Preliminary Design Document

HP: Big Data Analytics
Group 13

Oregon State University
CS 461
2016-2017

Prepared By:
Nic Desilets, James Stallkamp,
and Nathaniel Whitlock

December 1, 2016

Abstract

This document is focuses on a detailed brake down of ”how” the individual components of our project will be implemented. Each of the three essential pieces: SQL development environment; parallelization and partitioni ng; and reporting tools are covered with fine granularity.

Elaborate more here in the future...

CONTENTS

1 INTRODUCTION

The following sections delve into the development design and considerations that need to be outlined prior to the development cycle. James Stallkamp covered the first four sections focusing on context, composition, logical, and dependencies viewpoints. This was followed by three sections from Nic Desilets which covered the information, interface, and structure viewpoints. Finally, Nathaniel Whitlock spent time on the interaction and resources viewpoints.

2 CONTEXT VIEWPOINT

2.1 Design Concern

This toolkit will be used in the context of database queries hat are not utilizing the host machines resources. The optimizer analyzes the structure of a query and allocates resources for and executes the query. Some queries are structured such that the optimizer allocates resources in a way that much of the host machine's resources are not being utilized. The toolkit being designed would be a tool to help understand the performance of these queries that result in the under utilization of resources. This toolkit will be a package of scripts SQL scripts that can be run in an environment like SQL developer. he primary concern of the toolkit will be to gather and display performance statistics that are stored on internal performance tables in the database. This toolkit will provide a library of scripts that can be used to gather various kinds of information on a query and how the database executed the query. Users will be able to execute these scripts and have the output piped into files or displayed on a web page. A User could then use this information to better understand how the database is executing a query.

2.2 Design Elements

This toolkit will largely be standalone, it will interact with one other entity and have one type of user. The only other entity that this toolkit will interact with is the database that it is working with. The SQL scripts will interact with the database by querying internal performance tables for data. The user of this toolkit is a database user that is trying to improve or understand the performance of a query. Users will interact execute a script or batch of scripts and recieve an output. This output will be dependent on the user's input and scripts selected, it will be various performance metrics and information a given query. This data can be viewed by the user in a file or through a web browser.

3 COMPSITION VIEWPOINT

3.1 Design Concerns

This toolkit will gather and display performance statistics and will do this through SQL scripts. The toolkit will fundamentally be comprised of many SQL scripts, which are SQL statements which can be run independently. The combination of all the scripts will function as the toolkit and retrieve data on a specific query. Each of These scripts will retrieve various pieces of data or performance metrics. The data will be used together to display all the required information on the performance of a specific query. Each script acts as a function in that each script will retrieve or output a specific piece of data or statistic.

3.2 Design Elements

The individual scripts will be the elements of the system and will act like functions. Each script will provide a different function by producing or retrieving a different performance statistic. Most of the scripts or elements will be stand alane, a database user could run one or any number of the various scripts in sql developer and receive the combined output. The primary attribute of a script is its purpose, each script will have a different purpose, either to retrieve or output a specific performance metric.

3.3 Function Attribute

All of the scripts will be accessing various internal performance metric tables on the database. Each script would retrieve and analyze one of various performance metrics like memory usage, cpu usage, I/O, and many other performance metrics. The scripts will be different in that each script will retrieve or output a different kind of statistic. Each script within the toolkit will be providing a different a different data point on attributes like memory usage, cpu usage, etc.

4 Dependency Viewpoint

4.1 Design Concerns

This toolkit has two significant dependencies, it is however a very versatile tool because almost anyone with access to the database can use it. Any user of the database can use this tool to retrieve information on how their query ran. The toolkit is based on Oracle's linux SQL technology and as a result the toolkit will only work on a Oracle SQL technology based database. The toolkit requires Oracle sql because the toolkit uses SQL statements to access data stored on internal performance tables. The SQL statements also refer to entries in internal tables that may be specific to Oracle version 12c, meaning it is strongly recommended to be using this toolkit on a database that has Oracle version 12c or newer.

4.2 Design Elements

The scripts in the tool kit require access to the internal performance tables on the database. The user must have access to those database tables for the toolkit to access them.

5 Information Viewpoint

5.1 Design Concerns

The primary source of data for the toolkit will come from the internally managed performance metrics tables in Oracle Database 12c. Users will be able to request information about a particular query ran against the database through the use of this toolkit and these performance tables. The data returned from the toolkit regarding a particular query can include information such as time elapsed, processor usage, memory consumption, degrees of parallelism, and other related metrics. The primary persistent data structure for the toolkit is going to be the database itself.

5.2 Design Elements

The data items that are retrieved by the toolkit are stored in a permanent data store (Oracle 12c). Access of this data is provided through the use of SQL Developer in which the toolkit scripts are intended to be run with. SQL Developer initiates a connection to the database to retrieve the relevant performance information needed by the toolkit scripts.

More specifically, the information that is retrieved by the toolkit comes from different tables within the database. The records stored in these database tables are generated after the user runs a query. After completion of a query, the toolkit can then be used to query the database again for metrics information about the previous query ran by the user. This information is then processed and returned to the user in a human readable format.

5.2.1 Data Attribute

The metrics information that the database contains spans a broad range of different types of information used for profiling the performance of queries. The specific types of this information ranges from numbers, strings, to date representations. For example, timing information is represented as numbers, metrics descriptions are represented as strings, and start and end times would be date representations.

All of this information is categorized and put into specific tables that relate to a certain domain of information relating to a query. Examples of this can include storing memory usage information into a table that describes how much memory was consumed over time when processing a particular query. The data stored in these tables is intended to be used to offer insights into query performance.

5.3 Example Languages

6 Interface Viewpoint

6.1 Design Concerns

The primary interface for this toolkit is going to be SQL Developer. The toolkit itself consists of SQL queries that are to be executed inside the SQL Developer environment. Users will select which toolkit query they want to run based on what type of information they need. Upon selecting a query, the user will then load it into SQL Developer which sends the query to the database. Results returned by the database will then be presented to the user in a human readable format.

6.2 Design Elements

The primary elements of SQL Developer include the SQL Worksheet window. This where queries are loaded and then sent off to the database for execution. Once the database has completed processing a particular query, the results from the database are then returned and presented to the user in a new results window contained with the SQL Worksheet window. Users will be able to scroll through the results if there is a large amount of information.

6.3 Example Languages

7 Structure Viewpoint

7.1 Design Concerns

The two primary structures involved with the toolkit are SQL Developer and Oracle Database 12c. Oracle Database 12c is a data store and SQL Developer is an environment that is used to interface and interact with Oracle Database. SQL Developer must first initiate a connection to Oracle Database 12c in order to execute any queries. If the connection is successful, the user can then start loading and executing queries via SQL Developers database connection.

7.2 Design Elements

SQL Developer connects to Oracle Database 12c in order to retrieve information contained in its data store. The purpose of SQL Developer is to serve as a workspace for the user to input queries, have the queries executed, and view the output returned by the queries. The database on the other hand is a datastore that houses information relating to HP PageWide Press printers and also internal metrics relating to query performance. The connection from SQL Developer to Oracle Database 12c serves as the means of bi-directional communication between the two structures.

7.3 Example Languages

8 Interaction Viewpoint

8.1 Design Concerns

The main design concern in terms of the interaction between system entities is an open line of communication with the Oracle database. If this connections is severed, then there is no way for the users personal computer to execute a query on the database. Therefore, the toolkit will not perform and the transaction will be impossible. In terms of allocation of responsibilities in collaborations with the toolkits use at HP, the user's machine will only be responsible for composing the query and sending it through to the database running on the server. The calculations and each procedural step of the toolkit analysis will be processed remotely and the results and statistical report should return to the user.

8.2 Design Elements

The entities involved at various levels of action are as follows; Engineer/DBA, personal computer, server, and Oracle database. During the transaction from the user to the server there are multiple wait instances where one of the host systems sits idle waiting for the final result response. Methods used in the communication process are beyond the scope of our toolkit and will be handled by higher order systems.

8.3 Examples

The process of interaction between the entities involved in a transaction with our toolkit can be summarized as request from a user and a response from the target Oracle database which is composed of their desired results and a performance statistic report. The report will have information about hardware optimization and run-time analysis. This information can then be used to quantitatively compare queries run with different system parameter settings, by comparing the run-time of multiple experiments a optimal design can be identified.

Fig. 1. Sequence of actions once query of interest and toolkit are executed by the user.

9 Resource Viewpoint

9.1 Design Concerns

The purpose of the toolkit is to monitor the system processing a query in order to generate a report which outlines the performance metrics. There should be concern placed on the overhead processing time of the toolkit query. The only inherent performance concerns for our toolkit would be the syntax used to build the toolkit query, which could be evaluated with the SQL explain plan. This toolkit should be portable to all systems capable of hosting Oracle database software so it should be lightweight in terms of size.

9.2 Design Elements

The design entities we are interested in are modeled after the system properties we are interested in measuring. Time elapsed is the main entity but we also have CPU cores, disk I/O, and parallel coordinator. The time elapsed entity will allow us to track the processing time for the query being evaluated, this will be useful when comparing the run-time of identical queries given different configuration. The CPU cores entity will provide information about how many individual cores were used out of the total available cores and how much work each was assigned. The disk I/o entity

will be used to monitor swap space needed due to lack of physical memory resources. Together CPU cores and disk I/O will help us monitor how much of the available hardware is being utilized during SQL processing. Finally, the parallel coordinator entity will provide information about the type and level of parallelism that was used.

In terms of constraints for each entity, the time elapsed entity should ideally be as low as possible to enable real time access to WebPress performance data. There are a total of 72 core processors available during query execution, so the CPU core entity is limited by that number. The disk I/o entity should ideally have a minimal amount of activity meaning that the SQL plan the optimizer put together was efficient and the initial data fetch was as well. If a execute a query with parallelism enabled and provide a max of some number n individual processors, then each one of the available cores should be used.

9.3 Examples

By isolating each one of the above stated entities we will be able to generate a report with enough information to tune a users queries as well as provide the metrics needed to quantitative comparison. An example of this would be testing the effectiveness of a table partition design in respect to a query that performed calculations on specific values and took quite some time. By monitoring these entities, we will be able to see how many cores were made available to the coordinator, how much swap space was used, and how many parallel processes were estimated to be efficient. Additionally, we will have a custom metric calculated from time elapsed from the point of query execution.

Fig. 2. Simple visualization of the resource entities and the database.

10 Conclusion

Temp...

Kirby Sand	Date
Andy Weiss	Date
Nic Desilets	Date
James Stallkamp	Date
Nathaniel Whitlock	Date