**CS560 Final Project**

**Data Structure and processing performance Comparison**

**Spark SQL VS Oracle SQL**

Zhihua Zhang & Kaixiang Wang

# Introduction

A database is a comprehensive collection of data which is organized by fields, records, and files. To access the database conveniently, Database Management System (DBMS) is developed to provide an interface for the user to create, query, update and delete data in a database. Currently, the DBMS has been widely in various systems, like dynamic websites, accounting information systems, payroll systems, stock management systems (Bassil, 2012). One of the most popular DBMS is Oracle MySQL. It is a free, open-source, multithreaded, and multi-user SQL database management system which has more than 10 million installations. However, with the increasing growth of data, querying information quickly form Big Data is quite important and necessary. To achieve this goal, one query tools for Big Data, Spark SQL, is developed to work with big structured data, which provides a programming abstraction called DataFrame and can act as distributed SQL query engine. Therefore, to compare the performance of conventional DBMS and big data processing platform, in this project, a comparative analysis based on a performance benchmark was conducted to measure query performance (query processing time) between MySQL and Spark SQL.

# Background

This section discusses the history, versions, and features of Oracle MySQL and Spark SQL under test in this study.

## Oracle MySQL

Oracle MySQL is an open-source relational database management system (RDBMS). It was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. MySQL has numerous features, such as a broad subset of ANSI SQL 99, as well as extensions, Cross-platform support, Stored procedures, using a procedural language that closely adheres to SQL/PSM, Triggers, Cursors, Updatable views, ACID compliance when using InnoDB and NDB Cluster Storage Engines, SSL support, Query caching (Wikipedia, 2018b). MySQL has recently released its 8.0 version, but in this study, we used the MySQL 5.7.

## Spark SQL

Spark SQL is a component on top of Spark Core that introduced a data abstraction called DataFrames, which provides support for structured and semi-structured data. Spark SQL provides a domain-specific language (DSL) to manipulate DataFrames in Scala, Java, or Python. It also provides SQL language support, with command-line interfaces and ODBC/JDBC server. Many good features can be found in Spark SQL, such as integrated, unified data access, hive compatibility, standard connectivity, and scalability, which makes Spark SQL much efficient and popular. The Spark version used in this study is Spark 2.2.1 released on December 1st, 2017.

# Data and Working Procedure

The dataset was downloaded from kaggle (<https://www.kaggle.com>), which is a platform for predictive modeling and analytics competitions in which statisticians and data miners compete to produce the best models for predicting and describing the datasets uploaded by companies and users (Wikipedia, 2018a). The dataset is based on the Expedia search logs, including what customers searched for, how they interacted with search results (click/book), whether the search result was a travel package or not (Expedia, 2016). Table 1 presents the detail information about data fields. The total dataset is about 3.8 GB and is stored in the form of CSV file. To make a comprehensive comparison, we divided the dataset into different sizes, like 10 MB, 100MB, 500 MB, 1G, 2G, so that we can compare the performance in the different scale.

Table 1. Detail information about data fields

|  |  |  |
| --- | --- | --- |
| Column name | Description | Data type |
| date\_time | Timestamp | string |
| site\_name | ID of the Expedia point of sale (i.e. Expedia.com, Expedia.co.uk, ...) | int |
| posa\_continent | ID of continent associated with site\_name | int |
| user\_location\_country | The ID of the country the customer is located | int |
| user\_location\_region | The ID of the region the customer is located | int |
| user\_location\_city | The ID of the city the customer is located | int |
| orig\_destination\_distance | Distance between a hotel and a customer at the time of search. | double |
| user\_id | ID of user | int |
| is\_mobile | 1 when a user connected from a mobile device, 0 otherwise | tinyint |
| is\_package | 1 if the click/booking was generated as a part of a package (i.e. combined with a flight), 0 otherwise | int |
| channel | ID of a marketing channel | int |
| srch\_ci | Checkin date | string |
| srch\_co | Checkout date | string |
| srch\_adults\_cnt | The number of adults specified in the hotel room | int |
| srch\_children\_cnt | The number of (extra occupancy) children specified in the hotel room | int |
| srch\_rm\_cnt | The number of hotel rooms specified in the search | int |
| srch\_destination\_id | ID of the destination where the hotel search was performed | int |
| srch\_destination\_type\_id | Type of destination | int |
| hotel\_continent | Hotel continent | int |
| hotel\_country | Hotel country | int |
| hotel\_market | Hotel market | int |
| is\_booking | 1 if a booking, 0 if a click | tinyint |
| cnt | Numer of similar events in the context of the same user session | bigint |
| hotel\_cluster | ID of a hotel cluster | int |

Therefore, five data sets are used, the description of the datasets are presented in table 2. All the datasets have been loaded onto Oracle MySQL Server and Spark SQL to create a table in each system, so that different types of queries can be executed. Table 3 lists all the associate queries that are used for the 5 datasets, which are created based on the previous studies (Ahmed, Ahamed, Rafiq, & Rahim, 2017; Li & Zhou, 2015; Zhang, Liu, Lu, & Chen, 2017). Additionally, all the queries were executed in the same system (DELL, 64-bit Operating System, i7-6700 CPU @ 3.40 GHz, 16.00 GB Memory) and the running time was recorded for the corresponding query.

Table 2. Datasets description

|  |  |  |
| --- | --- | --- |
| Dataset | Size | Number of records |
| 1 | 10 MB | 94,000 |
| 2 | 100 MB | 940,000 |
| 3 | 500 MB | 4,650,000 |
| 4 | 1 GB | 9,300,000 |
| 5 | 2 GB | 18,600,000 |

Table 3. List of executed queries

|  |  |  |
| --- | --- | --- |
| Query | Type | SQL |
| Q1 | Query1 | SELECT \* FROM hotel WHERE srch\_ci = ‘2014-05-04’ AND srch\_co= ‘2014-05-07’; |
| Q2 | Query2 | SELECT \* FROM hotel WHERE srch\_ci in (‘2014-05-04’, ‘2014-05-07’, ‘2014-05-10’) |
| Q3 | Count | SELECT COUNT(\*) FROM hotel; |
| Q4 | Unique | SELECT DISTINCT user\_id FROM hotel; |
| Q5 | Group By | SELECT is\_mobile, COUNT(\*) FROM hotel GROUP BY is\_mobile; |
| Q6 | Order By | SELECT user\_id, orig\_destination\_distance FROM hotel WHERE srch\_ci = ‘2014-05-04’ AND srch\_co=’2014-05-07’ ORDER BY orig\_destination\_distance; |
| Q7 | Aggregate | SELECT user\_location\_country, AVG(orig\_destination\_distance) FROM hotel GROUP BY user\_location\_country |

# Results

Through the test data processed, results are shown in charts to compare the difference. In the charts, the X-axis represents the seven queries that are put in the tests. Y-axis represents the response time on needed based on different SQL on the same test system. Blue column represents MySQL and orange column represent Spark SQL. The detailed number is also listed in the table with the charts. Below are the charts represent five datasets in different sizes used in the test.

Chart 1: Query Response Time for MySQL and Spark SQL on 10M Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 10M |  | MySQL | Spark SQL |
| Time (s) | Q1 | 1.89 | 0.34 |
| Q2 | 1.9 | 0.09 |
| Q3 | 1.73 | 0.12 |
| Q4 | 1.78 | 0.76 |
| Q5 | 1.75 | 0.35 |
| Q6 | 1.86 | 0.12 |
| Q7 | 1.78 | 0.35 |

Chart 2: Query Response Time for MySQL and Spark SQL on 100M Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 100M |  | MySQL | Spark SQL |
| Time (s) | Q1 | 3.15 | 0.89 |
| Q2 | 3.46 | 0.05 |
| Q3 | 2.93 | 0.5 |
| Q4 | 3.29 | 1.51 |
| Q5 | 3.05 | 0.65 |
| Q6 | 2.93 | 0.54 |
| Q7 | 3.41 | 0.68 |

Chart 3: Query Response Time for MySQL and Spark SQL on 500M Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 500M |  | MySQL | Spark SQL |
| Time (s) | Q1 | 5.54 | 0.43 |
| Q2 | 5.17 | 0.04 |
| Q3 | 4.74 | 1.77 |
| Q4 | 19.09 | 3 |
| Q5 | 4.83 | 2.11 |
| Q6 | 4.72 | 2.07 |
| Q7 | 5.34 | 2.56 |

Chart 4: Query Response Time for MySQL and Spark SQL on 1G Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1G |  | MySQL | Spark SQL |
| Time (s) | Q1 | 8.18 | 0.57 |
| Q2 | 8.41 | 0.1 |
| Q3 | 7.35 | 3.49 |
| Q4 | 202.12 | 5.96 |
| Q5 | 7.54 | 3.9 |
| Q6 | 7.31 | 4.25 |
| Q7 | 8.83 | 4.73 |

Chart 5: Query Response Time for MySQL and Spark SQL on 2G Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 2G |  | MySQL | Spark SQL |
| Time (s) | Q1 | 20.47 | 1.59 |
| Q2 | 20.11 | 0.11 |
| Q3 | 17.38 | 6.8 |
| Q4 | 1080.82 | 9.98 |
| Q5 | 17.66 | 7.46 |
| Q6 | 22.23 | 8.59 |
| Q7 | 23.24 | 8.93 |

# Analysis

According to the result of tests, the response time needed by Spark SQL is always less than MySQL regardless the file size. As the file size goes larger, the difference is larger which Spark SQL is faster than MySQL.

There is more information is provided from this test result. The third query is much slower for MySQL as the file size goes larger. The count for all hotels takes longer than other queries for both SQL in the test. For MySQL, the increase of response time is way larger than others when the file size becomes larger. However, the difference is not that huge in Spark SQL. The third query is still the slowest in Spark SQL but the increase of response time is not too much than other queries. When the file size increase to 2G which is that largest data set in the test, the execution time of the third query is still similar to other queries on Spark SQL. This shows that the Spark SQL provided more stable execution time than MySQL.

Another result shown in this test is that the first and second query takes similar time on MySQL but a lot faster on Spark SQL compare to other queries. Chart 5 is can be rescaled to chart 6 show that more clearly. The first and second query have a similar response time and similar increase than other on MySQL. However, on Spark SQL, the execution time for the first two query is a lot faster than others. Especially the second query, There almost no increase time while the file size increase. The result only takes 0.11 seconds on 2G dataset while it takes 0.09 second on the 10M data set.

Chart 6: Query Response Time for MySQL and Spark SQL on 2G Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 2G |  | MySQL | Spark SQL |
| Time (s) | Q1 | 20.47 | 1.59 |
| Q2 | 20.11 | 0.11 |
| Q3 | 17.38 | 6.8 |
| Q4 | 1080.82 | 9.98 |
| Q5 | 17.66 | 7.46 |
| Q6 | 22.23 | 8.59 |
| Q7 | 23.24 | 8.93 |

# Conclusion

According to the test, the conclusion is that Spark SQL did a better performance and stability compare to MySQL. The overall response time on Spark SQL is faster and linearly increase as the file size increases. The Spark SQL also provide more stable execution than MySQL. All queries in this test case have closely response time without large increases when file size increases. The only exception is the first two queries is more efficient than others. For MySQL, the response time is slower and the third query increase dramatically while the file size increases. Other queries increase linearly as file size increase which is stable. However, the overall response time is likely twice as much as the Spark SQL.

# References

Ahmed, N., Ahamed, S., Rafiq, J. I., & Rahim, S. (2017). *Data processing in Hive vs. SQL server: A comparative analysis in the query performance.* Paper presented at the Engineering Technologies and Social Sciences (ICETSS), 2017 IEEE 3rd International Conference on.

Bassil, Y. (2012). A comparative study on the performance of the Top DBMS systems. *arXiv preprint arXiv:1205.2889*.

Expedia. (2016). Expedia Hotel Recommendations. Retrieved from <https://www.kaggle.com/c/expedia-hotel-recommendations/data>

Li, X., & Zhou, W. (2015). *Performance comparison of hive, impala and spark sql.* Paper presented at the Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2015 7th International Conference on.

Wikipedia. (2018a). Kaggle. Retrieved from <https://en.wikipedia.org/wiki/Kaggle>

Wikipedia. (2018b). MySQL. Retrieved from <https://en.wikipedia.org/wiki/MySQL>

Zhang, M., Liu, F., Lu, Y., & Chen, Z. (2017). *Workload Driven Comparison and Optimization of Hive and Spark SQL.* Paper presented at the Information Science and Control Engineering (ICISCE), 2017 4th International Conference on.