = 42,949,672,960	400
1	shared pool	9,126,805,504	8,704.00	8.50	shared_pool_size = 5,368,709,120	24
1	numa pool	6,262,012,792	5,971.92	5.83		

Details from gv\$SGAStat for instance = 1

I	Pool	Name	Bytes	MBytes	GBytes
1	buffer cache	60,397,977,600	57,600.00	56.25	
1	numa pool	gcs resources	2,219,791,720	2,116.96	2.07
1	shared pool	free memory	2,039,768,760	1,945.28	1.90
1	shared pool	SQCA	1,497,692,822	1,418.76	1.39

Summary from gv\$DB_Object_Cache (objects that are cached in the library cache) for instance = 1

I	Type	Namespace	DB-link	Kept	Sharable Memory (MB)	Count	Count distinct	Loads	Locks	Pins	Invalidations
1	TABLE	TABLE/PROCEDURE		NO	26	4,612	4,612	1,865,470	0	0	0
1	PTNC	PCINDEX		NO	21	416	17	416	0	0	0
1	XDB SCHEMA DATA	XML SCHEMA		YES	10	2	2	2	0	0	0
1	INDEX	INDEX		NO	8	318	318	226,421	0	0	0
1	MULTI-VERSIONED OBJECT	MULTI-VERSION OBJECT FOR TABLE		YES	8	336	207	363	0	0	0

This view shows :

- Dynamic allocation of physical memory to components
- Control of the distribution through manual specifications of minimum sizes

Optimization target is usually use of SGA memory for DB cache or inMemory store

Usage of DB cache by objekts

Menu: „SGA/PGA details“ / „DB-Cache“ / „DB-Cache usage current“

DB-cache usage

Inst. 1 Part. Show cache content

DB-cache usage: Instance=1, 2021-06-25 09:31:52

Status	Size (MB)	Blocks	%
read	0.094	12	0.00
free	17.039	2,181	0.03
scur	41,424.984	5,302,398	79.86
pi	109.703	14,042	0.21
xcur	6,834.234	874,782	13.17
cr	3,488.398	446,515	6.72

Summary

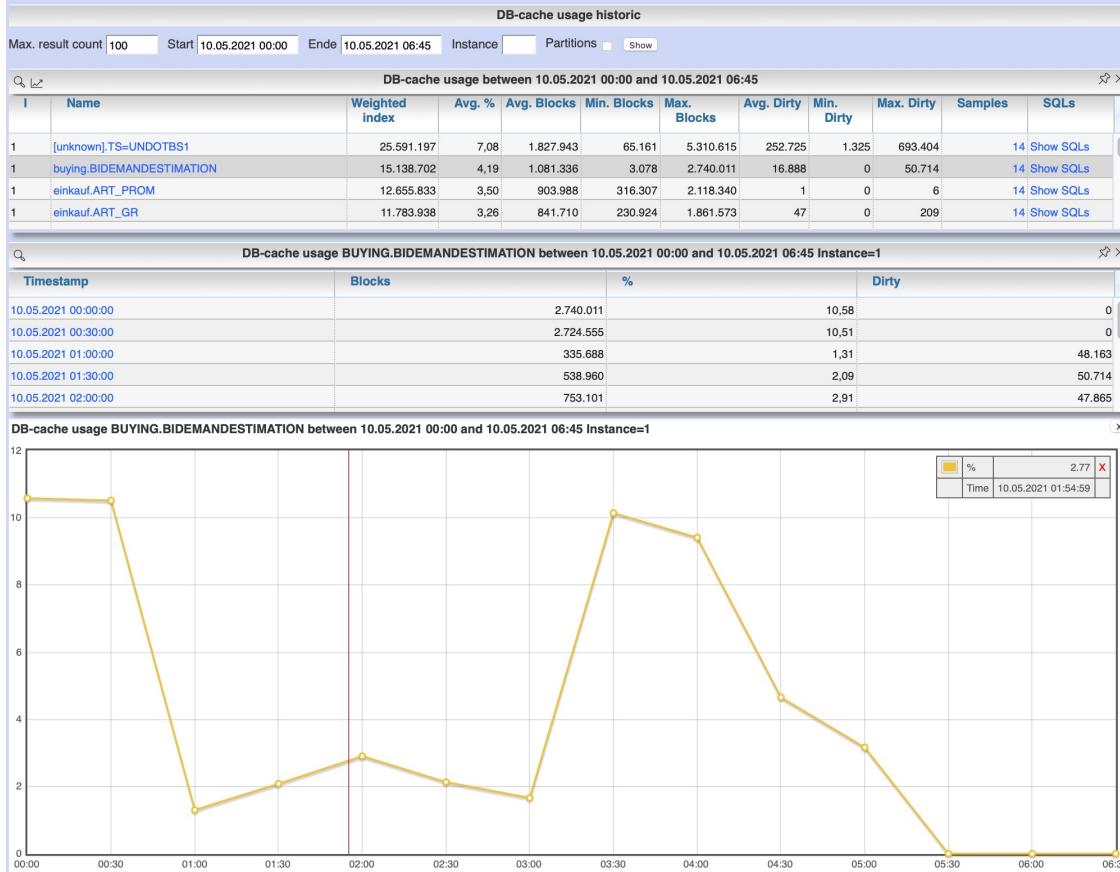
Name	Type	Tablename	SQLs	Size (MB)	Blocks	%	Dirty	cr	pi	read	scur	xcur	F.Read	F.W
WHR . IX_WHACCOUNT_NUNIQ	INDEX	WHR.WAREHOUSEACCOUNT	71	3,631.305	464,807	6.94	684	9,663	17	0	450,437	4,690	0	0
MOMAIN . CUSTORDER	TABLE PARTITION		164	2,492.648	319,059	4.77	2,966	15,120	0	0	286,196	17,743	0	0
CUST . IX_CSTBLCONST_CRTSTATID	INDEX PARTITION	CUST.CUSTBLOCKCONSEQU	0	2,327.625	297,936	4.45	6	4	2	0	296,192	1,738	0	0
WHR . WAREHOUSEACCOUNT	TABLE		74	2,181.133	279,185	4.17	5,867	5,828	40	0	266,619	6,698	0	0
AUFTRAG . IX_AU_GP_KONT_GPKAG	INDEX	AUFTRAG.AU_GP_KONTO	22	2,022.016	258,818	3.87	0	6	0	0	258,812	0	0	0
ITM . ITEMEFFORT	TABLE PARTITION		140	1,873.781	239,844	3.58	2	119,263	0	1	120,573	7	0	0
JOURNAL . OFMESSAGE	TABLE SUBPARTITION		18,720	1,687.500	216,000	3.23	3,839	13,273	604	0	190,688	11,435	0	0
SYS . FGA_LOG\$	TABLE		12	1,652.164	211,477	3.16	6	285	0	0	205,453	5,739	0	0
ITM . EMREPLICATIONLOG	TABLE PARTITION		72	1,114.563	142,664	2.13	667	2,830	367	0	115,807	23,660	0	0

Use of DB cache by tables and indexes.

- Ensure that the DB cache is really used for business-relevant objects
- Detect suboptimal SQLs that bring objects with low business relevance to the top of the LRU list

History of DB cache by objects

Menu: „SGA/PGA details“ / „DB-Cache“ / „DB-Cache usage historic“



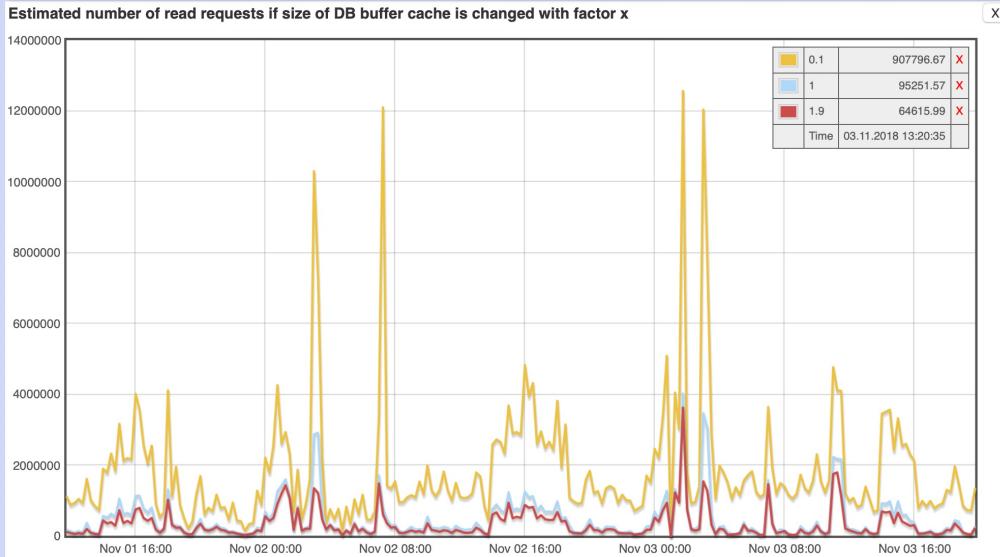
Use of DB cache by tables and indexes in a period of the past.

- Data source is the Panorama-Sampler, not AWR
- Time course of DB cache usage per object

Forecast when changing the DB cache size

Menü: „SGA/PGA Details“ / „DB-cache“ / „DB-cache advice“

Estimated number of read requests if size of DB buffer cache is changed with factor x													
	0.7	0.8	0.9	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9
54	221.843	202.931	188.383	168.107	155.953	148.503	138.023	125.239	122.366	119.871	113.121	111.138	109.777
12	177.549	133.739	128.205	124.862	114.165	102.925	99.728	96.950	93.998	81.801	76.721	75.395	73.239
36	136.021	127.653	106.612	86.631	83.768	81.838	79.392	72.946	69.767	59.895	51.927	50.354	48.116
00	180.118	168.879	149.520	129.184	125.957	122.218	103.441	102.197	99.858	79.844	75.690	73.367	71.256
10	148.905	115.622	99.992	93.183	89.115	79.991	67.211	65.203	62.889	61.286	59.419	51.368	50.544



Predicting the change in the number of I/O read requests when increasing/decreasing the DB cache at specific times.

- 1 = values for current Cache-Size
- 0.1 = Prediction for reducing cache to 10%
- 1.8 = Prediction for increasing cache by 80%

Configuration of the redo logs: Actual state

Menü: „DBA general“ / „Redologs“ / „Current“

Redo-Logfiles from GV\$LOG at 19.10.2018 08:13:05							
Instance	Group	Size (MB)	Status	Start timestamp	Members	Archived	Log switch interval
1	1	200,00	CURRENT	18.10.2018 10:36:38		1 NO	
1	3	200,00	INACTIVE	15.10.2018 07:47:04		1 NO	269.374
1	2	200,00	INACTIVE	14.10.2018 23:15:10		1 NO	30.714

This picture shows the default configuration after installing an Oracle DB.

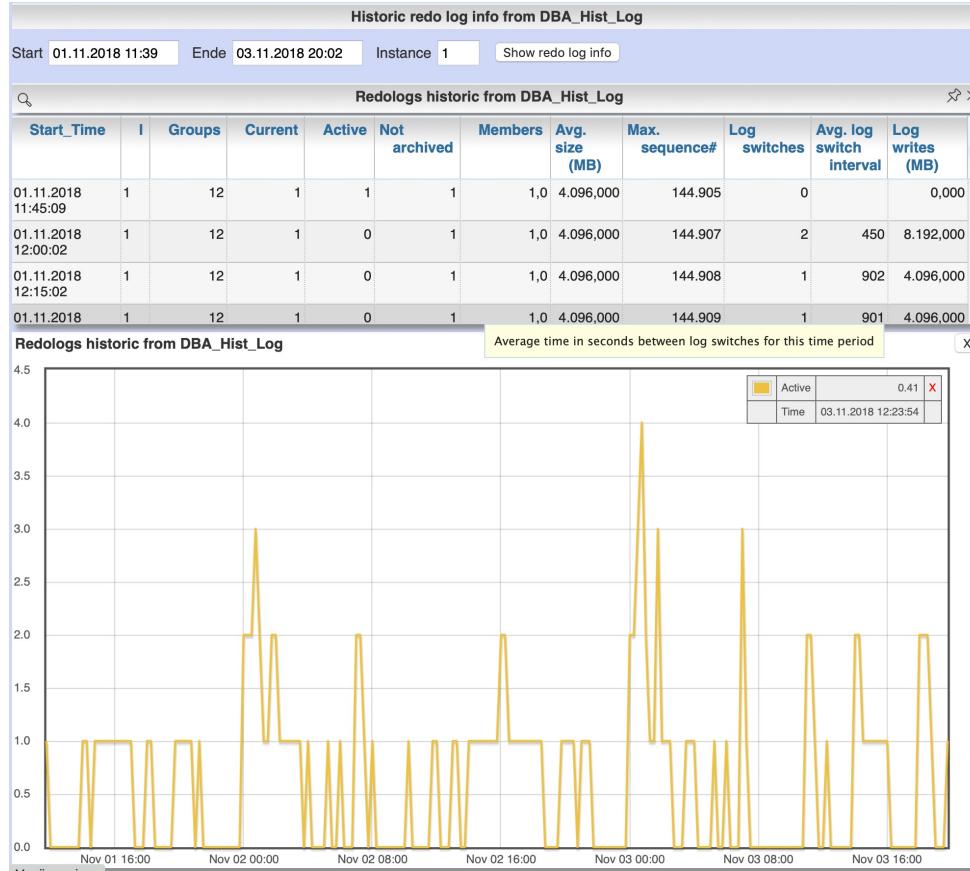
Unfortunately, this is also often the case in production systems..

Problem points :

- With only 3 redo log groups, there is a latent danger of "cannot switch redo log file" after the current log file has been written to full by the logwriter process.
Until a free redo log file is available again, all commit operations are stopped. (GAU for OLTP systems)
- 200 MB default size can be much too small
(Optimization target > 10 seconds between log switches)

Redo logs utilization in history

Menü: „DBA general“ / „Redologs“ / „Historic“



Evaluation of the utilization of the redo logs in the AWR cycle.

- Current + Active should normally never reach the number of available redo log groups
- Cross check also via alert.log with search for "cannot allocate new log" to detect possible problems within the AWR cycles
- This [blog post](#) describes the problem in detail

Agenda

- Presentation of the used tool "Panorama"
- Top/down analysis of session activities
- Executed SQL statements with detail analysis
- Segment statistics: Recording of characteristic values for objects (tables, indexes ...)
- Capacity and utilization of CPU, memory and I/O system, configuration of the instance
- Dragnet investigation: Systematic scan of the system for performance antipatterns

Dragnet investigation of performance antipatterns

OSP
Otto Group Solution Provider

Menü „Spec.additions“ / „Dragnet investigation“

Dragnet investigation for performance bottlenecks and usage of anti-pattern

Select dragnet-SQL for execution Filter: Include description Search

2. Detection of possibly unnecessary indexes

- 1. Detection of indexes not used for access or ensurance of uniqueness
- 2. Detection of indexes with only one or little key values in index**
- 3. Detection of indexes with multiple indexed columns
- 4. Detection of unused indexes by MONITORING USAGE
- 5. Detection of indexes without MONITORING USAGE
- 6. Detection of indexes with unnecessary columns because of pure selectivity
- 7. Coverage of foreign-key relations by indexes (detection of potentially unnecessary indexes)

Detection of indexes with only one or little key values in index

Indexes with only one or little key values may be unnecessary.
Exception: Indexes with only one key value may be useful for differentiation between NULL and NOT NULL.
Indexes with only one key value and no NULLS in indexed columns may be definitely removed.
If used for ensurance of foreign keys you can often relinquish on these index because resulting FullTableScan on referencing table in case of delete on referenced table may be accepted.

Min. number of rows in index	100000
Max. number of key values in index	10

[Do selection](#) [Show SQL](#)

Performance optimization: reactive event-driven or preventive?

Motivation

- Detection of all further occurrences of a once analyzed problem in the system
- Solutions as simple as possible to implement without interfering with architecture and design
- Fixing identified easy-to-solve issues system-wide instead of step-by-step after escalation
- Convenient embedding in further analysis workflow

Extensible with custom queries, further details in this [blog post](#)

Example: Unused and unnecessary indexes

Menü „Spec.additions“ / „Dragnet investigation“: Point 1.2.4



Detection of the use of indexes by SQL statements

- ALTER INDEX xxx MONITORING USAGE activates logging of index usage by SQL
- This allows a clear statement: Index has never been used since x days in 1st-level SQL
- Restriction: Recursive accesses to the index when checking a foreign key are not logged here.

For continuous monitoring of the usage state of indexes, cyclic execution of a script is recommended :

- Reactivation of the index monitoring after x days if a usage of the index was detected
- So that detection from when a previously active index is no longer used
- Execution of ALTER INDEX xxx MONITORING USAGE leads to invalidation of all cursors using this index and thus to load peaks when parsing again. This can be critical in high traffic OLTP systems.
- Therefore reactivation per ALTER INDEX xxx MONITORING USAGE only for indices which are currently not included in execution plans of the SGA
- [Blog post](#) contains the PL/SQL script for cyclic reactivation of the monitoring state as well as SQL queries

Conclusion: Automated detection of dispensable indexes is feasible.

Non-Unique indexes without foreign key and without usage for x days can be removed securely.

Final word



Panorama only has read-only access to the database and does not require its own PL/SQL objects.

So you can test and understand the functions without any risk.
Feel free to try it out.

Tip:

Started with environment variable "PANORAMA_LOG_LEVEL= debug" all SQL statements executed by Panorama are logged in the original in the log output of the Panorama server.

Description incl. Download link:

<https://rammpeter.github.io/panorama.html>

Docker image

<https://hub.docker.com/r/rammpeter/panorama>

Blog on the subject:

<https://rammpeter.blogspot.com>

Thank you for your interest

Otto Group Solution Provider (OSP) GmbH
Freiberger Str. 35 | 01067 Dresden
Telefon +49 (0)351 49723 0
osp.de