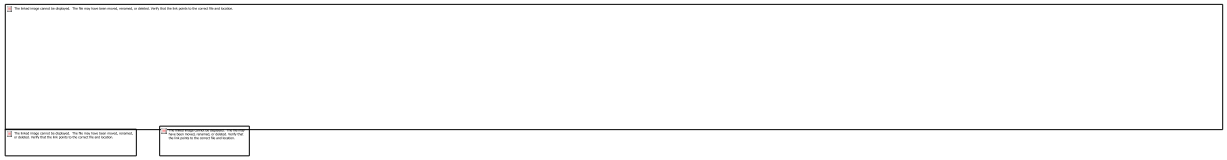

ORACLE Platinum Partner

The Oracle SQLT Utility

By Kevin Gilpin, Rolta TUSC

Wednesday March 14, 2012

Background

Background

- The **SQL Tuning Advisor** is one of several advisors that is available if the SQL Tuning Pack is licensed. The advisor performs in-depth analysis of SQL and derives pertinent configuration findings and performance recommendations that can be implemented with provided syntax, usually in the form of a SQL profile.
- The **Trace Analyzer** (TRCA) utility is closely related to the SQLT utility and is installed as part of the SQLT installation. This utility can also be downloaded, installed and used separately from SQLT. This utility creates an HTML report describing the pertinent information in a SQL trace file produced with the 10046 event.
- **SQL Plan Management** is a framework that Oracle Database uses to prevent performance regression of SQL statements as the data evolves and other conditions change such as index changes and database patches and upgrades are installed.
- A **SQL Plan Baseline** is a set of accepted SQL execution plans that have been used at various times for a given SQL statement.

Background

- A **SQL profile** is a set of performance data and corrections to execution plans that have been derived by the Oracle optimizer. SQL profiles are derived by the executing SQL Tuning Advisor to perform an in-depth analysis of the objects that a SQL statement uses and of the statement construct.
- A **SQL plan hash value** is an identifier for an execution plan that the Oracle optimizer has derived for a given SQL statement. This identifier is used by various features, such as SQL Baselines and SQL Plan Management to manage plans and metadata related to those plans.
- All of these features are used directly or indirectly by SQLT.

- Is it a coincidence that the SQLT version and the TRCA versions are the same (11.4.4.2) and were both released on 2/12/12? No.

What is the SQLT Utility?

What is the SQLT Utility?

- SQLT is a **free utility** developed by the Oracle Server Technologies Center of Expertise (CoE). It is downloadable from My Oracle Support Note 215187.1.
- It consists of some database schema objects and some associated SQL scripts and also includes the CoE Trace Analyzer utility.
- The PL/SQL code is unwrapped, allowing DBA's to learn about how the utility works and how Oracle manages execution plans, SQL Profiles and related items.
- The utility is **used to troubleshoot SQL performance** as a supplement to the Oracle SQL Tuning Advisor (STA) and provides a superset of the information produced by the STA.
- Studying how the SQLT and TRCA utilities work leads to an appreciation of **how far Oracle has come in providing automatic SQL tuning functionality** and of what the database is capable of doing to manage SQL performance. **SQL baselines, profiles and plan management** have become impressively **robust** in Oracle RDBMS 11.2 and it is worth the effort to master knowledge of how these structures work.

Why Use SQLT?

Why Use SQLT?

- Can be used with or without licenses for the Tuning and Diagnostics Packs.
  - Answer these script prompts during the install:
    - "T" if you have license for Diagnostic and Tuning
    - "D" if you have license only for Oracle Diagnostic
    - "N" if you do not have these two licenses
- Gives the **most comprehensive metadata report** about your SQL.
- Derived information presented in impressive and **useful HTML report**.
- The utility is **easy to use** and **saves time and effort** to manually collect configuration and metadata related to a SQL statement.

SQLT and RDBMS Version Compatibility

SQLT and RDBMS Version Compatibility

- Can be used with RDBMS 9.2 and higher.
- Be sure to **download the proper version** of SQLT for the database you intend to use it with.
- See MOS Note 215187.1.

SQLT Installation Overview

SQLT Installation Overview

- Download the proper zip file from MOS Note 215187.1.
- Unzip the file to any appropriate working directory.
- Navigate to the <SQLT_HOME>/install/ directory, invoke SQL*Plus and connect to the database that runs the relevant SQL and execute the **sqcreate.sql** script. Execute the script while connected as "sys as sysdba".
- The installation will create a **database user named SQLTXPLAIN** and these objects in its schema.

```
OBJECT_TYPE             COUNT(*)
------------------- ----------
PROCEDURE                    1
TYPE                         4
SEQUENCE                    12
PACKAGE                     17
PACKAGE BODY                17
LOB                         91
VIEW                       101
TABLE                      212
INDEX                      227
```

SQLT Installation Overview

- The SQLTXPLAIN schema (v11.4.4.2) **requires about 10MB** of space for the installation. **Allocate 50-100MB of space** for this schema to allow for data creation in the schema as the utility is used.

Installation prompts:

1. Connect identifier for TNS connections (optional).
2. SQLTXPLAIN database user password.
3. Default and temp tablespaces for SQLTXPLAIN user.
4. Main application user of the SQLT utility. This user will be the one to execute the SQLT scripts. Note that additional users can be added to the configuration later.
5. Management Pack license, if any:
   - "T" if you have license for Diagnostic and Tuning
   - "D" if you have license only for Oracle Diagnostic
   - "N" if you do not have these two licenses

- After the last prompt, the remainder of the installation should take just a few minutes to complete.

SQLT Installation Overview

- The **latter part of the installation steps could take much longer** than a few minutes if the database has a **large number of objects**, such as in an Oracle EBS installation.
- At the end of the installation, review the *.log files in the <SQLT_HOME>/install/log directory.
- To enable the use of the SQLT utility by additional database users, grant the **SQLT_USER_ROLE** role to those users.
- This role has the system privileges ADMINISTER SQL MANAGEMENT OBJECT and ADVISOR granted to it.
- If you wish to “start over” and just purge the entire SQLT schema and re-install it, execute the <SQLT_HOME>/install/**sqdrop.sql** in SQL*Plus as “sys as sysdba”.
- The installation will create six database directory objects that reference the user_dump_dest directory.

SQLT Installation Overview

- Note that SQLT uses the Oracle CoE Trace Analyzer (TRCA) utility.
- The SQLT installation installs the Trace Analyzer PL/SQL code. See MOS Note 224270.1 for a description of Trace Analyzer.
- If you wish, you can install and use TRCA separately in the same database as SQLT, although there is no need to.

How SQLT Does What it Does

How SQLT Does What it Does

- SQLT works by calling STA and TRCA and its own PL/SQL packages and scripts and performing various analyses and manipulations on SQL execution plans, SQL profiles and trace files.
- The tool outputs reports of its configuration findings and tuning recommendations and of the metadata about the relevant database objects and SQL statements as well as scripts to implement configuration changes and SQL profiles.
- SQLT invokes its own PL/SQL packages that call dbms_spm, dbms_sqltune and execute various dictionary queries. The code of these scripts is in <SQL_HOME>/install/ and <SQLT_HOME>/run/.

Using SQLT

Using SQLT

Usage Overview

- Determine the **appropriate mode** in which to run SQLT.
- **Execute the proper script** with the proper syntax as the database user that executes the particular SQL under normal application conditions.
- Some execution modes may take a couple of minutes or less and some may take much longer, depending on the SQL to be evaluated.
- Find the zip file, unzip it and open the ***main.html** file in a browser.
- Read the full report.
- **Key things to look for** – STA Report, STA Script, Session Events, the subsections in the Plans section of the report, “hot blocks” section of the TRCANLZR report.
- Read the recommendations report and script, if they exist.
- **Implement the recommendations in test** environment and evaluate the effects.
- Test the syntax to “un-implement” (back out) the recommendations in the test environment.
- Decide whether to implement the recommendations in production.
- Prepare to “un-implement” the recommendations from production if the results are detrimental.

Using SQLT

XTRACT Method

- Probably the **most common** SQLT usage method.
- This method captures information about the particular SQL from the library cache.
- Invoke SQL*Plus from the <SQLT_HOME>/run/ directory and connect as the database user that would normally execute a particular SQL statement or set of statements and execute this command:

  SQL> start sqltxtract <SQLID>

- The sqltxtract.sql script will produce a zip file in the <SQLT_HOME>/run/ directory. Unzip this zip file and open the *main.html file in a browser.

Using SQLT

XECUTE Method

- Takes as its input a text file containing a SQL statement or STS.
- Invoke SQL*Plus from the <SQLT_HOME>/run/ directory and connect as the database user that would normally execute a particular SQL statement or set of statements and execute this command:

  SQL> start sqltxecute <full_pathname_of_sql_script>

- The sqltxecute.sql script **executes the SQL statement** or statements in the STS.
- Therefore, the length of time that the sqltxecute.sql script takes is **bound by what those SQL statements take to execute**.
- The product of executing the script is a zipfile produced in the <SQLT_HOME>/run/ directory. To use this file, unzip it in any directory and open the *main.html file.

Using SQLT

XTRXEC Method

- The **XTREC** method is a **hybrid of the XTRACT and XTREC methods**. It performs its analysis on a currently running or recently run SQL statement and then produces a script that is used to execute the given statement.
- The literal values substituted for bind variables in the statement are those derived from **bind peeking** at the time of executing the statement with the most expensive plans derived by the optimizer for the statement.
- To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

  SQL> sqltxtrec <SQLID>

  where <SQLID> is the SQL ID of a currently running or recently run SQL statement. To find this SQLID, use Enterprise Manager's Top Activity page or query the v$sql dynamic performance view.

Using SQLT

XPLAIN Method

- The sqltxplain.sql script **executes explain plan** on the the SQL statement or statements in the STS.
- The sqltxplain.sql script does not execute the statement.
- It is re**commended that the XTRACT, XECUTE or XTREC methods be used** before considering using the XPLAIN method because they are more accurate. They are more accurate because **XPLAIN cannot utilize bind peeking**.
- The XPLAIN script will be **least accurate if the SQL statement has bind variables**. If it does not, then it should be fairly accurate.
- The product of executing the script is a zipfile produced in the <SQLT_HOME>/run/ directory. To use this file, unzip it in any directory and open the *main.html file.
- The product of executing the script is a zipfile produced in the <SQLT_HOME>/run/ directory. To use this file, unzip it in any directory and open the *main.html file.

Using SQLT

COMPARE Method

- The **COMPARE** method uses the data in the SQLTXPLAIN database schema from **two different databases** in which a given SQL statement or STS has inexplicably produced **different execution plans** and the goal is to find out why.
- The sqltcompare.sql script takes **two SQLID values** as input and assumes that the SQLTXPLAIN schema data has been exported from one of the databases and imported into the other.
- Alternatively, a third database could be used into which the SQLTXPLAIN data from both databases has been imported.
- The output file is similar to the other SQLT methods and should contain the **reason(s) for why the execution plans used by each database are different**.
- To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

  SQL> sqltcompare <SQLID1> <SQLID2>

Using SQLT

TRCANLZR Method

- This method **invokes the Trace Analyzer (TRCA) utility**.
- This method negates the need to separately download and install the TRCA utility because it is the same utility.
- Syntax:
- Invoke SQL*Plus from the <SQLT_HOME>/run/ directory and connect as the database user that would normally execute a particular SQL statement or set of statements and execute this command:

  SQL> start sqltrcanlzr <full pathname of an existing trace file generated with event 10046>

- The sqltrcanlzr.sql script will produce a zip file in the <SQLT_HOME>/run/ directory. Unzip this zip file and open the *.html file in a browser.

Using SQLT

TRCAXTR Method

- The **TRCAXTR** method extends the functionality of the TRCANLZR method by **first executing the TRCANLZR method** on the given SQL trace file and **then calling the SQLTXTRACT method** on the top SQL that is found in the file.
- To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

  SQL> sqltrcaxtr <full_pathname_of_trace_file>

Using SQLT

TRCASET Method

- The **TRCASET** method is **automatically executed by the TRCAXTR method**.
- In **standalone execution mode**, this method allows the user to select particular SQLID's upon which prior analyses have been performed.
- The user picks one prior analysis to have the **XTRACT method** performed on.
- This method can be executed in standalone mode by navigating to the <SQLT_HOME>/run/ directory invoking SQL*Plus, and connecting to the main application user that executes the SQL in question, and executing this command:

  SQL> sqltrcaset <application_user>

Using SQLT

TRCASPLIT Method

- The **TRCASPLIT** method takes a SQL trace file that was produced with events 10046 and 10053 and **splits the trace file into two separate trace files** – one for the 10046 data and one for the 10053 data.
- The two separate trace files produced could then be separately used with the TRCANLZR method, if desired.
- To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

  SQL> sqltrcasplit <full_pathname_of_trace_file>

Using SQLT

XTRSET Method

- The **XTRSET** method is an **extension of the XTRACT method** in which a set of SQLID's are passed to the script and the script derives the data about these SQLID's from the library cache and/or the AWR and **combines the output of all of them into one report**.
- To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

  SQL> sqltxtrset <SQLID1> <SQLID2> … <SQLIDn>

Using SQLT

PROFILE Method

- The **PROFILE** **requires that you first execute the XTRACT or XECUTE** method on a particular SQLID.
- The script accepts a SQLID as input. When the sqltprofile script is executed, it will prompt for a plan hash value from a list of plan hash values captured from previous executions of XTRACT or XECUTE on the indicated SQLID.
- The user can choose a particular one of these plan hash values and the sqltprofile.sql **script will output a script to install that plan hash value** as the one to be used for that SQLID by use of a SQL profile.
- To execute this method, navigate to the <SQLT_HOME>/run/ directory, invoke SQL*Plus and connect to the main application user that executes the SQL in question, and execute this command:

  SQL> sqltprofile <SQLID>

Using SQLT

XGRAM Method

- The **XGRAM** method involves the usage **of several scripts in the <SQLT_HOME>/utl/xtram/ directory**.
- These scripts are used to make **modifications to stored optimizer histograms** related to particular SQL.
- Before using these, read all of the scripts in the <SQLT_HOME>/utl/xgram/ directory to determine which is appropriate to execute in a given case.

Using SQLT

XPLORE Method

- The conditions to consider using the **XPLORE** method are **rare**.
- The scripts to use this method are in the <SQLT_HOME/utl/xplore/ directory.
- Read the **readme.txt** file in that directory before doing anything else with this method.
- This method is appropriate to use in **rare cases in which the optimizer is producing invalid results** following an upgrade.
- The scripts used by this method will “explore” variations in the optimizer_features_enabled and fix parameters that may have led to the problematic behavior.

SQLT Usage and Maintenance Recommendations

SQLT Usage and Maintenance Recommendations

- Remove old, orphaned SQL profiles.
- Delete old zip files produced by SQLT executions.
- Space consumption by the SQLTXPLAIN database schema.
- To add additional database users to the configuration so that you can run SQLT as those users, grant the role SQLT_USER_ROLE to those users.
- Upgrading SQLT – instructions in the sqlt_instructions.html file (in zip distribution bundle).
- If you wish to “start over” and just purge the entire SQLT schema and re-install it, connect as “sys as sysdba” and execute the <SQLT_HOME>/install/sqdrop.sql.
- Be patient when installing and running the utility on databases that have a *lot* of objects (Oracle EBS, BAAN, etc.). The utility executes queries on dictionary tables like all_objects, etc.
- You will get the best results from SQLTXTRACT and SQLTXECUTE if you execute these scripts as soon after the SQL of interest has started, if not right at the moment it has started execution.

SQLT Usage and Maintenance Recommendations

- Optimizer statistics on the SQLTXPLAIN schema are locked after the SQLT installation. To gather stats on them, you must first unlock them with dbms_stats.unlock_schema_stats(ownname=>'SQLTXPLAIN');
- If you have a database with a *lot* of extents (a lot of rows in sys.fet$), the installation may take a long time. Be patient with the install because subsequent executions of the SQLT scripts should then not suffer some of the performance issues caused by bugs in some database versions, like bugs 2948717, 5029334, 5259025. Some MOS notes like 422730.1 suggest gathering stats on sys.x$ktfbue. Otherwise, upgrading/patching may be required to resolve these.
- Due to bugs like this, it might be helpful, for example, to create and schedule a batch script that drops and re-installs SQLT periodically so that you have the freshest data in tables like SQLTXPLAIN.TRCA$_EXTENTS.
- It is highly prudent to run frequently used and high importance SQL through SQLT on a regular basis and review the findings and recommendations.

SQLT Usage and Maintenance Recommendations

- Consider setting the *optimizer_capture_sql_plan_baselines* parameter to TRUE (default=FALSE).
- Review of some SQLT reports (files provided separately).

References

References

My Oracle Support Notes

- 215187.1 – SQLT main note.
- 224270.1 – TRCANLZR – Tool for interpreting raw SQL traces.
- 1229904.1 – Real time SQL monitoring in 11g.
- 1366133.1 – SQL tuning health check script
- 1401111.1 – Announcement of SQLT webcast on May 15, 2012.
- 199083.1 – SQL query performance overview.
- 376442.1 – How to collect 10046 trace diagnostics for performance issues.
- 398838.1 – FAQ: SQL query performance.
- 276103.1 – Performance tuning using Advisors and manageability features

References

Related References

- 748642.1 – How to create AWR snapshots and baselines.
- 190124.1 – The CoE Performance Method
- 390374.1 – Oracle Performance Diagnostic Guide
- 461053.1 – OS Watcher Black Box Analyzer
- 122669.1 – How to perform a health check on an Oracle database
- Chapter 15 (SQL Plan Management) of Performance Tuning Guide
- Chapter 17 (Automatic SQL Tuning) of Performance Tuning Guide
- sqlt_instructions.html file in the SQLT zip bundle.
- Files in the <SQLT_HOME>/doc/ directory, including two excellent Powerpoint files.

PL/SQL Packages

- DBMS_SQLTUNE
- DBMS_AUTO_SQLTUNE
- DBA_SQL_PROFILES
- DBMS_SPM

References

Dictionary Views

- DBA_ADVISOR_TASKS
- DBA_ADVISOR_EXECUTIONS
- DBA_ADVISOR_FINDINGS
- DBA_ADVISOR_RECOMMENDATIONS
- DBA_ADVISOR_RATIONALE
- DBA_SQLTUNE_STATISTICS
- DBA_SQLTUNE_BINDS
- DBA_SQLTUNE_PLANS
- DBA_SQLSET
- DBA_SQLSET_BINDS
- DBA_SQLSET_STATEMENTS
- DBA_SQLSET_REFERENCES
- V$ADVISOR_PROGRESS
- V$SQL
- V$SQLAREA
- V$SQLSTATS
- V$SQL_BINDS

Q & A

Thank you for coming!
Enjoy using SQLT!
Enjoy the conference!
Thank you NYOUG!

Contact: gilpink@tusc.com