Technology Review

HP: Big Data Analytics
Group 13

Oregon State University
CS 461
2016-2017

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

November 13, 2016

Abstract

This document is a review of three essential roles within our project SQL development environment, Parallel and Partitioning, and Performance Reporting. From each of those integral pieces, three or more technology options were evaluated and compared. The desired SQL development environment was deemed to be Oracle's SQL Developer because ... Additionally, Oracle database software was chosen as the best framework for parallelism and partitioning. Finally, the decision was made to create a reporting tool from scratch in order to allow more customization for data output. The resulting decisions will act as the initial path forward until there is a need to consider one of the other options.

CONTENTS

1 Introduction

Our project to develop a SQL performance toolkit has been distilled into three different pieces: SQL development environment; Parallel and Partitioning; and Performance Reporting. The following sections review technologies for each of these pieces. The goal of this document is to compare available technologies that may be of use to our team during the developmental stages of the project.

Elaborate...

2 SQL Development Environment

2.1 Purpose of a SQL Development Environment

2.2 Existing Technologies

2.2.1 SQL Plus

2.2.2 Oracle SQL Developer

2.2.3 Toad

2.3 Conclusion

3 Parallelism and Partitioning

3.1 Purpose of Parallelism and Partitioning

When data sets become large or the aggregate functions used to extrapolate statistics from a table are too complex the database can take a long time to return anything useful. Long periods of latency between query execution and the return of desired data can make the system practically unusable. Techniques such as parallelism and partitioning can be implemented in order to increase computational efficiency and reduce the observed time it takes to return a query.

Parallelism refers to the concept of dividing a large task, such as the processing of a complex query, by breaking it up into specified chunks in order to distribute the work among additional nodes or processes. Before a query is executed the plan for execution can be viewed, this plan lists each of the steps functions and associated cost. During the optimization process, the cost of each step is minimized by tuning the processing options which results in an efficient execution plan [1]. Logically grouped steps from the execution plan can then be processed on different nodes in order to reduce the time it takes to complete processing. The degree of parallelism is a term which refers to the number of processors that are utilized during program execution [2]. As the number of processors splitting the workload increases, the time needed for computation decreases.

Partitioning refers to the logical separation of ranges in a table, index, or indexed table. The general idea is to make independent subdivisions that would assist in the processing of a query. If you had sales data for several years and wanted to calculate the average sales per month, the table of data could be logically partitioned by month and processed on separate nodes. By dividing the data set among many nodes, each table partition can be processed in concert leading to a quicker return of target data.

Together, parallel processing and partitioning tactics act in a divide an conquer like fashion. This is accomplished by breaking a large problem into workable sub units, computing, followed by a reassembly of data. These techniques are some of the most effective method at tuning SQL queries on large data sets.

Below is a discussion of how parallel queries and partitioning are implemented within the three database technologies of our interest.

3.2 Database Technologies

3.2.1 Oracle

The Oracle Corporation has been around for over 39 years and is largely known for database software which got its name from a CIA-funded project that one of the founders had previously worked on. Oracle 12c Enterprise edition has been designed to be used by companies and data intensive users. This means that it is very reliable and there is support available for the service, these benefits come with a license fee that can be quite costly. Oracle database relies on a share everything architecture, meaning that there is no requirement for pre-defined partitions. However, Oracle partitioning offers the same processing potentials as a share nothing architecture [3]. The current version of the Oracle database software is 12.1.0.2 and was released in July 2014.

According to Oracle documentation, every SQL statement undergoes optimization where parallel execution may be selected. If parallel execution is initiated, then the user session takes the role of query coordinator. The coordinator calculates how many parallel processes are needed, then the required read steps are performed in serial and the steps which can be executed in parallel are processed as such. Parallel queries and subqueries are handled by evaluating two components: the degree of parallelism (DOP) and the decision to parallelize. The degree of parallelism is determined most often by the table being modified by an INSERT, UPDATE, DELETE, or MERGE command. In order to use parallelization you must include a statement level hint to indicate the desire to parallel process, the objects referred to need a parallel declaration associated with them, and Auto DOP must be enabled [3].

The Oracle database software has wonderful documentation which outlines their predefined table partition methods. A list of the methods with a brief overview of its function are outlined below:

Partition Methods

- Range Partitioning
  - Maps data based on ranges of values of the partitioning key, this is the most common type of partitioning and is often implemented with dates or times [4].
- Hash Partitioning
  - Maps data based on based on the Oracle hashing algorithm which is applied to the partitioning key. All partitions have even distribution of data making them similar in size [4].
- List Partitioning
  - Maps data by applying a list of specified partitioning key values that should go in each partition [4].
- Composite Partitioning
  - After applying one of the above partitioning schemes, another partition scheme is applied to each partition. The result is subpartitions which represent a logical subset of the data [4].

3.2.2 PostgreSQL

PostgreSQL is an open source object-relational database system that was developed by a team of volunteers in 1996. Although this database software is free it is feature rich and standards compliant. It is also highly customizable allowing stored procedures written in more than a dozen programming languages to be executed. Some potential drawbacks of the software include: installation and configuration difficulties; psql command line different than traditional commands; sheer number of features can take years to learn [5].

PostgreSQL has a feature called ”parallel query” which evaluates an optimizer serial query plan in order to check if the query contains steps which can processed independently of each other. Though this is not true for small simpler

queries, more complex queries have steps which can be split off, processed separately, and recombined [6]. Queries that benefit from this feature the most occur when a query reaches out to a large amount of data for calculations and only returns a few rows. Within the PostgreSQL system each separate node or process used during parallelization are referred to as ”workers” . The number of workers the planner has available is determined by an internal variable called max_worker_processes, this variable is initialized at server start with the default value of eight but it can be manually set [6]. It is possible to utilize less than the maximum number of workers and still have an optimized parallel query plan.

Basic table partitioning is supported by PostgreSQL. The partitioning is implemented by creating a master table, as well as multiple child tables that inherit from the master table and act as logical partitions. Table constraints are then set in order to define what data is placed in each partition. The data from the master table is then copied to each appropriate partition and given a partitioning key, the result is smaller tables which are subdivided by a declared expression [7]. Though there was not predefined table partition methods in the PostgreSQL documentation, most of the methods outlined in the Oracle section (Range, List, Composite) could be achieved through the table constraints applied to each child partition.

3.2.3 Redshift

Redshift is a cloud based data warehousing product from Amazon Web Services (AWS) which is based on the 2005 release of PostgreSQL 8.0.2. Since Redshift is cloud based there are no upfront fees from hiring experienced individuals to set up servers or configure the system. According to the AWS site, Redshift is a petabyte scale data warehouse which is fault tolerant with built in encryption. They also claim that there is no need for tuning to maintain performance and speed since the system is self managed, this service is offered at $\$1,000$/TB per year [8].

Unlike traditional row-oriented databases, Redshift uses columnar or column-oriented storage. Columnar storage works by reading only the values from a row which are needed to complete the transaction. This means if a query is executed that only needs to evaluate the value of one out of four columns, then only the value from that column needs to be loaded into memory, thus drastically reducing the disk I/O [9]. With row-oriented storage, each of the rows containing the values of interest would have to be read into memory.

Redshift is a classified as a shared nothing or Massively Parallel Processing system. Shared nothing refers to the fact that no single node has a complete view of the database, and therefore cannot communicate with each other by sharing data during processing. A cluster is a set of nodes, each node having an independent operating system and memory. The disk storage of each node is broken down into units referred to as slices, the number of slices per node is dependent on the number of nodes within the cluster. Though there are settings for some system configuration, the distribution of work over the compute nodes is automatically handled in order to create a simple experience for the user [10]. Redshift does not support the partitioning concept, rather it relies on the distribution style option selected by the user as well as distribution and sort keys [11]. There are multiple options for distribution styles which work to redistribute rows to compute nodes when a join or aggregation is needed.

Distribution Styles

- Even Distribution
  - Leader node distributes rows across slices in equal proportions in a circular fashion. This is the default distribution style but is only suitable if the table does not participate in joins [10].
- Key Distribution
  - The rows are distributed among slices based on the value of one column. This process is analogous to range partitioning in terms of this system [10].
- ALL Distribution
  - A copy of the entire table is stored in each slice in order to ensure that every row is collocated for each join. This causes slow function return on the data set [10].

3.3 Conclusion

After careful analysis of each option, there are a couple of properties between the three technologies that seem appealing. First, the column-oriented storage mentioned in the Redshift documentation seemed promising given the analytic procedures of our client. However, in 2013 Oracle announced that they we going to support column-oriented storage [12]. Since this feature was not isolated to Redshift alone, Redshift lost some of it's appeal over the Oracle database software the clients already have up and running. Second, was the ability to customize logical table partition designs. Both Oracle and PostgreSQL offered these features through different implementations. However, Redshift relies on the concept of data distribution styles of node slices. Though it is basically the same concept, Redshift has been developed for "ease-of-use" user experience and therefore does not offer as detailed customization as the other two database technologies. Out of the three considered database technologies, the winner is Oracle. The reasons behind this decision are based on Oracles customization and reliability. Though it is a paid service, our client already has a license and hardware setup in order to utilize the service, so the level of granularity in terms of system configuration is more appealing for the Oracle technology than any of the alternatives reviewed.

4 TOOLKITS

4.1 Purpose of a Toolkit

The main purpose of a database toolkit is to assist a Database Administrator (DBA) with analyzing the performance of queries made to a database. The different types of toolkits that can be used with Oracle Database 12c vary widely from PL/SQL queries that can be run directly in the database to complex, fully featured web applications. Likewise, the use cases for these toolkits can range from quick and easy query statement diagnostics all the way to automating the analysis and optimization of parameters within the database to a particular SQL query or multiple queries.

When analyzing queries, being able to effectively quantify and visualize multiple aspects of a query statement is important to a DBA. Some of the information gathered might include the time it took for a query to run, how much memory was allocated to a process handling a query, and the amount of disk input and output operations. This information can then be combined and analyzed in order to identify problematic areas within the database with a particular query or multiple queries.

For example, if you have a particular slow running query, you might use a toolkit to analyze what is happening when the query is being run. After analyzing the output of the toolkit you then find out that the way the database is allocating memory is suboptimal for the query thus resulting in a lot of swap space usage. With this knowledge you can then adjust parameters within the database to allocate more memory to the processes that handle queries.

4.2 Existing Toolkits

4.2.1 Oracle Enterprise Manager

Oracle Enterprise Manager (EM) is already included in the Enterprise edition of Oracle Database 12c. It is intended to be a one stop, all in one application that can be used for analyzing and optimizing query performance. Unlike other toolkits which are comprised of PL/SQL queries that can be executed in the database, EM is a full fledged web application that is packed with many features beyond analyzing query performance.

Some features that are included in EM are: the ability to manage and maintain several databases from one web application; automating repetitive tasks involved with maintaining databases; testing changes before rolling out to production; diagnosing performance problems and automated database tuning. Additional examples of what EM can do are it can determine if indexes will be helpful for performance of a particular table and analyzing sql query statement structure for potential performance issues.

One particularly useful feature of EM for analyzing query performance is a subcomponent called Active Session History (ASH). The ASH tool runs in the background and samples each active session within the database once per second. Each

sample for each session contains detailed info about what resources are being used by the session and how the session is using it. This is especially useful for identifying bottlenecks and other performance issues of running SQL queries.

4.2.2 SQLd360

SQLd360 is a tool that is written by Mauro Pagano, an ex-Oracle Database Engineer of about ten years, which analyzes SQL queries. Unlike EM, it will only analyze a single specified query and will not perform any database optimization for you. Instead, it is only intended for performance analysis of a query. SQLd360 is very similar to EMs ASH tool as it operates under the same underlying principle of periodically sampling the session that the query is running in to gather performance metrics. This information can be specified to be output in either html, text, csv, or graphical charts. One important difference is the SQLd360 is provided as a completely free tool while EMs ASH requires additional licensing from Oracle. This makes it particularly useful for users who do not have the required licensing but still need to obtain key metrics from session data. After the tool is finished running, it will dump all of the collected information into a zip file for later analysis.

4.2.3 Snapper

Snapper is another toolkit written by Tanel Poder, an Oracle Certified Master DBA, which also analyzes SQL queries. Similar to SQLd360, it is also provided completely free of use and does not require additional licensing from Oracle. It also operates similarly to SQLd360 in that it uses the databases session information to extract metrics information from running queries. It will also not provide any recommendations about how to optimize queries or database parameters. In addition to sampling data from query sessions, it also gathers data from the V$ and X$ tables within the database which both house extra information regarding query performance. Snapper combines this information together in order to provide a data rich visual representation of how exactly a query is running in the database.

4.3 Creating a Toolkit

Of the three toolkits mentioned above, it might sound like Oracles EM is the obvious way to go since it is very feature rich and is even capable of automatically optimizing SQL queries and database parameters for you. However, in our case we are still going to develop a toolkit of our own that will provide similar functionality. While avoiding reinventing the wheel is typically important when it comes to designing and creating software to prevent wasting time, in our case it makes sense for us to design a toolkit of our own that provides similar functionality to the three toolkits described above. The reason for this is, is that it will force us to become familiar with the inner workings of the database. While we could just use of the already existing toolkits to perform query performance analysis for us, we would probably not be as effective in modifying important database parameters when it comes to memory management, database table partition design, and parallelism. The expected outcome of developing our own toolkit is that not only will we have a custom made toolkit tailored to our needs but we should come out much more knowledgeable about Oracle 12c. This in turn should allow us to become more proficient in customizing the parameters of the database to increase query performance.

4.4 Conclusion

Add content here...

REFERENCES

[1] Oracle Company. *Query Optimizer Concepts*. [Online]. Available: https://docs.oracle.com/...
[2] Surajit Chaudhuri. *An Overview Of Query Optimization In Relational Systems*. [Online]. Available: http://web.stanford.edu/...
[3] Oracle Company. *How Parallel Execution Works*. [Online]. Available: https://docs.oracle.com/...
[4] Oracle Company. *Partitioning Concepts*. [Online]. Available: https://docs.oracle.com/...

[5] Postgresql Volunteers. *General Information*. [Online]. Available: https://www.postgresql.org/about/
[6] Postgresql Volunteers. *Parallel Query*. [Online]. Available: https://www.postgresql.org/docs/...
[7] Postgresql Volunteers. *Documentation: 9.1: Partitioning*. [Online]. Available: https://www.postgresql.org/docs/...
[8] Amazon. *Tuning Query Performance*. [Online]. Available: https://aws.amazon.com/redshift/
[9] Amazon. *Tuning Query Performance*. [Online]. Available: http://docs.aws.amazon.com/redshift/...
[10] Amazon. *Choosing A Data Distribution Style*. [Online]. Available: http://docs.aws.amazon.com/redshift/...
[11] Amazon. *Unsupported Postgresql Features*. [Online]. Available: http://docs.aws.amazon.com/redshift/...
[12] Alex Woodie. *Oracle Gives 12c Database a Column-Oriented Makeover*. [Online]. Available: https://www.datanami.com/...