



# GLUENT SQL PERFORMANCE TESTING TOOLKIT

## Abstract

A guide on how to SQL performance test candidate tables for offloading

Karl Arao

## Table of Contents

<b>Installation .....</b>	<b>0</b>
Create a list of tables for collection.....	0
Create Performance Testing Objects .....	1
<b>Collect SQLs in PROD environment .....</b>	<b>2</b>
run_all.sql - Get topn SQL_IDs .....	2
gensql.sh - Get SQLD360 files.....	4
<b>Arrange Collected SQLs in DEV/QA environment .....</b>	<b>4</b>
run_arrange.sh - group SQL files by Table .....	4
<b>Execute Collected SQLs in DEV/QA environment.....</b>	<b>5</b>
baselineperf - Running a baseline .....	5
gluentperf - Running SQLs on Gluent hybrid views.....	6
<b>Mining the SQL Performance Repository and files .....</b>	<b>7</b>
SQL reports/operations .....	7
rpt_elap.sql .....	7
rpt_elap2.sql .....	7
rpt_elap3.sql .....	8
rpt_waits.sql .....	9
rpt_delete.sql.....	10
Export to HTML and Create pivot table .....	10
rpt_report_pivot.sql .....	10
Create pivot table .....	10
Validate Executed SQLs with List of Tables .....	12
prodperf – getting the PROD run times .....	13
Get Impala memory usage.....	14
<b>Packaging SQL troubleshooting files to Gluent Support.....</b>	<b>15</b>
<b>Appendices .....</b>	<b>16</b>
db_app_table.txt.....	16

## Installation

The zip file aeg-ctl-Gluent-SQL-Performance-Testing-Toolkit.zip contains the following:

- gluent\_sql\_perftest\_toolkit.zip
  - packaged tool for SQL perf testing
- HOWTO-aeg-ctl-Gluent-SQL-Performance-Testing-Toolkit.docx
  - the documentation/HOWTO
- doc\_templates folder
  - contains document templates for project tracking/reporting that are mentioned in this doc

It is best to unzip the tool (gluent\_sql\_perftest folder) in a shared directory for easy mining of the files/sub directories.

```
unzip gluent_sql_perftest.zip
```

## Create a list of tables for collection

The Gluent Advisor identifies the tables that are candidate for offloading. These tables are then consolidated in a master list and grouped by Database and App Team. And then evaluated by the SMEs/Project Team based on access patterns, retention, and app dependencies if they will be offloaded or not (include Y/N).

For performance testing these tables need to be refreshed (refreshed\_date) so that the table sizes are big enough to show comparable run times when test SQLs are executed (baselineperf).

Once refreshed and baselined, the tables are then offloaded to Hadoop and the test SQLs are executed again but on the Gluent hybrid views (gluentperf).

In summary each of the table goes from refreshed -> baselineperf -> offloaded -> gluentperf and this is recorded on the excel sheet for project tracking while the underlying performance statistics are recorded on the table GLUENT\_APP.GET\_RUN\_STATS. The final state of all tables should be gluentperf and all tables marked with this status are evaluated for production migration given that any related SQL issues that occurred are resolved/fixed which are also applied on migration/cutover.

	A	B	C	D	E	F	G	H	I	J	K
1	Prod DB Name	app team	Owner	Table Name	include (Y/N)	refreshed_date	notes	baseline	offloaded	gluentperf	Status (pending/collected/error/ baselineperf/glulentperf)
28	DWBS001P	REG1	STAGE	TRAF_TRAFFIC_8XX_DETAIL	Y	9/24/2018		9/26/2018	9/24/2018	9/26/2018	gluentperf
29	DWBS001P	CDIMS	STAGE	AT_VENDOR_DEMOGRAPHIC	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
30	DWBS001P	CDIMS	STAGE	FW_LU_QUAL	N	8/31/2018	156632308 rows	8/31/2018	9/12/2018	9/12/2018	baselineperf
31	DWBS001P	RNA	STAGE	ENS_REVENUE_DET	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
32	DWBS001P	RNA	STAGE	EPWF_PAYMENT	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
33	DWBS001P	RNA	STAGE	LATIS_NETWORKX_TAX_EVENT_DTL	Y	9/4/2018	2103787016 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
34	DWBS001P	RNA	STAGE	LATIS_NME_HDR	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
35	DWBS001P	RNA	STAGE	LATIS_NME_TRANS_DTL	Y	9/4/2018	107135567 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
36	DWBS001P	RNA	STAGE	RIB_USA6	N						baselineperf
37	DWBS001P	RNA	STAGE	RJF_BOOKED_REVENUE	N						error
38	DWBS001P	RNA	STAGE	RJF_JOURNAL_DETAIL	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
39	DWBS001P	EISPA	STAGE	CDW_CPROD10T_DLY	N						baselineperf
40	DWBS001P	EISPA	STAGE	CPROD10T_HIST	Y	9/4/2018	159624103 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
41	DWBS001P	EISPA	STAGE	PRODR_ORD_ORDER	Y	9/4/2018	17543 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
42	DWBS001P	EISPA	STAGE	PRODR_PRO_ACCT_CMP	Y	9/4/2018	253034648 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
43	DWBS001P	NI	STAGE	ASSIA_PORTS	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
44	DWBS001P	PSO	STAGE	SFDC_ACCOUNT	N						pending

The list of tables on the excel sheet should also be created on a text file called **db\_app\_table.txt** in the following format (also shown in [Appendix](#)):

```
cat sql/collect_arrange/db_app_table.txt
DWBS001P,REG1,TRAF_TRAFFIC_8XX_DETAIL
DWBS001P,CDIMS,AT_VENDOR_DEMOGRAPHIC
DWBS001P,CDIMS,FW_LU_QUAL
DWBS001P,RNA,ENS_REVENUE_DET
DWBS001P,RNA,EPWF_PAYMENT
DWBS001P,RNA,LATIS_NETWORKX_TAX_EVENT_DTL
DWBS001P,RNA,LATIS_NME_HDR
```

```
DWBS001P,RNA,LATIS_NME_TRANS_DTL
DWBS001P,RNA,RIB_USA6
DWBS001P,RNA,RJF_BOOKED_REVENUE
DWBS001P,RNA,RJF_JOURNAL_DETAIL
DWBS001P,EISPA,CDW_CPROD10T_DLY
DWBS001P,EISPA,CPROD10T_HIST
DWBS001P,EISPA,PRODR_ORD_ORDER
DWBS001P,EISPA,PRODR_PRD_ACCT_CMP
DWBS001P,NI,ASSIA_PORTS
```

This text file (under sql/collect\_arrange directory) will be needed later when arranging the SQL files to their corresponding table folders. This text file should be updated whenever there are new tables/updates on the excel sheet.

For Collect, Arrange, Execute sections of this HOWTO guide we will focus on DWBS database and only on these two tables

- STAGE.TRAF\_TRAFFIC\_8XX\_DETAIL
- STAGE.EPWF\_PAYMENT

## Create Performance Testing Objects

To record the elapsed times and performance statistics, the objects of the SQL Performance Test Tool must be installed.

The following objects are created:

- table gluent\_app.get\_run\_stats
- package gluent\_app.get\_snap\_time
  - procedure begin\_snap
  - procedure end\_snap

Execute as SYSDBA:

```
cd 01_install

sqlplus "/ as sysdba"
@run_stats_create.sql
```

All performance runs are recorded in a table called GLUENT\_APP.GET\_RUN\_STATS and from here we run SQLs to extract the test results and put it in excel for reporting purposes.

The table contains the following columns:

```
desc gluent_app.get_run_stats
Name                                                    Null?    Type
-----
TEST_NAME                                                VARCHAR2 (1000)
TEST_TYPE                                                VARCHAR2 (1000)
TEST_CATEGORY                                            VARCHAR2 (1000)
TEST_NOTES                                              VARCHAR2 (4000)
SNAP_TYPE                                                VARCHAR2 (1000)
SNAP_TIME                                                DATE
STAT_CLASS                                              VARCHAR2 (1000)
NAME                                                    VARCHAR2 (4000)
VALUE                                                    NUMBER
```

Here is the definition of the columns:

- TEST\_NAME – the file name of the extracted SQL, the file name that came from SQLD360 has the SQL\_ID in it
- TEST\_TYPE – baselineperf or gluentperf

- TEST\_CATEGORY – this column can be blank, or can be used to group application team/names
- TEST\_NOTES – this column can be blank, or can be used for any comments/notes about the run
- SNAP\_TYPE – this is automatically set to BEGIN or END when begin\_snap or end\_snap is called
- SNAP\_TIME - this is automatically set to current time when begin\_snap or end\_snap is called
- STAT\_CLASS – statistics class
- NAME – statistics name
- VALUE – statistics value

For each of the TEST\_NAME the following statistics information (STAT\_CLASS, NAME, VALUE) are collected:

- Elapsed – wall clock elapsed time
- CPU time – the CPU time is greater than Elapsed if parallelism was used
- Wait events
- SQL\_ID – this is the parsed SQL\_ID when executed in the database
- Tables Accessed - this is critical info for each run, and this shows us if only base table is used or both base and external table + Gluent aggregate objects

## Collect SQLs in PROD environment

Collecting the SQLs for performance testing is a two-step process.

- Get the topn SQL report for each table by the following:
  - total elap
  - elap/exec
  - exec
- Then gather SQLD360 files for each SQL\_ID

## run\_all.sql - Get topn SQL\_IDs

Having three data set gives us more info about the profile of the SQL\_IDs and guarantees that both frequently executed and long running ones are baselined and ran against Gluent objects.

The data we get on the data set have the following columns:

- DB
- PSCHEMA
- MODULE
- OBJ\_NAME
- SQL\_ID
- PLAN\_HASH\_VALUE
- FMS (force matching signature)
- SQL\_TEXT
- ELAPEXEC <-- topn SQLID
- EXECS <-- topn SQLID
- ETIME <-- topn SQLID
- CPUTIME
- IOTIME
- PIO
- LIO
- TIME\_RANK
- SQLDETAIL

To generate the data set (3 CSV output files). Do the following:

- 1) Create a working directory under 02\_collector

```
cd 02_collector/  
mkdir DWBS  
cd DWBS
```

- 2) Copy all files under sql/collect\_arrange directory to the working directory

```
cp ../../sql/collect_arrange/* .
```

- 3) Edit each of the run\_awr\* SQL file and put the table names for collection on the WHERE clause of SPACESQL

```
ls -ltr *sql  
run_awr_topsql_bigobj_topn_v3_by_elap_exec.sql  
run_awr_topsql_bigobj_topn_v3_by_elap.sql  
run_awr_topsql_bigobj_topn_v3_by_exec.sql  
run_all.sql
```

```
FROM dba_hist_sqlstat s,  
st_temp stt,  
(SELECT sql_id, object_name  
FROM dba_hist_sql_plan  
WHERE object_name in (  
'TRAF_TRAFFIC_8XX_DETAIL',  
'EPWF_PAYMENT'  
)) spacesql
```

- 4) As SYSDBA execute the run\_all.sql

```
sqlplus "/ as sysdba"  
@run_all.sql
```

Once the data set is gathered, the project team will arrange meetings with individual app teams and these output files can be used as an essential material to discuss what SQLs are critical for performance testing or if they have anything not included on the list that is critical to be tested.

```
ls -ltr *csv  
awr_topsql_bigobj_by_elap-dwbs001p4-pddcdbadm04.corp.intranet.csv  
awr_topsql_bigobj_by_elap_exec-dwbs001p4-pddcdbadm04.corp.intranet.csv  
awr_topsql_bigobj_by_exec-dwbs001p4-pddcdbadm04.corp.intranet.csv
```

The data output above and the app team supplied SQLs should complete the scope of SQL performance testing.

The 3 CSV data set above will also serve as a driver for the next step which will parse the SQL\_IDs to generate SQLD360 collection commands.

## gensql.sh - Get SQLD360 files

The next step is to gather SQLD360 output for each SQL\_ID

Follow the steps below:

- 1) On the working directory create **tablelist.txt** file with list of tables to be collected  
  

```
vi tablelist.txt
TRAF_TRAFFIC_8XX_DETAIL
EPWF_PAYMENT
```
- 2) Execute **gensql.sh** and pass the **tablelist.txt** as an argument. This will generate the **gensql360.sql** file.  
  

```
./gensql.sh tablelist.txt
```

Copy the **gensql360.sql** file to the SQLD360 directory

```
$ cp gensql360.sql /db_backup_denx3/p1/glueant/get_run_stats/glueant_sql_perftest/sql/SQLD360
```
- 3) Execute the **gensql360.sql** on the PROD database.  
  

```
$ cd /db_backup_denx3/p1/glueant/get_run_stats/glueant_sql_perftest/sql/SQLD360

$ sqlplus "/ as sysdba"
@gensql360.sql
```

The result is SQLD360 zip files like the one below:

```
ls -ltr sqld360*.zip
sqld360_537514_0pjav6qqywn3_382425_20181215_1646.zip
sqld360_537514_0y1mmq4gawshq_382425_20181215_1646.zip
```

## Arrange Collected SQLs in DEV/QA environment

The next step is move the zip files into their individual table folders.

## run\_arrange.sh - group SQL files by Table

Follow the steps below:

- 1) Move the SQLD360 zip files to the working directory  
  

```
$ cd /db_backup_denx3/p1/glueant/get_run_stats/glueant_sql_perftest/02_collector/SQLD360
$ mv sqld360*.zip /db_backup_denx3/p1/glueant/get_run_stats/glueant_sql_perftest/02_collector/DWBS
```
- 2) Execute the **run\_arrange.sh** script  
  

```
./run_arrange.sh
```

```
Opjavs6qqywn3 is in the list
creating and moving to directory DWBS001P/REG1/TRAF_TRAFFIC_8XX_DETAIL
0y1mmq4gawshq is in the list
creating and moving to directory DWBS001P/RNA/EPWF_PAYMENT

3) Verify the SQLs are moved to the table folders

find DWBS001P
DWBS001P
DWBS001P/RNA
DWBS001P/RNA/EPWF_PAYMENT
DWBS001P/RNA/EPWF_PAYMENT/sqlld360_537514_0y1mmq4gawshq_382425_20181215_1646.zip
DWBS001P/REG1
DWBS001P/REG1/TRAF_TRAFFIC_8XX_DETAIL
DWBS001P/REG1/TRAF_TRAFFIC_8XX_DETAIL/sqlld360_537514_0pjavs6qqywn3_382425_20181215_1646.
zip
```

This base folder can be moved to the DATABASES directory for execution of baselineperf and gluentperf.

## Execute Collected SQLs in DEV/QA environment

In this section, each SQLD360 file is unzipped and the standalone SQL (with binds) is copied to be used for benchmarking. The other data collected are placed under **archive** directory where we can backtrack/compare the performance of the SQL\_ID in the PROD environment if the baselineperf or gluentperf performance is slower. The metadata info is also collected which is useful to recreate the SQL\_ID objects to investigate the query logic.

When **run.sh** is executed all the SQLD360 SQL files available in the same directory are executed. That's why in the process of unzipping SQLD360 files all DML/DDI SQLs are placed under **dml\_sqls** directory.

Finally, it's a best practice to inspect each SQL files and test (by running the SQL standalone) if the binds work before executing **run.sh**. Also make sure they are pointed to the base tables when running baselineperf or the Gluent hybrid view for gluentperf.

## baselineperf - Running a baseline

To run the test SQLs on base tables. Follow the steps below:

```
1) On the table directory, copy and execute the unzip.sh file

$ cd DWBS001P/REG1/TRAF_TRAFFIC_8XX_DETAIL

$ ls -ltr
sqlld360_537514_0pjavs6qqywn3_382425_20181215_1646.zip

$ cp /db_backup_denx3/p1/gluent/get_run_stats/gluent_sql_perftest/sql/unzip.sh .
$ ./unzip.sh

$ ls -ltr
unzip.sh
00009_sqlld360_537514_0pjavs6qqywn3_2a_3_standalone_sql.sql
dml_sqls
archive

2) Then, copy the run.sh file and edit the GETRUNSTATSSQLDIR variable

$ cp /db_backup_denx3/p1/gluent/get_run_stats/gluent_sql_perftest/sql/run.sh .
```



```
# sql directory
GETRUNSTATSSQLDIR=/db_backup_denx3/p1/gluent/get_run_stats/gluent_sql_perftest/sql

3) Execute the run.sh file

$ ./run.sh
Usage: ./run.sh <baselineperf|gluentperf>
<adhoc|ctledw|reg1|cdims|rna|eispa|ni|sdsa|margin> <optional: "comment text">

$ ./run.sh baselineperf reg1 "run 1"

4) When run.sh completes, the planx and Gluent SQL Monitor instrumentation output are
dumped in a directory

$ ls -ld dir*
dir_00009_sqld360_537514_0pjav6qqywn3_2a_3_standalone_sql.sql

5) For better organization of files. Move all baselineperf files and directories in a
new directory called baselineperf

$ mkdir baselineperf
$ mv 000*sql dir* run.sh baselineperf

$ ls -ltr baselineperf/
dir_00009_sqld360_537514_0pjav6qqywn3_2a_3_standalone_sql.sql
00009_sqld360_537514_0pjav6qqywn3_2a_3_standalone_sql.sql
```

To get the elapsed times and performance statistics of the baselineperf SQLs follow the steps on [generating SQL reports](#).

## gluentperf - Running SQLs on Gluent hybrid views

To run the test SQLs on Gluent hybrid views. Follow the steps below:

```
1) Copy all baselineperf test SQLs and run.sh in a new directory called gluentperf

$ mkdir gluentperf
$ cp baselineperf/000*sql baselineperf/run.sh gluentperf

2) Rename all tables in the FROM clause to point to the Gluent hybrid views

3) Execute the run.sh file

$ ./run.sh
Usage: ./run.sh <baselineperf|gluentperf>
<adhoc|ctledw|reg1|cdims|rna|eispa|ni|sdsa|margin> <optional: "comment text">

$ ./run.sh gluentperf reg1 "run 1"
```

To get the elapsed times and performance statistics of the gluentperf SQLs follow the steps on [generating SQL reports](#).

## Mining the SQL Performance Repository and files

All performance related data are stored either in GLUENT\_APP.GET\_RUN\_STATS or individual table directories which can be mined by SQL or text/grep search.

### SQL reports/operations

#### rpt\_elap.sql

- This script outputs the TEST\_NAME, TYPE, CATEGORY, NOTES, END\_SNAP, ELAPSED, CPU\_USED, SQLID, TABLES\_ACCESSED
  - The TABLES\_ACCESSED column is shown in comma separated values
- Two parameters can be passed to this script (TEST\_NAME, TABLES\_ACCESSED), if left blank it will output all data of GLUENT\_APP.GET\_RUN\_STATS

```
@rpt_elap
Enter value for test_name:
Enter value for tables_accessed: %TRAF_TRAFFIC_8XX_DETAIL%
```

```
dwbs001sl(sys): @rpt_elap
Enter value for test_name:
Enter value for tables_accessed: %TRAF_TRAFFIC_8XX_DETAIL%

TEST_NAME                                TYPE          CATEGORY    NOTES          END_SNAP          ELAPSED    CPU_USED    SQLID          TABLES_ACCESSED
-----
00009_sqld360_537514_csklahy7jpn76_2a_3_ baselineperf  regl        runl          08/30/18 16:22:42      14         1.53      csklahy7jpn76  GLUENT_APP.GET_RUN_STATS,STAGE.TRAF_TRAFFI
C_8XX_DETAIL_GLT
00009_sqld360_537514_gdffya2scry6a_2a_3_ baselineperf  regl        runl          08/30/18 16:23:05        1         .12      gc72bl9sptdvm  STAGE.TRAF_TRAFFIC_8XX_DETAIL_GLT
00009_sqld360_537514_csklahy7jpn76_2a_3_ baselineperf  regl        09-26        09/26/18 16:36:47      93        13.78     d9qg8lj2dv56w  STAGE.TRAF_TRAFFIC_8XX_DETAIL_GLT
00009_sqld360_537514_gdffya2scry6a_2a_3_ baselineperf  regl        09-26        09/26/18 16:38:31      95        13.86     2w0jxu35fkprq  STAGE.TRAF_TRAFFIC_8XX_DETAIL_GLT
00009_sqld360_537514_csklahy7jpn76_2a_3_ gluentperf    regl        09-26        09/26/18 15:25:55      51         7.37     csklahy7jpn76  STAGE.TRAF_TRAFFIC_8XX_DETAIL_GLT,STAGE_H.
TRAF_TRAFFIC_8XX_DETAIL_G_SMD6
00009_sqld360_537514_gdffya2scry6a_2a_3_ gluentperf    regl        09-26        09/26/18 15:27:17      60         6.78     g67bws453lptn  STAGE.TRAF_TRAFFIC_8XX_DETAIL_GLT,STAGE_H.
TRAF_TRAFFIC_8XX_DETAIL_G_2UJB

6 rows selected.
```

#### rpt\_elap2.sql

- This script outputs the TEST\_NAME, TYPE, CATEGORY, END\_SNAP, INFO, TABLES\_ACCESSED
  - The INFO column is concatenation of SQLID, ELAPSED, CPU\_USED, TEST\_NOTES
  - The TABLE column shows distinct table names accessed
  - The TABLES\_ACCESSED shows the detailed tables accessed (both base table and Gluent objects)
- Two parameters can be passed to this script (TEST\_NAME, TABLES\_ACCESSED), if left blank it will output all data of GLUENT\_APP.GET\_RUN\_STATS
- In addition to SQL run statistics the bottom section shows the tables that have been offloaded so far, the key, and high watermark value

```
@rpt_elap2
Enter value for test_name: %9hgts04h5msvv%
Enter value for tables_accessed: %EPWF_PAYMENT%
```

```

Hvbs001sl(sys): @rpt_elap2
Enter value for test_name: %9hgts04h5msvv%
Enter value for tables_accessed: %EPWF_PAYMENT%

```

TEST_NAME	TYPE	CATEGORY	END_SNAP	INFO	TABLE_NAME	TABLES_ACCESSED
00030_sqld360_537514_9hgts04h5msvv_2a_baselineperf			07/26/18 15:31:55	2kg28hnhjycrf,2,32,	EPWF_PAYMENT	STAGE.EPWF_PAYMENT
		rna	08/31/18 15:11:30	2kg28hnhjycrf,2,42,08-31	EPWF_PAYMENT	STAGE.EPWF_PAYMENT_GLT
g-00030_sqld360_537514_9hgts04h5msvv_2_gluentperf		rna	09/05/18 20:33:59	5jy239nnxy08p,106,46.15,09-05	EPWF_PAYMENT	STAGE.EPWF_PAYMENT_GLT
			09/06/18 21:05:58	5jy239nnxy08p,101,45.24,09-06	EPWF_PAYMENT	STAGE_H.EPWF_PAYMENT_GLT_EXT
			09/17/18 14:02:52	5jy239nnxy08p,44,53.43,09-17 px4	EPWF_PAYMENT	STAGE_H.EPWF_PAYMENT_GLT_EXT
			09/17/18 14:07:19	5jy239nnxy08p,33,50.36,09-17 px4	EPWF_PAYMENT	STAGE_H.EPWF_PAYMENT_GLT_EXT
			09/17/18 14:07:58	5jy239nnxy08p,30,49.77,09-17 px4	EPWF_PAYMENT	STAGE_H.EPWF_PAYMENT_GLT_EXT

12 rows selected.

  

HYBRID_OWNER	OFFLOADED_OWNER	OFFLOADED_TABLE	IKEY	HIGH_VALUE
STAGE_H	STAGE	ASSIA_PORTS_GLT	META_END_EFF_TMSTMP	2018-07-01 00:00:00 (TmStmp)
STAGE_H	STAGE	AT_VENDOR_DEMOGRAPHIC_GLT	META_LOAD_TMSTMP	2018-05-01 00:00:00 (TmStmp)
STAGE_H	STAGE	CPROD10T_HIST_GLT	META_END_EFF_TMSTMP	2018-07-01 00:00:00 (TmStmp)
STAGE_H	STAGE	ENS_REVENUE_DET_L_GLT	META_LOAD_DTTM	2018-07-01 (Date)
STAGE_H	STAGE	EPWF_PAYMENT_GLT	META_LOAD_TMSTMP	2018-07-01 00:00:00 (TmStmp)
STAGE_H	STAGE	FW_LU_QUAL_GLT	META_END_EFF_TMSTMP	2018-07-01 00:00:00 (TmStmp)
STAGE_H	STAGE	LATIS_NETWORK_TAX_EVTNT_DTL_GLT	META_BEGIN_EFF_TMSTMP	2018-07-01 00:00:00 (TmStmp)
STAGE_H	STAGE	LATIS_NME_HDR_GLT	META_LOAD_TMSTMP	2018-07-01 00:00:00 (TmStmp)
STAGE_H	STAGE	LATIS_NME_TRANS_DTL_GLT	META_LOAD_TMSTMP	2018-07-01 00:00:00 (TmStmp)
STAGE_H	STAGE	PRODR_ORD_ORDER_GLT	META_END_EFF_TMSTMP	2018-07-01 00:00:00 (TmStmp)
STAGE_H	STAGE	PRODR_PRD_ACCT_CHP_GLT	META_END_EFF_DTTM	2018-07-01 (Date)
STAGE_H	STAGE	RJF_JOURNAL_DETAIL_GLT	META_LOAD_TMSTMP	2018-07-22 00:00:00 (TmStmp)
STAGE_H	STAGE	SFDC_ACCOUNT_GLT	META_END_EFF_DTTM	2018-07-01 (Date)
STAGE_H	STAGE	SFDC BILLING ACCOUNT_C_GLT	META_END_EFF_DTTM	2018-07-01 (Date)
STAGE_H	STAGE	TRAF_TRAFFIC_8XX_DETAIL_GLT	META_LOAD_TMSTMP	2018-05-01 00:00:00 (TmStmp)

15 rows selected.

### rpt\_elap3.sql

- This script outputs the TEST\_NAME, TYPE, CATEGORY, NOTES, END\_SNAP, ELAPSED, CPU\_USED, SQLID, TABLE\_NAME
  - The TABLE\_NAME column shows distinct table names accessed
- Two parameters can be passed to this script (TEST\_NAME, TABLES\_ACCESSED), if left blank it will output all data of GLUENT\_APP.GET\_RUN\_STATS
- In addition to SQL run statistics the bottom section shows the tables that have been offloaded so far, the key, and high watermark value
- This script is called by rpt\_report\_pivot.sql to export data to HTML that can be used to create a pivot table for reporting

```

@rpt_elap3
Enter value for 1: %9hgts04h5msvv%
Enter value for 2: %EPWF_PAYMENT%

OR

@rpt_elap3 %9hgts04h5msvv% %EPWF_PAYMENT%

```

```
dwbs001sl(sys): @rpt_elap3
Enter value for 1: %9hgts04h5msvv%
Enter value for 2: %EPWF_PAYMENT%
```

TEST_NAME	TYPE	CATEGORY	NOTES	END_SNAP	ELAPSED	CPU_USED	SQID	TABLE_NAME
00030_sqld360_537514_9hgts04h5msvv_2a_24	baselineperf			07/26/18 15:31:55	2	.32	2kg28hnhjycrf	
		rna	08-31	08/31/18 15:11:30	2	.42	2kg28hnhjycrf	EPWF_PAYMENT
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	rna	09-05	09/05/18 20:33:59	106	46.15	5jy239nnxy08p	EPWF_PAYMENT
			09-06	09/06/18 21:05:58	101	45.24	5jy239nnxy08p	EPWF_PAYMENT
			09-17 px4	09/17/18 14:02:52	44	53.43	5jy239nnxy08p	EPWF_PAYMENT
				09/17/18 14:07:19	33	50.36	5jy239nnxy08p	EPWF_PAYMENT
				09/17/18 14:07:58	30	49.77	5jy239nnxy08p	EPWF_PAYMENT

12 rows selected.

HYBRID_OWNER	OFFLOADED_OWNER	OFFLOADED_TABLE	IKEY	HIGH_VALUE
STAGE_H	STAGE	ASSIA_PORTS_GLT	META_END_EFF_TMSTMP	2018-07-01 00:00:00 (TmStamp)
STAGE_H	STAGE	AT_VENDOR_DEMOGRAPHIC_GLT	META_LOAD_TMSTMP	2018-05-01 00:00:00 (TmStamp)
STAGE_H	STAGE	CPROD10T_HIST_GLT	META_END_EFF_TMSTMP	2018-07-01 00:00:00 (TmStamp)
STAGE_H	STAGE	ENS_REVENUE_DET_L_GLT	META_LOAD_DTTM	2018-07-01 (Date)
STAGE_H	STAGE	EPWF_PAYMENT_GLT	META_LOAD_TMSTMP	2018-07-01 00:00:00 (TmStamp)
STAGE_H	STAGE	FW_LU_QUAL_GLT	META_END_EFF_TMSTMP	2018-07-01 00:00:00 (TmStamp)
STAGE_H	STAGE	LATIS_NETWORK_TAX_EVNT_DTL_GLT	META_BEGIN_EFF_TMSTMP	2018-07-01 00:00:00 (TmStamp)
STAGE_H	STAGE	LATIS_NME_HDR_GLT	META_LOAD_TMSTMP	2018-07-01 00:00:00 (TmStamp)
STAGE_H	STAGE	LATIS_NME_TRANS_DTL_GLT	META_LOAD_TMSTMP	2018-07-01 00:00:00 (TmStamp)
STAGE_H	STAGE	PRODR_ORD_ORDER_GLT	META_END_EFF_TMSTMP	2018-07-01 00:00:00 (TmStamp)
STAGE_H	STAGE	PRODR_PRD_ACCT_CMP_GLT	META_END_EFF_DTTM	2018-07-01 (Date)
STAGE_H	STAGE	RJF_JOURNAL_DETAIL_GLT	META_LOAD_TMSTMP	2018-07-22 00:00:00 (TmStamp)
STAGE_H	STAGE	SFDC_ACCOUNT_GLT	META_END_EFF_DTTM	2018-07-01 (Date)
STAGE_H	STAGE	SFDC BILLING ACCOUNT_C_GLT	META_END_EFF_DTTM	2018-07-01 (Date)
STAGE_H	STAGE	TRAF_TRAFFIC_8XX_DETAIL_GLT	META_LOAD_TMSTMP	2018-05-01 00:00:00 (TmStamp)

15 rows selected.

## rpt\_waits.sql

- This script outputs the TEST\_NAME, TYPE, END\_SNAP, STAT\_CLASS, NAME, DELTA
  - STAT\_CLASS and NAME contain the definition of measures
  - DELTA contains the measure value
- Two parameters can be passed to this script (value of END\_SNAP, TEST\_NAME). If left blank it will output all data of GLUENT\_APP.GET\_RUN\_STATS
- This script can be used to characterize the response time profile of a benchmark run or comparing across multiple benchmarks

```
@rpt_waits
Enter value for end_snap_time_filter: 09/17/18 14:07:19
Enter value for test_name:
```

```
dwbs001sl(sys): @rpt_waits
Enter value for end_snap_time_filter: 09/17/18 14:07:19
Enter value for test_name:
```

TEST_NAME	TEST_TYPE	END_SNAP	STAT_CLASS	NAME	DELTA
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	ELAPSED	secs - elapsed time	33
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	User	secs - CPU used by this session	50.36
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Cache	MB/s - physical read total bytes	4617.07813
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	sql_id	5jy239nnxy08p	0
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	tables accessed	STAGE_EPWF_BATCH_MEMBERSHIP,STAGE_EPWF_BATCH_TRANSACTION,STAGE_EPWF_NDANKX_DPC,STAGE_EPWF_PAYMENT_GLT,STAGE_H_EPWF_PAYMENT_GLT_EXT	0
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Idle - PX Deq: Execute Reply	TIME_WAITED_MICRO	32092178
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Idle - SQL*Net message from client	TIME_WAITED_MICRO	369208
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Other - events in waitclass Other	TIME_WAITED_MICRO	55341
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	User I/O - cell smart table scan	TIME_WAITED_MICRO	48927
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	User I/O - Disk file operations I/O	TIME_WAITED_MICRO	3244
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	User I/O - external table write	TIME_WAITED_MICRO	2065
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Concurrency - row cache lock	TIME_WAITED_MICRO	1631
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Application - enq: KO - fast object checkpoint	TIME_WAITED_MICRO	512
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Cluster - gc current grant 2-way	TIME_WAITED_MICRO	449
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Idle - PX Deq: Join ACK	TIME_WAITED_MICRO	426
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	Concurrency - library cache lock	TIME_WAITED_MICRO	162
g-00030_sqld360_537514_9hgts04h5msvv_2a_	gluentperf	09/17/18 14:07:19	User I/O - direct path read	TIME_WAITED_MICRO	59

17 rows selected.

## rpt\_delete.sql

- This script deletes all entries for a particular run on the table GLUENT\_APP.GET\_RUN\_STATS
- The script doesn't automatically commit and it should be explicitly invoked after the delete for changes to take effect

### @rpt\_delete

Enter value for delete\_time: 09/17/18 14:07:19

898 rows deleted.

### @rpt\_waits

Enter value for end\_snap\_time\_filter: 09/17/18 14:07:19

Enter value for test\_name:

no rows selected

commit;

## Export to HTML and Create pivot table

### rpt\_report\_pivot.sql

- This script spools the benchmark run in HTML format
- The table can be easily copied to Excel for pivot table creation and reporting/tracking

### @rpt\_report\_pivot %

```
$ ls rpt_report_pivot*.html
rpt_report_pivot-dwbs001s1-%.html
```

TEST_NAME	TYPE	CATEGORY	NOTES	END_SNAP	ELAPSED	CPU_USED	SQLID	TABLE_NAME
00009_sqlid360_537514_7vc04zxpawshr_2a_3_	baselineperf	eispa	09-26	09/26/18 18:28:20	1095	99.31	9uf73jz1bwzpw	CPROD10T_HIST
00009_sqlid360_537514_18ac0dx5ta7ys_2a_3_	baselineperf	eispa	09-26	09/26/18 18:47:19	18	14.29	08k4kczrxgqp8	PRODR_PRD_ACCT_CMP
00009_sqlid360_537514_9qufc5vv3qh6n_2a_3_	baselineperf	eispa	09-26	09/26/18 19:06:19	1102	1498.15	7xgd0an0qx9wn	PRODR_PRD_ACCT_CMP
00009_sqlid360_537514_a83rkcaq3f8zg_2a_3_	baselineperf	eispa	09-26	09/26/18 20:12:20	6216	459.4	cy5yk6kt6hrjg	CPROD10T_HIST
g-00030_sqlid360_537514_8kf8b8u0dyz4u_2a_	gluentperf	cdims	09-05	09/05/18 16:04:12	2293	201.6	204rtujz54r8p	AT_VENDOR_DEMOGRAPHIC
				09/05/18 16:20:05	1715	226.75	975ms5ww5mwy3	AT_VENDOR_DEMOGRAPHIC

## Create pivot table

- Copy the data to excel and filter all columns -> select "TABLE NAME" -> uncheck Blanks -> OK
- Then copy the entire table again to a new sheet. Further filtering and formatting of data can be done.
- Once the table is clean the pivot table can be created.
- The following files could serve as reference on how the pivot table can be achieved and reported:
  - PerformanceTestingRuntime.xlsx
  - aeg-ctl-Gluent-BASECAMP-Completion-Report.pptx

- o aeg-ctl-Gluent-ALL-DBS-Completion-Report.pdf

## Initial formatting

PerformanceTestingRuntime.xlsx

File Home Insert Page Layout Formulas Data Review View Tell me what you want to do

Get Data From Text/CSV Recent Sources From Web Existing Connections Refresh All Edit Links

Queries & Connections Properties Edit Links

Sort & Filter Sort Filter Clear Reapply Advanced

J21

	A	B	C	D	E	F	G	H	I	J
	TEST_N AME	TYPE	CATEG ORY	NOTE	END_SNAP	ELAPSE D	CPU_US ED	SQLID	TABLE NAME	
21	00030_sqld360_537514_19k4c24b66w6q_2a_24	baselineperf	rma	run 1	8/5/2018 19:00					
23	00030_sqld360_537514_1ahh1u24cp36a8_2a_24	baselineperf	rma	run 1	8/5/2018 19:00					
24	00030_sqld360_537514_28agjzcxcp7072a_2a_24	baselineperf	rma	run 1	8/5/2018 19:00					
	00030_sqld360_537514_4a2b3a00xohf	baselineperf	rma	run 1	8/5/2018 19:00					

Filter by Color

Clear Filter From "TABLE\_NAME"

Filter by Color

Text Filters

Search

- ☒ FW\_LU\_QUAL
- ☒ LATIS\_NETWORKX\_TAX\_EVTNT\_DTL
- ☒ LATIS\_NME\_HDR
- ☒ LATIS\_NME\_TRANS\_DTL
- ☒ PRODR\_ORD\_ORDER
- ☒ PRODR\_PRD\_ACCT\_CMP
- ☒ RIB\_USA6
- ☒ RJF\_JOURNAL\_DETAIL
- ☒ TRAF\_TRAFFIC\_8XX\_DETAIL
- ☐ (Blanks)

OK Cancel

## Detailed data used for Pivot table

PerformanceTestingRuntime.xlsx - Excel

Arao, Kristofferson

File Home Insert Page Layout Formulas Data Review View Tell me what you want to do

H106 296.93

	B	C	D	E	F	G	H	I	J	K	L	M
	NAME	TYPE	CATEG ORY	NOTE	END_SNAP	ELAPSED	CPU_US ED	EXEC	ROW	SQLID	TABLE_NAME	inclu ed
90	00009_sqld360_537514_1zku8163a624t_2a_2	properf baseline	rma			1573.68		2			RJF_JOURNAL_DETAIL	y
91	00009_sqld360_537514_4hkh5b16y3w98_2a_2	properf baseline	rma			450		1			RJF_JOURNAL_DETAIL	y
93	00009_sqld360_537514_csk1ahy7ipn76_2a_3	baselineperf	reg1	26-Sep	9/26/2018 16:36	93	13.78			d9qg812dv56w	TRAF_TRAFFIC_8XX_DETAIL	y
94	00009_sqld360_537514_gdf1ya2scn6a_2a_3	baselineperf	reg1	26-Sep	9/26/2018 16:38	95	13.86			2v0jvu35lkpzq	TRAF_TRAFFIC_8XX_DETAIL	y
95	00009_sqld360_537514_csk1ahy7ipn76_2a_3	gluentperf	reg1	26-Sep	9/26/2018 15:25	51	7.37			csk1ahy7ipn76	TRAF_TRAFFIC_8XX_DETAIL	y
96	00009_sqld360_537514_gdf1ya2scn6a_2a_3	gluentperf	reg1	26-Sep	9/26/2018 15:27	60	6.78			g67bws4531ptn	TRAF_TRAFFIC_8XX_DETAIL	y
98	00030_sqld360_537514_4a2b3q00xqhd_2a_2	baselineperf	rma	26-Sep	9/26/2018 17:08	7	0.82			10bsywgghn6c8	LATIS_NETWORKX_TAX_EVTNT_DTL	y
99	00030_sqld360_537514_3dssupusdam5k_2a_2	baselineperf	rma	26-Sep	9/26/2018 17:09	5	0.65			db1d4d42k3q	LATIS_NETWORKX_TAX_EVTNT_DTL	y
100	00030_sqld360_537514_4a2b3q00xqhd_2a_2	gluentperf	rma	26-Sep	9/26/2018 17:02	2	0.54			a1mmgpm2h9b1	LATIS_NETWORKX_TAX_EVTNT_DTL	y
101	00030_sqld360_537514_3dssupusdam5k_2a_2	gluentperf	rma	26-Sep	9/26/2018 17:02	0.44	0.44			fvkr9by3sk62b	LATIS_NETWORKX_TAX_EVTNT_DTL	y
103	00030_sqld360_537514_5p7sgcz3056aw_2a_2	baselineperf	rma	26-Sep	9/26/2018 17:30	246	343.59			694zvfwn74qf	LATIS_NME_HDR,LATIS_NME_TRANS_DTL	y
104	00030_sqld360_537514_732jnvwrfsmh6_2a_2	baselineperf	rma	26-Sep	9/26/2018 17:34	208	311.77			0nmzhrn06b8z6g	LATIS_NME_HDR,LATIS_NME_TRANS_DTL	y
105	00030_sqld360_537514_5p7sgcz3056aw_2a_2	gluentperf	rma	26-Sep	9/26/2018 17:24	448	2749.56			482t5trk3kq	LATIS_NME_HDR,LATIS_NME_TRANS_DTL	n
106	00030_sqld360_537514_5p7sgcz3056aw_2a_2	gluentperf	rma	26-Sep	9/26/2018 17:24	278	296.93				LATIS_NME_HDR,LATIS_NME_TRANS_DTL	y
107	00030_sqld360_537514_732jnvwrfsmh6_2a_2	gluentperf	rma	26-Sep	9/26/2018 17:33	480	2725.53			1f2wqps38swcj	LATIS_NME_HDR,LATIS_NME_TRANS_DTL	n
108	00030_sqld360_537514_732jnvwrfsmh6_2a_2	gluentperf	rma	26-Sep	9/26/2018 17:33	257	262.36				LATIS_NME_HDR,LATIS_NME_TRANS_DTL	y
110	00009_sqld360_537514_7vc04zxpawshr_2a_2	baselineperf	eispa	26-Sep	9/26/2018 18:28	1095	93.31			9uf73jz1bwzpw	CPRD10T_HIST	y
111	00009_sqld360_537514_a83kcaq3f8zg_2a_3	baselineperf	eispa	26-Sep	9/26/2018 20:12	6216	459.4			cy5yk6kt6hvjg	CPRD10T_HIST	y
112	00009_sqld360_537514_7vc04zxpawshr_2a_2	gluentperf	eispa	26-Sep	9/26/2018 18:25	913	115.15			7vc04zxpawshr	CPRD10T_HIST	y
113	00009_sqld360_537514_a83kcaq3f8zg_2a_3	gluentperf	eispa	26-Sep	9/26/2018 20:32	7638	1017			6m1qcvg5dg7gc	CPRD10T_HIST	y
115	00009_sqld360_537514_anvn862vdd86b_2a_2	baselineperf	eispa	26-Sep	9/26/2018 18:13	1	0.19			83y4gqmwhkq8w	PRODR_ORD_ORDER	y
116	00009_sqld360_537514_anvn862vdd86b_2a_2	gluentperf	eispa	26-Sep	9/26/2018 18:12	2	0.48			anvn862vdd86b	PRODR_ORD_ORDER	y
118	00009_sqld360_537514_18ac0d5ta7ys_2a_3	baselineperf	eispa	26-Sep	9/26/2018 18:47	18	14.29			08k4kcznngap8	PRODR_PRD_ACCT_CMP	y
119	00009_sqld360_537514_3qufc5vv3qh6n_2a_2	baselineperf	eispa	26-Sep	9/26/2018 19:06	1102	1498.15			7xgd0an0qx3vn	PRODR_PRD_ACCT_CMP	y
120	00009_sqld360_537514_18ac0d5ta7ys_2a_3	gluentperf	eispa	26-Sep	9/26/2018 18:41	19	11.14			cvnn87d3c6j3g	PRODR_PRD_ACCT_CMP	y
121	00009_sqld360_537514_3qufc5vv3qh6n_2a_2	gluentperf	eispa	26-Sep	9/26/2018 19:06	1432	2340.28			3qufc5vv3qh6n	PRODR_PRD_ACCT_CMP	y
122												
123												
124												

Sheet2 DWTST2 DWTST Elbow-Query-Testing Summary-Test dwtst-legend QGEMTST DWBSTST DWOPTST CCDWTST DWDNTST

## Pivot table for performance tracking

PerformanceTestingRuntime.xlsx - Excel													PivotTable Tools		Arao, Kristofferson									
File Home Insert Page Layout Formulas Data Review View Analyze Design Tell me what you want to do																								
A30													fx		EPWF_PAYMENT									
1	include																							
2																								
3																								
4																								
5																								
6																								
7																								
8																								
9																								
10																								
11																								
12																								
13																								
14																								
15																								
16																								
17																								
18																								
19																								
20																								
21																								
22																								
23																								
24																								
25																								
26																								
27																								
28																								
29																								
30																								
31																								
32																								
33																								
34																								
35																								
36																								
37																								
38																								
39																								
40																								
41																								
42																								
43																								
44																								
45																								
46																								
47																								
48																								

## Summary data used for final report

PerformanceTestingRuntime.xlsx - Excel										Chart Tools		Arao, Kristofferson	
File Home Insert Page Layout Formulas Data Review View Design Format Tell me what you want to do													
Chart 7													
	B	C	D	E	F	G	H	I	J	K	L		
		Query#	App query	Baseline Avg Elapsed	Gluent Avg Elapsed	Baseline Average of CPU_USED	Gluent Average of CPU_USED	Table	SQL Identifier	App Team			
2	Application												
3	Basecamp	1	cdims-1	1587.00	1078.00	358.47	65.56	AT_VENDOR_DEMOGRAPHIC	00030_sqld360_537514_8kf8b8u0dyz4u_2a_	cdims			
4	Basecamp	2	eispa-2	1095.00	919.00	115.15	99.31	CPROD10T_HIST	00009_sqld360_537514_7vc04zxpawshr_2a_3_	eispa			
5	Basecamp	3	eispa-3	6216.00	7638.00	1017.00	459.40	CPROD10T_HIST	00009_sqld360_537514_a83rkcaq3f8zg_2a_3_	eispa			
6	Basecamp	4	eispa-4	1.00	2.00	0.48	0.19	PRODR_ORD_ORDER	00009_sqld360_537514_anvn862vdd86b_2a_3_	eispa			
7	Basecamp	5	eispa-5	18.00	19.00	11.14	14.29	PRODR_PRD_ACCT_CMP	00009_sqld360_537514_18ac0dx5ta7ys_2a_3_	eispa			
8	Basecamp	6	eispa-6	1102.00	1492.00	2340.28	1498.15	PRODR_PRD_ACCT_CMP	00009_sqld360_537514_9qufc5vv3qh6n_2a_3_	eispa			
9	Basecamp	7	ni-7	474.00	197.00	190.48	30.62	ASSIA_PORTS	00009_sqld360_537514_b7z31j3yctzms_2a_3_	ni			
10	Basecamp	8	ni-8	6.00	2.00	0.32	0.65	ASSIA_PORTS	00009_sqld360_537514_gs5dbr0xs0ny0_2a_	ni			
11	Basecamp	9	reg1-9	93.00	51.00	7.37	13.78	TRAF_TRAFFIC_8XX_DETAIL	00009_sqld360_537514_csk1ahy7jpn76_2a_3_	reg1			
12	Basecamp	10	reg1-10	95.00	60.00	6.78	13.86	TRAF_TRAFFIC_8XX_DETAIL	00009_sqld360_537514_gdfyya2scry6a_2a_3_	reg1			
13	Basecamp	11	rna-11	42.00	838.00	7469.75	59.99	ENS_REVENUE_DETL					
14	Basecamp	12	rna-12	11.00	149.00	1558.69	33.60	ENS_REVENUE_DETL					
15	Basecamp	13	rna-13	2.00	44.00	53.43	0.37	EPWF_PAYMENT					
16	Basecamp	14	rna-14	1981.00	318.00	233.92	104.41	EPWF_PAYMENT					
17	Basecamp	15	rna-15	90.00	137.00	114.59	37.60	EPWF_PAYMENT					
18	Basecamp	16	rna-16	7.00	2.00	0.54	0.82	LATIS_NETWORK_TAX_EVTNT					
19	Basecamp	17	rna-17	5.00	0.44	0.44	0.65	LATIS_NETWORK_TAX_EVTNT					
20	Basecamp	18	rna-18	145.00	377.00	279.43	62.90	LATIS_NME_HDR					
21	Basecamp	19	rna-19	11.50	85.00	179.70	2.11	LATIS_NME_HDR					
22	Basecamp	20	rna-20	246.00	276.00	296.93	349.59	LATIS_NME_HDR,LATIS_NME_HDR					
23	Basecamp	21	rna-21	208.00	257.00	262.36	311.77	LATIS_NME_HDR,LATIS_NME_HDR					
24	Basecamp	22	rna-22	47.00	1289.00	5046.55	19.95	RJF_JOURNAL_DETAIL					
25	Basecamp	23	rna-23	47.00	232.00	799.09	18.87	RJF_JOURNAL_DETAIL					
26	Basecamp	24	rna-24	42.00	64.00	8.00	15.77	RJF_JOURNAL_DETAIL					

Basecamp

Query	Baseline Avg Elapsed	Gluent Avg Elapsed
cdims-1	1587.00	1078.00
eispa-2	1095.00	919.00
eispa-3	6216.00	7638.00
eispa-4	1.00	2.00
eispa-5	18.00	19.00
eispa-6	1102.00	1492.00
ni-7	474.00	197.00
ni-8	6.00	2.00
reg1-9	93.00	51.00
reg1-10	95.00	60.00
rna-11	42.00	838.00
rna-12	11.00	149.00
rna-13	2.00	44.00
rna-14	1981.00	318.00
rna-15	90.00	137.00
rna-16	7.00	2.00
rna-17	5.00	0.44
rna-18	145.00	377.00
rna-19	11.50	85.00
rna-20	246.00	276.00
rna-21	208.00	257.00
rna-22	47.00	1289.00
rna-23	47.00	232.00
rna-24	42.00	64.00

Collection-TablesByAppTeam DWTST2 DWTST Elbow-Query-Testing Summary-Test dwtst-legend QGEMTST DWBSTST DWOPTST ...



- Also the tab “Collection-TablesByAppTeam” of PerformanceTestingRuntime.xlsx Excel sheet would also serve as a project tracker that shows the status for each table

	A	B	C	D	E	F	G	H	I	J	K
1	Prod DB Name	app team	Owner	Table Name	include (Y/N)	refreshed_date	notes	baseline	offloaded	gluentperf	Status (pending/collected/error/ baselineperf/gluentperf)
28	DVBS001P	REG1	STAGE	TRAF_TRAFFIC_800_DETAIL	Y	9/24/2018		9/26/2018	9/24/2018	9/26/2018	gluentperf
29	DVBS001P	CDIMS	STAGE	AT_VENDOR_DEMOGRAPHIC	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
30	DVBS001P	CDIMS	STAGE	FV_LU_QUAL	N	8/31/2018	156632308 rows	8/31/2018	9/12/2018		baselineperf
31	DVBS001P	RNA	STAGE	ENS_REVENUE_DET	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
32	DVBS001P	RNA	STAGE	EPWF_PAYMENT	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
33	DVBS001P	RNA	STAGE	LATIS_NETWORK_TAX_EVENT_DTL	Y	9/4/2018	2103787016 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
34	DVBS001P	RNA	STAGE	LATIS_NME_HDR	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
35	DVBS001P	RNA	STAGE	LATIS_NME_TRANS_DTL	Y	9/4/2018	107135567 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
36	DVBS001P	RNA	STAGE	RIB_USA6	N						baselineperf
37	DVBS001P	RNA	STAGE	RIF_BOOKED_REVENUE	N						error
38	DVBS001P	RNA	STAGE	RIF_JOURNAL_DETAIL	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
39	DVBS001P	EISPA	STAGE	CDV_CPRODWI_DLY	N						baselineperf
40	DVBS001P	EISPA	STAGE	CPRDWI_HIST	Y	9/4/2018	158624103 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
41	DVBS001P	EISPA	STAGE	PRODR_PRD_ORDER	Y	9/4/2018	17543 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
42	DVBS001P	EISPA	STAGE	PRODR_PRD_ACCT_CMP	Y	9/4/2018	259034648 rows	9/26/2018	9/12/2018	9/26/2018	gluentperf
43	DVBS001P	NI	STAGE	ASSIA_PORTS	Y	8/31/2018		8/31/2018	9/5/2018	9/5/2018	gluentperf
44	DVBS001P	PSO	STAGE	SFDC_ACCOUNT	N						pending

## prodperf – getting the PROD run times

- Aside from having baselineperf and gluentperf elapsed times, it is also best to have the production elapsed times in order to assess how comparable the Performance Testing vs the production scale data volume or run time. This will give us a ballpark how will the gluentperf will perform when migrated to production.
- The output of ash\_elap\_hist\_sqlid.sql file can be manually put in the pivot table after the baselineperf and gluentperf are executed. The data produced by this script is divided into two parts:
  - Time series recent elapsed times
  - Summarized statistics by plan hash value (number of executions and avg,min,max elapsed times)

@ash\_elap\_hist\_sqlid

DBA\_HIST\_ACTIVE\_SESS\_HISTORY - ash\_elap by exec (recent)

~~~~~

Enter value for sql\_id: 0z9ru49mxu230

Example output (0z9ru49mxu230):

| DBNAME                                                                       | I#          | INST                | HOST_NAME                 | CREATED            | LOG_MODE                 | GLOBAL_NAME  |
|------------------------------------------------------------------------------|-------------|---------------------|---------------------------|--------------------|--------------------------|--------------|
| DWBS001P                                                                     | 4           | DWBS001P4           | pddccdbadm04              | 06/24/14 21:37     | ARCHIVELOG               | DWBS001P     |
| Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production |             |                     |                           |                    |                          |              |
| dwbs001p4(ac26646): @ash_elap_hist_sqlid                                     |             |                     |                           |                    |                          |              |
| DBA_HIST_ACTIVE_SESS_HISTORY - ash_elap by exec (recent)                     |             |                     |                           |                    |                          |              |
| ~~~~~                                                                        |             |                     |                           |                    |                          |              |
| Enter value for sql_id: 0z9ru49mxu230                                        |             |                     |                           |                    |                          |              |
| SQL_ID                                                                       | SQL_EXEC_ID | SQL_PLAN_HASH_VALUE | SQL_EXEC_START            | RUN_TIME_TIMESTAMP |                          | RUN_TIME_SEC |
| 0z9ru49mxu230                                                                | 83886717    | 286530956           | 03-NOV-18 03.30.39.000000 | AM                 | +0000000000 00:00:19.517 | 19.517       |
| 0z9ru49mxu230                                                                | 67109792    | 286530956           | 03-NOV-18 04.19.23.000000 | AM                 | +0000000000 00:00:02.348 | 2.348        |
| 0z9ru49mxu230                                                                | 67109793    | 286530956           | 03-NOV-18 05.13.49.000000 | AM                 | +0000000000 00:00:06.544 | 6.544        |
| 0z9ru49mxu230                                                                | 83886718    | 286530956           | 03-NOV-18 05.53.40.000000 | AM                 | +0000000000 00:00:16.630 | 16.63        |
| 0z9ru49mxu230                                                                | 83886719    | 286530956           | 03-NOV-18 06.35.42.000000 | AM                 | +0000000000 00:00:07.344 | 7.344        |
| 0z9ru49mxu230                                                                | 67109794    | 286530956           | 03-NOV-18 07.12.33.000000 | AM                 | +0000000000 00:00:01.164 | 1.164        |
| 0z9ru49mxu230                                                                | 83886720    | 286530956           | 03-NOV-18 07.50.02.000000 | AM                 | +0000000000 00:00:10.918 | 10.918       |
| 0z9ru49mxu230                                                                | 83886721    | 286530956           | 03-NOV-18 08.30.58.000000 | AM                 | +0000000000 00:00:06.869 | 6.869        |
| 0z9ru49mxu230                                                                | 83886722    | 286530956           | 03-NOV-18 09.09.38.000000 | AM                 | +0000000000 00:00:09.042 | 9.042        |
| 0z9ru49mxu230                                                                | 83886723    | 286530956           | 03-NOV-18 09.46.35.000000 | AM                 | +0000000000 00:00:03.984 | 3.984        |
| 0z9ru49mxu230                                                                | 83886724    | 286530956           | 03-NOV-18 10.24.02.000000 | AM                 | +0000000000 00:00:09.036 | 9.036        |

... output snipped ...



```

0z9ru49mxu230      83886128      286530956 17-DEC-18 08.47.00.000000 PM +000000000 00:00:09.411      9.411
0z9ru49mxu230      83886129      286530956 17-DEC-18 09.32.57.000000 PM +000000000 00:00:10.121      10.121

999 rows selected.

DBA_HIST_ACTIVE_SESS_HISTORY - ash_elap exec avg min max
-----
SQL_PLAN_HASH_VALUE  COUNT(*)  AVG      MIN      MAX
-----
286530956           1517      10.5      .26      54.85
                  1517      10.5      .26      54.85

2 rows selected.

```

## Get Impala memory usage

There would be a case that a SQL would consume a lot of memory on the Hadoop side and as a result errors with “Memory limit exceeded”. The Hadoop admin can either increase the memory limit or the SQL can be tuned to efficiently use memory.

```

@g-d8vv2w24abj7t.sql

PL/SQL procedure successfully completed.

select /*+ monitor */
*
ERROR at line 1:
ORA-12801: error signaled in parallel query server P001, instance podclodmdb03.corp.intranet:dwtst3 (3)
ORA-29913: error in executing callout
ORA-29400: data cartridge error
KUP-04095: preprocessor command /u01/app/gluent/DWTST/offload/bin/smart_connector.sh encountered error
"checkStatus(): FetchResults
has runtime error (0): Memory limit exceeded: FunctionContextImpl::AllocateLocal's allocations exceeded memory
limits.
PLAN_ROOT_SINK Exprs could not allocate 6.00 B without exceeding limit.
Error occurred on backend polccdhdn006.test.intranet:22001 by fragment e247d997970bdf05:dad41f1c00000000
Memory left in process limit: -212992.00 B
Query(e247d997970bdf05:dad41f1c00000000): Total=925.67 MB Peak=925.67 MB
Fragment e247d997970bdf05:dad41f1c00000000: Total=59.44 MB Peak=59.76 MB
EXCHANGE_NODE (id=1): Total=0 Peak=0
DataStreamRecv: Total=48.68 MB Peak=48.69 MB
PLAN_ROOT_SINK: Total=7.00 MB Peak=7.00 MB
PLAN_ROOT_SINK Exprs: Total=7.00 MB Peak=7.00 MB
CodeGen: Total=13.37 KB Peak=566.50 KB
Block Manager: Limit=8.00 GB Total=0 Peak=0
Fragment e247d997970bdf05:dad41f1c00000005: Total

```

To proactively find the SQLs that consume high memory in Hadoop. The SQL Performance Test files can be mined and correlated with Impala profile data.

Follow the steps below:

- 1) Go to the impala\_memory folder  

```
cd sql/impala_memory
```
- 2) Execute the impala\_memory.sh file  

```
./impala_memory.sh
```
- 3) Review the file impala\_memory\_output.txt
- 4) On the benchmark table query the file using the filter "test name". Use @rpt\_elap  
you'll see the baselineperf and gluentperf versions of the file.

## Example output (d8vv2w24abj7t):

```
cat impala_memory_output.txt
--
./DWP/PROD/baseline2-GluentPerf/between_perf-GluentHybrid-d8vv2w24abj7t-issue/dir_00030_sqld360_766573_5y9qdn7d38wq5_2a_3_standalone_sql.sql-issue-afterPX/sqlmon_5y9qdn7d38wq5_50331649_20180724_165
00:SCAN HDFS 6 444.856ms 789.064ms 26.79M 2.68M 1.22 GB 1.89 GB g_ctledw_dim.delta_charge_p...
./DWP/PROD/baseline2-GluentPerf/between_perf-GluentHybrid-d8vv2w24abj7t-issue/dir_00030_sqld360_914483_d8vv2w24abj7t_2a_24_standalone_sql.sql-issue-beforePX/sqlmon_5y9qdn7d38wq5_50331649_20180720_1
00:SCAN HDFS 12 25426ms 25616ms 428.72M 42.87M 1.88 GB 1.89 GB g_ctledw_dim.delta_charge_p...
--
./DWP/PROD/baseline2-GluentPerf/dir_00030_sqld360_914483_d8vv2w24abj7t_2a_24_standalone_sql.sql-issue/sqlmon_5y9qdn7d38wq5_50331649_20180720_154924.html
00:SCAN HDFS 12 25426ms 25616ms 428.72M 42.87M 1.88 GB 1.89 GB g_ctledw_dim.delta_charge_p...
```

```
dwtst3(sys): @rpt_elap
Enter value for test_name: %d8vv2w24abj7t%

TEST_NAME                                TYPE                CATEGORY  NOTES                                END_SNAP              ELAPSED    CPU_USED  SQLID              TABLES_ACCESSED
-----
00030_sqld360_914483_d8vv2w24abj7t_2a_24 baselineperf        07/19/18 16:17:12      18         66.45 bpnf93jppyn2t    DIM.DELTA_BILL_PULL_TBL_GLT,DIM.DELTA_CHARGE_PULL_TBL_GLT,DIM.DELTA_INVOICE_ITEM_PULL_GLT,DIM.DELTA_INVOICE_ITEM_PULL_GLT,DIM.ENS_BAN_LEVEL_NPANOX_DT,DIM.ENS_PRODUCT_LEVEL_NPANOX_DT,REF.ENS_BILL_FEATURE_T,REF.ENS_FTR_COMBO_DT_GLT,REF.ENS_GEO_GEOGRAPHY_T,STAGE.GL_DETAIL_LOAD_TEMP,STAGE.GL_PLUS_JURISDICTION_XREF

00030_sqld360_914483_d8vv2w24abj7t_2a_24 baselineperf        07/23/18 17:06:54      12         64.57 f48956257a98h    DIM.DELTA_BILL_PULL_TBL_GLT,DIM.DELTA_CHARGE_PULL_TBL_GLT,DIM.DELTA_INVOICE_ITEM_PULL_GLT,DIM.ENS_BAN_LEVEL_NPANOX_DT,DIM.ENS_PRODUCT_LEVEL_NPANOX_DT,LUENT_APP.GET_RUN_STATS,REF.ENS_BILL_FEATURE_T,REF.ENS_FTR_COMBO_DT_GLT,REF.ENS_GEO_GEOGRAPHY_T,STAGE.GL_DETAIL_LOAD_TEMP,STAGE.GL_PLUS_JURISDICTION_XREF

00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54     5310       5037.1 5y9qdn7d38wq5    DIM.DELTA_BILL_PULL_TBL_GLT,DIM.DELTA_CHARGE_PULL_TBL_GLT,DIM.DELTA_INVOICE_ITEM_PULL_GLT,DIM.ENS_BAN_LEVEL_NPANOX_DT,DIM.ENS_PRODUCT_LEVEL_NPANOX_DT,DIM.H.DELTA_BILL_PULL_TBL_GLT_EXT,DIM.H.DELTA_INVOICE_ITEM_PULL_GLT_EXT,DIM.H.DELTA_INVOICE_ITEM_PULL_G_0KQV,REF.ENS_BILL_FEATURE_T,REF.ENS_FTR_COMBO_DT_GLT,REF.ENS_GEO_GEOGRAPHY_T,STAGE.GL_DETAIL_LOAD_TEMP,STAGE.GL_PLUS_JURISDICTION_XREF,SYS.DIR$,SYS.OBJ$

3 rows selected.
```

```
dwtst3(sys): @rpt_waits
Enter value for end_snap_time_filter: 07/20/18 17:17:54
Enter value for test_name:

TEST_NAME                                TYPE                END_SNAP              STAT_CLASS              NAME                                DELTA
-----
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 ELAPSED                 secs - elapsed time                                5310
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 User                   secs - CPU used by this session                    5037.1
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Cache                  MB/s - physical read total bytes                    245032.922
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 sql_id                 5y9qdn7d38wq5                                       0
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 tables accessed        DIM.DELTA_BILL_PULL_TBL_GLT,DIM.DELTA_CHARGE_PULL_TBL_GLT,DIM.DELTA_INVOICE_ITEM_PULL_GLT,DIM.ENS_BAN_LEVEL_NPANOX_DT,DIM.ENS_PRODUCT_LEVEL_NPANOX_DT,DIM.H.DELTA_BILL_PULL_TBL_GLT_EXT,DIM.H.DELTA_INVOICE_ITEM_PULL_GLT_EXT,DIM.H.DELTA_INVOICE_ITEM_PULL_G_0KQV,REF.ENS_BILL_FEATURE_T,REF.ENS_FTR_COMBO_DT_GLT,REF.ENS_GEO_GEOGRAPHY_T,STAGE.GL_DETAIL_LOAD_TEMP,STAGE.GL_PLUS_JURISDICTION_XREF,SYS.DIR$,SYS.OBJ$

00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 User I/O - external table read                    TIME_WAITED_MICRO                                2577974073
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Idle - PX Deq: Execute Reply                      TIME_WAITED_MICRO                                257832885
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Idle - PX Deq Credit: need buffer                  TIME_WAITED_MICRO                                6238471
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Idle - PX Deq Credit: send blk                     TIME_WAITED_MICRO                                5633732
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Idle - SQL*Net message from client                 TIME_WAITED_MICRO                                373446
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Other - events in waitclass Other                  TIME_WAITED_MICRO                                135968
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 User I/O - external table open                     TIME_WAITED_MICRO                                103975
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 User I/O - cell smart table scan                   TIME_WAITED_MICRO                                34864
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 System I/O - control file sequential read          TIME_WAITED_MICRO                                17603
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 User I/O - cell list of blocks physical read        TIME_WAITED_MICRO                                4227
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Concurrency - library cache pin                    TIME_WAITED_MICRO                                1218
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 User I/O - cell multiblock physical read            TIME_WAITED_MICRO                                1150
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 User I/O - direct path read                        TIME_WAITED_MICRO                                1017
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Cluster - gc cr grant 2-way                        TIME_WAITED_MICRO                                388
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Application - enq: KO - fast object checkpoint    TIME_WAITED_MICRO                                228

00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Cluster - gc current grant busy                    TIME_WAITED_MICRO                                203
00030_sqld360_914483_d8vv2w24abj7t_2a_24 gluentperf           07/20/18 17:17:54 Cluster - gc cr multi block request                TIME_WAITED_MICRO                                165

22 rows selected.
```

## Packaging SQL troubleshooting files to Gluent Support

In the process of benchmarking and running SQLs we would encounter performance issues where baselineperf or gluentperf would run longer than expected or the query would error and terminate. These slow performance patterns or issues can be addressed faster if comprehensive troubleshooting data is collected.

Under the sql/SQLD360 folder is a standardized performance data collection tool that is based on a SQL\_ID which outputs all needed troubleshooting data. The following are collected:

- Gluent SQL monitor (based on the SQL\_ID)
- Oracle SQL monitor reports (all kinds of SQL monitor report type)
- metadata info (metadata info around the SQL\_ID)
- standalone SQL (the SQL with binds that is used for Performance Testing)
- planx.sql (contains ASH, response time info, table, index and column statistics, AWR\_PLAN\_CHANGE, etc.)
- snapper.sql (based on the SQL\_ID)

All collected data are placed in folder dir\_<SQL\_ID> and can be easily zipped to share with the Gluent team.

To collect and package the troubleshooting files. Follow the steps below:

1) Go to sql/SQLD360 directory. Run the sqld360 file and pass the SQL\_ID

```
cd sql/SQLD360
@sqld360 1ahhu24cp36a8
```

2) Zip the output directory

```
ls -ld dir*
dir_1ahhu24cp36a8

zip -r dir_1ahhu24cp36a8 dir_1ahhu24cp36a8

ls -l dir*
dir_1ahhu24cp36a8.zip
```

## Appendices

### db\_app\_table.txt

```
/db_backup_denx3/p1/gluent/get_run_stats/get_run_stats2-master > cat db_app_table.txt
DWPROD,CTLEDW,DAILY_DISCOUNT_QTY_T
DWPROD,CTLEDW,DELTA_BILL_PULL_TBL
DWPROD,CTLEDW,DELTA_CHARGE_PULL_TBL
DWPROD,CTLEDW,DELTA_CUSTOMER_EXT_TBL
DWPROD,CTLEDW,DELTA_INVOICE_ITEM_PULL_TBL
DWPROD,CTLEDW,ENS_BILL_GL_DETAIL_T
DWPROD,CTLEDW,ENS_CSM_SALES_DT
DWPROD,CTLEDW,ENS_CSM_SUMMARY_DT
DWPROD,CTLEDW,OMS_FINAL_FEATURES_T
DWPROD,CTLEDW,OMS_MEMO_T
DWPROD,CTLEDW,P3_CUST_PROFILE_DT
DWPROD,CTLEDW,SERVICE_AGREEMENT
DWPROD,CTLEDW,SERVICE_FEATURE
QGEM001P,REG1,NWX_PERF_FPM_DATA
QGEM001P,REG1,NWX_PFM_E2E_DAY_SUMM
QGEM001P,REG1,NWX_PFM_PS_HOUR_SUMM
QGEM001P,REG1,NWX_PFM_SLA_KD
QGEM001P,REG1,NWX_PFM_SLA_KE
QGEM001P,REG1,FDM_BUS_DETAIL
QGEM001P,REG1,MASTER_CSO_MONTHLY
QGEM001P,REG1,MASTER_NWX_MONTHLY
QGEM001P,REG1,SUMMARY_PUC_MONTHLY
DWBS001P,REG1,TRAF_TRAFFIC_8XX_DETAIL
DWBS001P,CDIMS,AT_VENDOR_DEMOGRAPHIC
DWBS001P,CDIMS,FW_LU_QUAL
DWBS001P,RNA,ENS_REVENUE_DETL
DWBS001P,RNA,EPWF_PAYMENT
```

DWBS001P,RNA,LATIS\_NETWORKX\_TAX\_EVENT\_DTL  
 DWBS001P,RNA,LATIS\_NME\_HDR  
 DWBS001P,RNA,LATIS\_NME\_TRANS\_DTL  
 DWBS001P,RNA,RIB\_USA6  
 DWBS001P,RNA,RJF\_BOOKED\_REVENUE  
 DWBS001P,RNA,RJF\_JOURNAL\_DETAIL  
 DWBS001P,EISPA,CDW\_CPROD10T\_DLY  
 DWBS001P,EISPA,CPROD10T\_HIST  
 DWBS001P,EISPA,PRODR\_ORD\_ORDER  
 DWBS001P,EISPA,PRODR\_PRD\_ACCT\_CME  
 DWBS001P,NI,ASSIA\_PORTS  
 CDW001P,RNA,JRNL\_DETL\_FACT  
 CDW001P,RNA,JRNL\_DETL\_FACT\_ARC  
 CDW001P,EISPA,EIS\_DAILY\_SMRY  
 CDW001P,EISPA,EIS\_MNTH\_SMRY  
 CDW001P,EISPA,END\_IN\_SRVC\_FACT  
 CDW001P,EISPA,PROD\_ACTY\_FACT  
 DWDN001P,REG1,PRE\_BILL\_AUDIT  
 DWDN001P,REG1,SEMI\_ANNUAL  
 DWDN001P,SDSA,DOC\_ACTION\_DISPOSITION  
 DWDN001P,SDSA,DOC\_CUSTOMER\_SESSION  
 DWDN001P,SDSA,DOC\_EXECUTION\_NODE  
 DWDN001P,SDSA,DOC\_RXPS\_TRANS\_LOG  
 DWDN001P,SDSA,EXTERNAL\_DATA\_PULL  
 DWDN001P,SDSA,TST\_TEST\_RESULT\_CACHE\_V1  
 DWDN001P,MARGIN,PCM\_DRIVER\_FACT  
 DWDN001P,MARGIN,PCM\_DRIVER\_FACT\_ARC\_P  
 DWDN001P,MARGIN,PCM\_DRIVER\_FACT\_ARC\_R  
 DWDN001P,MARGIN,PCM\_SPECIFIC\_USE\_DETAIL  
 DWDN001P,MARGIN,PCM\_SPECIFIC\_USE\_DETAIL\_ARC\_R  
 DWDN001P,MARGIN,PCM\_SPECIFIC\_USE\_DETAIL\_P\_2016  
 DWDN001P,MARGIN,PCM\_SPECIFIC\_USE\_DETAIL\_P\_2017  
 DWDN001P,MARGIN,PCM\_SPECIFIC\_USE\_DETAIL\_R\_2015  
 DWDN001P,MARGIN,PCM\_SPECIFIC\_USE\_DETAIL\_R\_2016  
 DWDN001P,MARGIN,PCM\_SPECIFIC\_USE\_DETAIL\_R\_2017  
 DWDN001P,CDIMS,DQ\_ADDR\_CLEANSER  
 DWOPS01P,SDSA,CALCTR\_KPI\_LOG\_FACT  
 DWOPS01P,SDSA,CCD\_IDR\_IVR\_FACT\_T  
 DWOPS01P,SDSA,CCD\_IDR\_IVR\_FACT\_T  
 DWOPS01P,SDSA,DRIVR\_KPI\_LOG  
 DWOPS01P,SDSA,E2E\_CUSTOMER\_EVENT\_T  
 DWOPS01P,SDSA,E2E\_CUSTOMER\_PROFILE\_T  
 DWOPS01P,SDSA,EP\_VPAPPLOG  
 DWOPS01P,CDIMS,ORDACT\_PROD\_T  
 DWOPS01P,RNA,CDR\_T  
 DWOPS01P,EISPA,RSORLN\_T  
 DWOPS01P,EISPA,RSORSO\_T  
 DWOPS01P,EISPA,RSORSU\_STATUS\_T  
 DWOPS01P,EISPA,RSORSU\_T  
 DWOPS01P,EISPA,SOP\_FIDS\_T  
 DWOPS01P,EISPA,VISP\_SOP\_FACILITIES\_FIDS\_T  
 DWOPS01P,NI,ASSIA\_PE  
 DWOPS01P,NI,FW\_PRJ\_WBS\_ELEMENT\_T  
 DWOPS01P,NI,PERF\_FPM\_DATA\_T  
 DWOPS01P,NI,PFMODS\_E2E\_DAY\_SUMM\_T  
 DWOPS01P,NI,PFMODS\_E2E\_HOUR\_SUMM\_T  
 DWOPS01P,NI,PFMODS\_PS\_HOUR\_SUMM\_T  
 DWOPS01P,NI,SAP\_OHZMM\_OH01\_T  
 DWOPS01P,NI,SAP\_OHZPUR\_OH01\_T  
 DWOPS01P,NI,SAP\_OHZPUR\_OH02\_T  
 DWOPS01P,NI,USA\_TP\_SALES\_ACTIVITY

/db\_backup\_denx3/p1/glueut/get\_run\_stats/get\_run\_stats2-master > cat db\_app\_table.txt | wc -l  
 88