SIJITCS Vol.: 01 II Issue 01 II Pages 01-45 II April 2017

Performance Optimization of Transactional Business Intelligence Queries In Cloud Using Innovative SQL Monitoring, Capturing & Replay Tool

Arjun K Sirohi [1*], Dr Vidushi Sharma [2]

[1] School of ICT, Gautam Buddha University, Yamuna Expressway, Greater NOIDA, Gautam Buddha Nagar, UP, India, asirohi@yahoo.com

[2] School of ICT, Gautam Buddha University, Yamuna Expressway, Greater NOIDA, Gautam Buddha Nagar, UP, India, vidushi@gbu.ac.in

*Corresponding Author asirohi@yahoo.com

Abstract: Objectives: Optimize performance of Transactional Business Intelligence (TBI) SQL queries in Cloud applications by capturing, storing and executing SQL workloads for any application.

Methods/Statistical analysis: Benchmark experiments were conducted using Oracle RDBMS 11gR2 using representative SQL queries from Oracle's Fusion CRM TBI Applications. Regression analysis was used to predict the impact of system environment changes on the performance of a SQL workload with high accuracy.

Findings: The benchmark experiments established very promising results. We recorded repeatable, significant gains in not only the four measures of individual SQL performance but also at the database resources level. The proposed architecture enables the creation of a repository of SQL queries that is used for regression analysis. Query Response Time (RT) improvements ranging from 5% to 745 times, Hard-Parse Time improvements from 1% to 208 times, Logical I/O or Buffer Gets' improvement ranging from 43% to 454 times and SQL-Shared-Memory reduction by up to 52%.

Application/Improvements: Our proposed architecture is directly applicable to improve performance of all TBI applications that use Oracle databases, especially in the Software-as-a-Service (SaaS) and Cloud Models.

***Keywords*:** RDBMS; SQL query performance; Cloud and SaaS Applications; Transactional Business Intelligence; Regression Analysis.

1. INTRODUCTION

The last few years have witnessed a big rise in the use of Software as a Service (SaaS) applications and the shift towards Cloud computing. Oracle [1] is a widely used and popular database platform used by businesses large and small as well as institutions like governments and defence departments. According to Gartner, Oracle is number 1 in worldwide RDBMS software market share with 41.6 percent market share [2] and is used by many popular Cloud and SaaS application vendors like Salesforce.

There are many popular enterprise applications like Salesforce [3] and Oracle Fusion [4] and Transactional Business Intelligence Applications [5] that use Oracle as the underlying database. To manage these large number of database installations, businesses and institutions employ a variety of Information Technology (IT) staff including Database Administrators (DBAs) and application developers. Users of such applications are usually not aware of and are not exposed to the underlying Oracle database that drives the application. They view an application as being good or bad based on their experience with it, especially on how fast they can get responses from the application. The response times (RT) that users experience while clicking on various options on the application browser is a direct manifestation of the performance of the underlying Oracle database which is the research subject of many database researchers [6, 7, 8, 9, 10, 11, 12,

13]. The one artefact of Oracle database that lies at the centre of this performance is the physical SQL (Structured Query Language) which the standard query language for all relational databases like Oracle. It is not uncommon to hear that an application's performance is as good as the underlying SQLs' performance.

In brief, SQL performance is one of the most important areas of focus for DBAs as well as for application architects and developers. Over the years, many different methodologies and processes have been used to capture slow SQLs and tune them. Many independent vendors like Dell's Toad [14] and Idera [15] have created whole businesses around SQL Monitoring and Tuning. SQL monitoring and tuning is an essential task and is usually performance by DBAs and application developers. However, it requires a high level of expertise in many different areas like SQL design, access path optimization etc. and is also a very time consuming and involved process.

Oracle database provides some add-on features like STS (SQL Tuning Sets) to help with this task but these features do not go far enough. The high level architecture and components of a SQL Tuning Set is depicted in Figure 1. The limitation around such tools prevent their full exploitation at industry scale. For example, one cannot start capturing SQLs or stop the capture at will using these tuning sets. In addition, generating easily readable reports from the SQL performance testing can be an arduous task. Regression testing for SQL performance consistently and repeatedly is a big challenge while upgrading database or application versions or when applying mandatory patches or changing database parameters. This directly and indirectly affects the productivity of DBAs, application developers as well as business users of the applications.

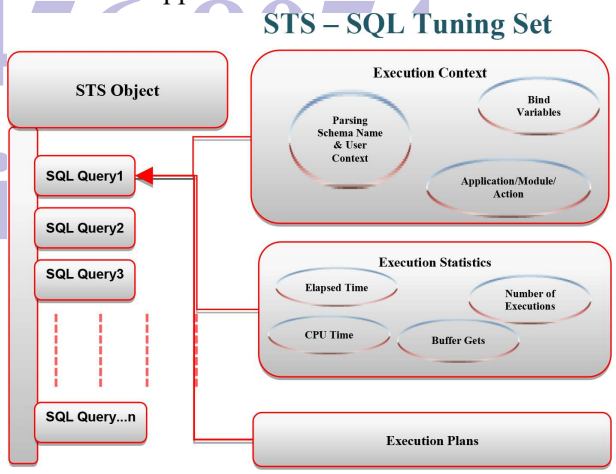

Figure 1: Oracle SQL Tuning Set.

2. MOTIVATION, GOALS AND SCOPE

This paper presents the research and creation of a SQL Monitoring and Capturing Tool to accomplish following business goals and fill gaps in existing solutions:

- Establish cost savings to the business by increasing the productivity of DBAs and application developers through the easy-to-use and consistent tool for monitoring and capturing SQLs.
- Decrease the time needed to conduct regression testing when applying database or application patches or while making changes to database parameters that will affect SQL performance.
- Increase the availability of databases and applications for business users by eliminating the downtime needed for rolling back patches and fixes to database and applications that may have caused regression in performance, an unintended consequence of some patches.

• Increase the productivity of applications' business users by ensuring that the use of the tool leads to better performing databases and applications in the enterprise.

The scope of this project is to build a SQL tracing and capturing toolkit using the Oracle SQL Tuning Set framework that that will help capture and store SQL workload information of any application. The main objectives are:

- Support DBAs and application developers with a toolset to capture SQLs and facilitate them with their task of SQL monitoring, capture and performance tuning as well as trending.
- Provide an easy to implement and use infrastructure to maintain data for SQL analysis/tuning.
- Enable DBAs and application developers to track slow SQLs more efficiently.
- Out of scope: SQL tuning itself is beyond the scope of this project. However, the captured tuning sets can be replayed with changed parameters settings or patches and performance compared using standard Oracle build-in DBMS packages.
- Statement of Work: For Oracle DBAs and application developers, build a SQL tracing and capturing toolkit using the Oracle SQL Tuning Set framework that that will help capture and store SQL workload information of any application on demand.

3. REQUIREMENTS, BUSINESS RULES AND ASSESSMENT CRITERIA FOR TOOL

The system will include the following functionality:

- API to start and Stop Capture of SQLs on demand from command line SQLPlus.
- Ability to run the SQL capture in background mode at the database without any dependencies for the application being tested.
- Ability to set up automatic capture of SQLs.
- Built-in and easy-to-use filtering of SQLs based on given criteria.
- For long term use, ability to provide statistics for a specific time period.
- Ability to easily generate HTML reports from captured data.
- Ability to customize reports by accepting a list of SQL sets.

Graphically, the requirements can be illustrated with the help of diagram in Figure 2 for ease of understanding. This led us to our formulation of the

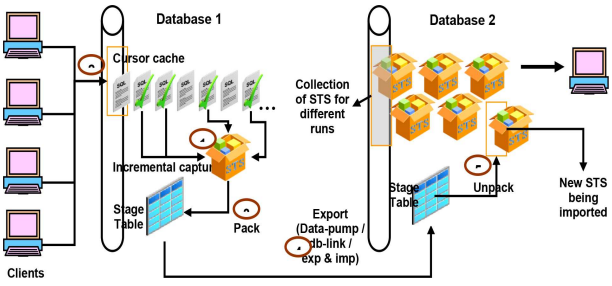

Figure 2: Architectural Scheme for SQL Monitoring and Capturing Tool.

The system supports the following Business Rules:

- The SQL Monitoring and Capturing toolkit will be implemented as a set of database procedures and tables on top of the STS framework provided by the Oracle database.
- Users of the SQL Monitoring and Capturing toolkit will have SYSDBA privileges to install the package in the database.
- Once setup, the toolkit API will be used to capture SQLs and save them as SQL tuning sets.
- The SQL tuning sets should be transportable across databases.
- The toolkit should be completely independent of business applications connecting to the database i.e. there should be no requirement for Business Applications' implementers or end-users to install or configure anything in the applications.
- Only DBAs, application developers or SQL tuning experts need to understand and be trained on using the toolkit.
- Application end-users do not have permission to access captured SQL tuning sets.

The following Assessment Criteria was used for evaluation of the SQL Monitoring and Capturing toolkit:

- Be based on Oracle RDBMS standards.
- Be easy for DBAs and application developers to implement.
- Be easy to use.
- Be simple and clear.
- Have appropriate security inbuilt.
- Be as "flat" as possible. This means minimal inputs should be needed from DBAs or application developers using the toolkit.

4.DISCUSSION AND PROPOSED ARCHITECTURE

Given the many factors affecting the performance of TBI SQLs, it was essential to design a tool that captured all required information that is needed to re-run SQL query workloads repeatedly with different parameters and bind variable values in order to compare performance from run to run. The object identification begins with a trace of the user requirements. The five main modules identified in the tool construction are:

- Setup. A setup script will be provided to be run on the database schema to setup the required packages and tables.
- Capture. This will provide mechanisms to capture SQL Tuning Sets - manual creation of STS as well as automated creation of STS.
- Stop Capture. This package will provide a mechanism to stop the scheduled or manual capture jobs.
- Querying the captured SQL Sets. This will encompass pre-configured scripts to generate HTML reports which can be customized to accept a list (or) a range of SQL tuning sets.
- Modification of report parameters. A function will be provided to modify the configuration settings of report module.

The SQL Tuning Sets framework design captures the essential scripts, packages and objects that perform the desired functions projected in the requirements. The design addresses two important issues:

- Maintainability. The system shall be capable of being maintained by people who are not involved

in the original design. Every step of the design process shall therefore be well documented.

- Testability. Test cases shall be designed prior to the development of code for each class. This ensures that every module works as desired, and there will be no surprises at the end.

Modularity is inbuilt into the tool. The capability to incorporate and build modules in future includes the following functionality:

- Creation of a central SQL repository in another database that can store all captured SQLs.
- Providing a mechanism to automatically transport the captured SQL sets to the above SQL repository.
- Providing an interface to use Oracle database provided DBMS_SQLPA package to help users predict the impact of system environment changes on the performance of a SQL workload
- Use DBMS_SQLPA package provided by Oracle database to implement the SQL Performance Analyser using the CREATE_ANALYSIS_TASK Function to create an analysis task for a set of captured SQLs, execute this created analysis task multiple times using EXECUTE_ANALYSIS_TASK function and then use the REPORT_ANALYSIS_TASK Function to display the results of the analysis task.

5. RESULTS FROM IMPLEMENTATION OF PROPOSED ARCHITECTURE

Our proposed architecture was implemented for capturing and monitoring of SQL sets in Oracle databases for Oracle Fusion Applications. To recap the measurement criteria, as stated in the project proposal, is shown below.

Requirement	Measurement
API to Start and Stop Capture of SQLs on demand from command line SQLPlus	Start and stop capture of SQLs in a database where SQL Monitoring tool is installed, should be simple and easy to use.
Ability to run the SQL capture in background mode at the database without any dependencies for the application being tested	The SQL capturing process should be completely independent of any application.
Ability to set up automatic capture of SQLs	The tool should provide an easy way to setup and start automatic capture of SQLs in the database.
Built-in and easy-to-use filtering of SQLs based on given criteria	The user should be able to easily filter SQLs using given criteria
For long term use, ability to provide statistics for a specific time period	Generation of SQL statistics for any given time period or SQL Sets in a simple and easy way.
Ability to easily generate HTML reports from captured data	Nicely formatted HTML reports that should be easy to read.
Ability to customize reports by accepting a list of SQL sets	Easy customization of reports by making simple text changes.

The following table provides a comparison between the completed project and the specified measurement criteria.

Requirement	Completed Project	Specified Measurement Criteria
API to Start and Stop Capture of SQLs on demand from command line SQLPlus	Procedures schedule capture and start capture for starting and stop scheduler procedure to stop capture provide this functionality.	Start and stop capture of SQLs in a database where SQL Monitoring tool is installed, should be simple and easy to use.
Ability to run the SQL capture in background mode at the database without any dependencies for the application being tested	Using the procedures schedule capture with relevant parameters	The SQL capturing process should be completely independent of any application.
Ability to set up automatic capture of SQLs	Procedures schedule capture with relevant parameters like (PARSING_SCHEMA =>'BAW,ORAESS', JOB_REPEAT_INTERVAL=>'FREQ=HOURLY;BYHOUR=7,19', CAPTURE_MAX_TIME_LIMIT=> 3600)	The tool should provide an easy way to setup and start automatic capture of SQLs in the database.
Built-in and easy-to-use filtering of SQLs based on given criteria	Procedures schedule capture and start capture with filtering on schema/user	The user should be able to easily filter SQLs using given criteria
For long term use, ability to provide statistics for a specific time period	sqlmfrpt.sql scripts with relevant parameters	Generation of SQL statistics for any given time period or SQL Sets in a simple and easy way.
Ability to easily generate HTML reports from captured data	sqlmfrpt.sql scripts with relevant parameters	Nicely formatted HTML reports that should be easy to read.
Ability to customize reports by accepting a list of SQL sets	Procedure modify parameters for customizing the report criteria	Easy customization of reports by making simple text changes.

As can be seen from the above table, the completed project completely satisfies the requirements of the system, and that the completed project is consistent with the measurement criteria as stated in the project proposal.

6. CONCLUSION

In this paper, we presented a tool for monitoring and capturing of SQL queries for later replay and regression testing. A future work would be to create a unified repository database that could be loaded with all captured SQLs from applications connecting to such a database in cloud environments. For example, in any software development or IT organization that creates or uses software applications running against an Oracle database, it would really be very helpful if they could create a central repository where all SQLs being executed in their database could be captured. This would allow them to repeatedly test these SQLs by replaying these using standard Oracle database options or patches etc. With the advancements in artificial intelligence and machine learning, it would be possible to train existing databases like Oracle to learn from the captured data on the execution patterns of such SQL queries.

The proposed improvement over existing SQL Tuning Sets architectures has direct applicability to many SaaS and Cloud applications and can lead to the improvement in their adoption rates. We feel that our work is only a humble beginning and we expect further research to be done on the performance and monitoring solutions in enterprise applications as more and more vendors as well as consumers move towards the Cloud and SaaS model. The aim should be to minimize the impacts of using such monitoring and capturing tools on the performance of enterprise business applications so that adoption becomes easy.

REFERENCES

[1] Oracle Database Concepts 11gR2 accessed on the World Wide Web at http://docs.oracle.com/cd/E14072_01/server.112/e10713/qllangu.htm#CHDFCAGA

[2] Gartner Blog accessed on World Wide Web at http://blogs.gartner.com/merv-adrian/2016/04/12/dbms-2015-numbers-paint-a-picture-of-slow-but-steady-change/

[3] Salesforce Applications and Platform accessed on World Wide Web at https://www.salesforce.com/products/platform/overview/?d=70130000000rz4W

[4] Oracle Corporation. Oracle Fusion CRM Application accessed on the World Wide Web at http://www.oracle.com/us/products/applications/fusion/customer-relationship-management/index.html

[5] Oracle Corporation. Oracle Transactional Business Intelligence Administration Guide accessed on the World Wide Web at http://docs.oracle.com/cd/E15586_01/fusionapps.1111/e21363/frameset.htm?apa.html#OTBI_Admin_guide_book_12

[6] Oracle Corporation. Oracle Database Performance Tuning Guide 11g Release 2 (11.2) accessed on World Wide Web at http://docs.oracle.com/cd/E11882_01/server.112/e16638.pdf

[7] Ilyas, I.F. and W.G. Aref, "Adaptive Rank aware Query Optimization in Relational Databases", ACM Transactions on Database Systems (TODS), Volume 31 Issue 4, pp. 257-1304, December 2006.

[8] A. Sirohi, V. Sharma, "Performance Optimization in Transactional Business Intelligence Applications' Query-Generation Development Cycle", Indian Journal of Science and Technology, accessed on World Wide Web at

http://www.indjst.org/index.php/indjst/author/submissionReview/100471

[9] K.S.K. Kihong, K. C. Sang, "Optimizing Multidimensional Index Trees for Main Memory" SIGMOD '01 Proceedings of the 2001 ACM SIGMOD international conference on Management of data, pp. Pages 139-150, 2001.

[10] W. Powley, P. Martin, and P. Bird. "Dbms workload control using throttling: experimental insights", CASCON '08, Proceedings of the 2008 conference of the center for advanced studies on collaborative research: meeting of minds, Article No. 1, New York, NY, USA, 2008.

[11] K. P. Brown, M. Mehta, M. J. Carey, and M. Livny. "Towards Automated Performance Tuning for Complex Workloads", VLDB '94, Proceedings of the 20th International Conference on Very Large Data Bases, pp. 72-84, 1994.

[12] H. Pang, M. J. Carey and M. Livny, "Multiclass query scheduling in real-time database systems," in IEEE Transactions on Knowledge and Data Engineering, vol. 7, no. 4, pp. 533-551, Aug 1995.

[13] J. Krueger, C. Tinnefeld, M. Grund, A. Zeier, and H. Plattner, "A case for online mixed workload processing", In Proceedings of the Third International Workshop on Testing Database Systems (DBTest '10), Article 8 , 6 pages, New York, NY, USA.

[14] Dell Corporation. Toad for Oracle, accessed on World Wide Web at https://software.dell.com/products/toad-for-oracle/

[15] Idera. SQL Diagnostic Manager accessed on World Wide Web at https://www.idera.com/productssolutions/sqlserver/sqldiagnosticmanager/best-sql-monitoring-tools

[16] Oracle Corporation. SQL Tuning Sets and Automatic SQL Tuning accessed on World Wide Web at http://docs.oracle.com/cd/E28271_01/server.1111/e16638/sql_tune.htm.

(17). Leena N.F., Jaykumar V., Issac S.S., 'Assessing CRM Practices in Hotel Industry: A Look at the Progress and Prospects', Indian Journal of Science and Technology, 2015 Mar, 8(S6), pp .1-9.