

What Did that SQL Query Do?

Oracle Edition

When investigating database performance issues related to SQL queries, it is often useful to know the execution plan of the statement. However, unless it was executed recently, we only have the execution plan that we can generate via the explain plan command. This may or may not be the actual plan that was used when the SQL ran. Using Foglight Performance Investigator for Oracle, we can easily go back and review *actual* plans that were used when the statement was executed. In addition, we can compare different plans as they change over time due to database object changes, optimizer statistics, etc.



Performance Tree

Tops: 25

History | Change Tracking

View as PDF

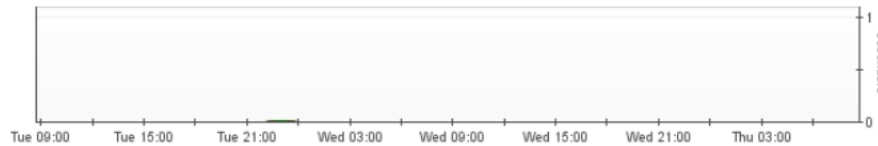
Instance View

- SQL Statements
 - SELECT d.file_name AS datafile_...
 - select sum(sizemegabytes) total...
 - call dbms_stats.gather_database...
 - SELECT * FROM (SELECT t.table...
 - MERGE /*+ dynamic_sampling(S...
 - /* SQL Analyze(40,1) */ SELECT...
 - SQL ID: 00p9qn448ras6
 - SQL ID: 751537thnawvw
 - /* SQL Analyze(40,1) */ select /...
 - delete from histgrm\$ where obj...
 - SELECT sqlset_row(sql_id, force...
 - insert into sys.wri\$optstat_histo...
 - /* SQL Analyze(1) */ select /*+...
 - SELECT t.name undo_tbs, d.unc...
 - SELECT TABLESPACE_ID, HEADI...
 - /* SQL Analyze(1) */ select /*+...
 - select sql_id, plan_hash_value, b...
 - /* SQL Analyze(1) */ select /*+...
 - select null as nrw,null as nlb,cour...
 - SQL ID: 37g281wz56rv3
 - select sum(decode(NAME, 'redo...
 - truncate table sys.wri\$heatmap...
 - select i.obj#,i.ts#,i.file#,i.block#...
 - select s.active_sessions + 0 activ...
 - SQL ID: 13ys8ux8xvrbm

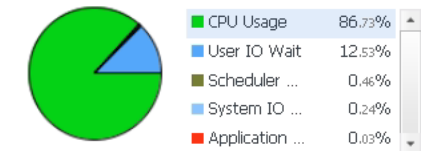
Dimension Filter: Instance View > SQL Statements

Resource Consumption

Baseline | Breakdown



Resource Breakdown



Top Wait Events

Overview | Blocking History

Top SQL Statements

Select Metric | View Full Text | Analyze Plan | Tune SQL | Compare | Drill to SQL Statement | Export to CSV

SQL Statements	Active Time	Active Time Percent	Average
SELECT d.file_name AS datafile_name, t.tablespace_name AS tables... = t.tablespace_name AND t.con_id=f.con_id AND rownum <= 500	81.03	36.10	
select sum(sizemegabytes) total_allocated_space,max(autoextensib...e_space GROUP BY file_id) f WHERE d.file_id = f.file_id(+)	66.97	29.84	
call dbms_stats.gather_database_stats_job_proc ()	29.31	13.06	
SELECT * FROM (SELECT t.tablespace_name tablespace_name, t.BLOCK..., 'SYSTEM', ' '),DECODE(:5 ,1,'SYSAUX', ' ') and rownum <= 500	24.11	10.74	
MERGE /*+ dynamic_sampling(ST 4) dynamic_sampling_est_cdh(ST) /*...SS, 100*:B16 OSIZE, I.OBJ# OBJ#, 20 TYPE#, :B18 + :B13 AFLAGS	9.63	4.29	
/* SQL Analyze(40,1) */ SELECT object_id, con_id, (NVL(owner, 'N...:79 AND con_id = :80) OR (object_id = :81 AND con_id = :82)	2.87	1.28	
SQL ID: 00p9qn448ras6	2.69	1.20	
SQL ID: 751537thnawvw	2.18	0.97	
/* SQL Analyze(40,1) */ select /*+ RULE */ version "db_version",...select count(*) cell_count from sys.v_\$cell where rownum<2) c	0.88	0.39	
delete from histgrm\$ where obj#= :1 and intcol#= :2 and row#= :3	0.65	0.29	

In the screenshot above, a time period was isolated and the top SQL statements sorted by Active Time are shown. To review the execution plans, all you need to do is pick a SQL and click "Analyze Plan".



Performance Tree

Tops: 25

History **Change Tracking**

Dimension Filter: Instance View > SQL Statements

- Instance View
 - SQL Statements
 - SELECT d.file_name AS datafile...
 - select sum(sizemegabytes) total...
 - call dbms_stats.gather_database...
 - SELECT * FROM (SELECT t.table...
 - MERGE /*+ dynamic_sampling(S...
 - SQL ID: 751537thnawww
 - SQL ID: 7hu2k3a31b6j7
 - delete from histgrm\$ where obj...
 - SQL ID: 6x37axnz8hwnn
 - SQL ID: 02ft29j09k6jq
 - SQL ID: 6wrwqq7jkmv3w
 - SQL ID: 9jbk8duuad16j
 - insert into sys.wri\$_optstat_histo...
 - SELECT sqlset_row(sql_id, force...
 - select sql_id, plan_hash_value, b...
 - SQL ID: dvbv42b3hfyru
 - select /*+ no_parallel_index(t, "...
 - SQL ID: 49mbt9756gjv2
 - SELECT TABLESPACE_ID, HEADI...
 - SELECT t.name undo_tbs, d.unc...
 - SQL ID: 0am5c070vn5ds
 - SQL ID: 28q8vcz753k3d
 - SQL ID: 5mnsfrvbs2n41
 - SQL ID: 1ng5b8npftgbw
 - /* SQL Analyze(1) */ select /*+...

Resource Consumption



Categories

<input checked="" type="checkbox"/>	Execution Plan	8	●
<input type="checkbox"/>	Oracle Configuration	62	●
<input type="checkbox"/>	Oracle Schema	10,964	●
<input type="checkbox"/>	System Configuration	0	●
<input type="checkbox"/>	User Defined	0	●

Add Change

Date	Description
12/4/15 10:43 PM	New plan(s) found for statement: 02ft29j09k6jq
12/4/15 10:43 PM	New plan(s) found for statement: 52ygatmzx2pww
12/20/15 6:13 AM	New plan(s) found for statement: 5x9mhpz2xv75g
12/6/15 2:13 PM	New plan(s) found for statement: MERGE /*+ dynamic_sampling(ST 4) dynamic_sampling_est_cdn(ST) */...SS, 100*:B16 OSIZE, I.OBJ# OBJ#, 20 TYPE#, :B18 + :B1...
12/20/15 1:43 AM	New plan(s) found for statement: SELECT d.file_name AS datafile_name, t.tablespace_name AS tables... = t.tablespace_name AND t.con_id=f.con_id AND rownum <...
12/12/15 9:28 PM	New plan(s) found for statement: select job_name job_identifier, 'DBMS_SCHEDULER' job_type, JOB_C... from cdb_jobs WHERE last_date is not null and rownum <= 5
12/28/15 10:42 PM	New plan(s) found for statement: select sum(sizemegabytes) total_allocated_space,max(autoextensib...e_space GROUP BY file_id) f WHERE d.file_id = f.file_id(+)
12/1/15 3:48 PM	New plan(s) found for statement: select sum(sizemegabytes) total_allocated_space,max(autoextensib...e_space GROUP BY file_id) f WHERE d.file_id = f.file_id(+)

You can also review changes to execution plans by switching to “Change Tracking” from the “History” view. Execution plan changes are overlaid on the timeline and workload baseline graph, and you can simply click into one of the changes in the grid.



1/28/16 4:11 AM Actual 806138454 d77s9jkcw3w3
Resolving Date Type Plan Hash Value SQL ID

Compare Plans Generate Plan

Plan Analysis

Total cost: 8,525 | Total I/O cost: 6,882 | Total CPU cost: 43,557,241,151

Plan Details Operation Analysis Object Analysis

Operation	Object Name	Object Type	Cost	CPU Cost	I/O Cost	Cardinality	Bytes	Time (seconds)	Temp Space	Access Predicates
SELECT STATEMENT			5.98 %	0	0	0	0	0	0	
COUNT STOPKEY			0.00 %	0	0	0	0	0	0	
HASH JOIN OUTER			5.98 %	2,418,925,465	447	227	78,088	1	0	"T"."TABLESPACE_NAME"="FS"."TABLESPACE_NAME" AND "T".
HASH JOIN OUTER			5.00 %	225,969,448	420	5	1,245	1	0	"T"."TABLESPACE_NAME"="DF"."TABLESPACE_NAME" AND "T".
HASH JOIN OUTER			4.87 %	146,952,435	411	5	810	1	0	"T"."TABLESPACE_NAME"="R"."TABLESPACE_NAME" AND "T".
HASH JOIN OUTER			0.08 %	39,115,253	6	5	650	1	0	"T"."TABLESPACE_NAME"="U"."TABLESPACE_NAME" AND "T".
VIEW	SYS.CDB_TABLESPACES	VIEW	0.04 %	57,980	3	5	435	1	0	
NESTED LOOPS			0.04 %	57,980	3	5	175	1	0	
TABLE ACCESS FULL	SYS.TS\$	CLUSTER	0.04 %	40,230	3	5	160	1	0	
FIXED TABLE FIXED INDEX	SYS.X\$KCFI1TSA (ind:1)	TABLE (FIXED)	0.00 %	3,550	0	1	3	0	0	
VIEW			0.05 %	38,456,023	3	1	43	1	0	

SQL Text

```

SELECT * FROM
  (SELECT t.tablespace_name tablespace_name,
    t.BLOCK_SIZE block_size,
    case when contents='CONTENTS' and EXTENT_MANAGEMENT='LOCAL ' then NVL(u.bytes,0) else NVL(df.user_bytes,0) - NVL (fs.bytes, 0) END /1024/ 1024 used,
    case when contents='CONTENTS' and EXTENT_MANAGEMENT='LOCAL ' then nvl(u.bytes - r.space*t.BLOCK_SIZE,0) else nvl(df.user_bytes - NVL (fs.bytes, 0) - r.space*t.

```

This drilldown shows you Plan Details, Operator Analysis and Object Analysis on separate tabs.



1/28/16 4:11 AM | Actual | 806138454 | d77s9jkc3w3
Resolving Date | Type | Plan Hash Value | SQL ID

Compare Plans | Generate Plan

Plan Analysis

Total cost: 8,525 | Total I/O cost: 6,882 | Total CPU cost: 43,557,241,151

Plan Details | **Operation Analysis** | Object Analysis

Operation	Cost	CPU Cost	I/O Cost	Cardinality	Time (seconds)	Temp Space	Bytes
HASH JOIN OUTER	38.65 %	10,156,730,722	3,030	932	8	0	315,962
VIEW	19.16 %	19,659,870,792	1,121	273,261	19	0	15,285,784
HASH JOIN	10.31 %	657,668,407	862	26,764	12	0	2,560,464
HASH GROUP BY	8.90 %	8,984,606,287	525	90,931	8	0	3,911,281
NESTED LOOPS	6.72 %	3,322,846,030	487	2,397	29	0	121,645
TABLE ACCESS FULL	5.24 %	57,005,973	446	94,629	20	0	1,966,556
TABLE ACCESS BY INDEX ROWID	4.62 %	55,546,650	393	2	2	0	43
TABLE ACCESS CLUSTER	0.06 %	63,669	5	7	3	0	174
INDEX FULL SCAN	0.05 %	79,284	4	254	4	0	3,492
NESTED LOOPS SEMI	0.04 %	22,614	3	1	1	0	29
MERGE JOIN CARTESIAN	0.04 %	195,457	3	20	1	0	1,320
INDEX RANGE SCAN	0.02 %	15,293	2	1	1	0	9

SQL Text

```

SELECT * FROM
  (SELECT t.tablespace_name tablespace_name,
    t.BLOCK_SIZE block_size,
    case when contents='CONTENTS' and EXTENT_MANAGEMENT='LOCAL ' then NVL(u.bytes,0) else NVL(df.user_bytes,0) - NVL (fs.bytes, 0) END /1024/ 1024 used,
    case when contents='CONTENTS' and EXTENT_MANAGEMENT='LOCAL ' then nvl(u.bytes - r.space*t.BLOCK_SIZE,0) else nvl(df.user_bytes - NVL (fs.bytes, 0) - r.space*t.

```



1/28/16 4:11 AM |
 Actual |
 806138454 |
 d77s9jkcw3w3
Resolving Date | Type | Plan Hash Value | SQL ID

Compare Plans |
 Generate Plan

Plan Analysis

Total cost: 8,525 | Total I/O cost: 6,882 | Total CPU cost: 43,557,241,151

Plan Details |
 Operation Analysis |
Object Analysis

Name	Type	Cost	CPU Cost	I/O Cost	Cardinality	Time (seconds)	Temp Space	Bytes
SYS.OBJ\$	TABLE	55.67 %	111,073,572	784	92,384	2	0	1,940,064
SYS.CDB_RECYCLEBIN	VIEW	28.61 %	68,090,174	402	729	1	0	16,038
SYS.DBA_FREE_SPACE	VIEW	8.78 %	2,057,954,902	72	90,663	4	0	2,719,890
SYS.TS\$	CLUSTER	2.48 %	453,869	35	39	15	0	770
SYS.RECYCLEBIN\$	TABLE	1.06 %	1,039,611	15	2,196	3	0	25,620
SYS.X\$KTFBUE	TABLE (FIXED)	0.92 %	500,000,000	0	100,000	1	0	6,500,000
SYS.DBA_DATA_FILES	VIEW	0.57 %	1,543,635	8	5	1	0	1,405
SYS.CDB_TABLESPACES	VIEW	0.42 %	115,980	6	10	2	0	480
SYS.FILE\$	TABLE	0.28 %	17,202	4	5	2	0	56
SYS.X\$KTFBUE (ind:1)	TABLE (FIXED)	0.28 %	162,080,000	0	25	1	0	1,950
SYS.USER_EDITIONING\$	TABLE	0.28 %	15,222	4	2	2	0	12
SYS.I_USER2	INDEX (UNIQUE)	0.21 %	70,763	3	247	3	0	3,464

SQL Text

```

SELECT * FROM
  (SELECT t.tablespace_name tablespace_name,
    t.BLOCK_SIZE block_size,
    case when contents='CONTENTS' and EXTENT_MANAGEMENT='LOCAL ' then NVL(u.bytes,0) else NVL(df.user_bytes,0) - NVL (fs.bytes, 0) END /1024/ 1024 used,
    case when contents='CONTENTS' and EXTENT_MANAGEMENT='LOCAL ' then nvl(u.bytes - r.space*t.BLOCK_SIZE,0) else nvl(df.user_bytes - NVL (fs.bytes, 0) - r.space*t.
  
```

If there is more than 1 historical execution plan available, the “Compare Plans” button will be active. You can click into that drilldown to compare historical plans. Note that there will only be a historical plan stored for the first time it was detected as being changed.



Databases > Performance > Execution

alvsdcw24-ORAPROD12C

1/1/16 10:01 PM
Resolving Date

Actual
Type

Plan Analysis

Total cost: 147,635 | Total I/O

Plan Details | Operation Analysis

Name
SYS.SYS_DBA_SEGS
SYS.SYS_OBJECTS
SYS.TAB\$
SYS.IND\$
SYS.CLU\$
SYS.OBJ\$
SYS.LOB\$
SYS.SEG\$
SYS.DBA_LOBS
SYS.TS\$
SYS.I_OBJ2
SYS.USER\$

SQL Text

SELECT TABLESPACE_ID, HEAD

Execution Plan Comparison

1/1/16 10:01 PM

Analyze Plan

10/24/15 10:09 AM

Analyze Plan

- Cost Changed 0 SELECT STATEMENT
Cost: 4,513 Bytes: 0 Cardinality:0
 - Cost Changed 1 SORT UNIQUE
Cost: 4,513 Bytes: 17,763 Cardinality:63
 - 2 UNION-ALL
Cost: 0 Bytes: 0 Cardinality:0
 - Cost Changed 3 HASH JOIN
Cost: 2,192 Bytes: 15,960 Cardinality:40
 - 4 VIEW
SYS.DBA_LOBS
Cost: 44 Bytes: 396 Cardinality:2
 - 5 UNION-ALL
Cost: 0 Bytes: 0 Cardinality:0
 - 6 HASH JOIN
Cost: 13 Bytes: 176 Cardinality:1
 - 7 NESTED LOOPS

- Cost Changed 0 SELECT STATEMENT
Cost: 4,502 Bytes: 0 Cardinality:0
 - Cost Changed 1 SORT UNIQUE
Cost: 4,502 Bytes: 17,103 Cardinality:63
 - 2 UNION-ALL
Cost: 0 Bytes: 0 Cardinality:0
 - Cost Changed 3 HASH JOIN
Cost: 2,180 Bytes: 15,162 Cardinality:38
 - 4 VIEW
SYS.DBA_LOBS
Cost: 44 Bytes: 396 Cardinality:2
 - 5 UNION-ALL
Cost: 0 Bytes: 0 Cardinality:0
 - 6 HASH JOIN
Cost: 13 Bytes: 176 Cardinality:1
 - 7 NESTED LOOPS

SQL Text

SELECT TABLESPACE_ID, HEADER_FILE, HEADER_BLOCK, SEGMENT_OBJD, SYS_DBA_SEGS.TABLESPACE_NAME FROM SYS_DBA_SEGS, DBA_LOBS WHERE DBA_LOBS.0

1, 2017 12 months | Reports

Powered by SQL PI

Compare Plans | Generate Plan

Space	Bytes
0	17,872
0	2,137,434
0	291,628
0	307,226
0	294
0	7,200,585
0	34,380
0	43,077
0	396
0	418
0	33
0	11,253

Close

NAME = :B1 AND DBA_LOBS.SEGI

You can then pick from the date/time pulldowns different historical plans to compare.



1/1/16 10:01 PM
Resolving Date

Actual
Type

2266419613
Plan Hash Value

Select Plan

Resolving Date	Type	Plan Hash Value
1/1/16 10:01 PM	Actual	2266419613
10/24/15 10:09 AM	Actual	2908081919
10/9/15 10:01 PM	Actual	736279828
8/27/15 10:01 PM	Actual	973229257
8/8/15 2:10 PM	Actual	2848504063

Analyze execution plan Cancel

Plan Analysis

Total cost: 147,635 | Total I/O cost: 138,479 | Total C...

Plan Details | Operation Analysis | Object Analysis

Name	Type
SYS.SYS_DBA_SEGS	VIEW
SYS.SYS_OBJECTS	VIEW
SYS.TAB\$	CLUSTER
SYS.IND\$	CLUSTER
SYS.CLU\$	CLUSTER
SYS.OBJ\$	TABLE
SYS.LOB\$	CLUSTER
SYS.SEG\$	CLUSTER
SYS.DBA_LOBS	VIEW
SYS.TS\$	CLUSTER
SYS.I_OBJ2	INDEX (UNIQUE)
SYS.USER\$	CLUSTER

seconds)	Temp Space	Bytes
5	0	17,872
5	0	2,137,434
9	0	291,628
7	0	307,226
3	0	294
10	0	7,200,585
7	0	34,380
27	0	43,077
1	0	396
26	0	418
5	0	33
15	0	11,253

SQL Text

```
SELECT TABLESPACE_ID, HEADER_FILE, HEADER_BLOCK, SEGMENT_OBJD, SYS_DBA_SEGS.TABLESPACE_NAME FROM SYS_DBA_SEGS, DBA_LOBS WHERE DBA_LOBS.OWNER = :B2 AND DBA_LOBS.TABLE_NAME = :B1 AND DBA_LOBS.SEGI
```

You may also analyze a specific plan by picking from the date/time dropdown shown above. This is great functionality to have because it lets you review plans that may have executed weeks or months ago, and also makes it very easy for you to detect when a plan has changed. If you also manage SQL Server databases, I've posted a similar tutorial on the site as well.