# Comparing Big Data Management Systems: Myria, Spark and Vertica

Shrainik Jain Aditya Vashishta Maaz Ahmad
{shrainik,adityav,maazsaf}@cs.washington.edu

## 1. Abstract

Over the past decade, improvements in data collection and storage technology has made big data prevalent in many scientific and business domains. As a response to the rapidly inflating needs for data management and analytics, the database community has developed numerous systems capable of efficiently storing and querying data at a very large scale. Different systems are often better suited to different workloads due to their unique design and specialized optimizations. Determining the best system for a given situation is not always obvious and can require considerable expertise. Furthermore, choosing the wrong system not only leads to suboptimal performance but can also inflict a significant cost in terms of money and effort expended. In our class project, we empirically compare and contrast three popular big data analytic systems and use the results of our experiments to make recommendations as to each systems appropriateness for the TPCH workload.

## 2. Background

At a high level, there are three classes of solutions available to the user for their data processing needs. First, we have the commercial big data management systems such as Vertica, IBM DB2, Oracle and Teradata. Typically developed by established companies, these systems are engineered by domain experts and offer very high levels of performance and support. Unfortunately, these systems are often too expensive since they require licensing on top of the hardware costs. The costs also typically scales as the amount of data handled or the amount of queries is increased. A more affordable alternative to such commercial systems are the several industry supported open-source systems such as Apache Spark, Hadoop, Impala, Apache Beam and TensorFlow. These systems do not require any licensing and can therefore be rather inexpensive. Without licensing restrictions, they can also be scaled to reach the desired performance levels. As they are used in many industry deployments, they tend to have stable releases and provide long term support. A yet another class of big data management system are those developed in the academia. Examples of such systems include Myria (by the University of Washington), H-Store (by the Massachusetts Institute of Technology) and Peloton (by the Carnegie Mellon University). The key difference between academic systems and industry supported systems is their experimental nature. While systems developed in the academia may lack the stability, resources or support of the ones developed by the industry, they often offer the

latest and greatest in terms of technology. For example, C-Store offered column oriented data storage well before commercial products such as Vertica or IBM DB2 were available.

For this class project, we selected one system from each of the three classes: namely Vertica, Spark and Myria. While Myria and Spark represent modern license-free distributed database systems with cloud offerings, Vertica is a highly optimized commercial distributed database system. Moreover, the underlying architecture of the three systems is also different. While Vertica is a column-store database engine, Myria is row-store since it uses PostgreSQL as the database engine. Spark is based on MapReduce paradigm and could be used either as row-store or column-store. There are differences in the use of distributed memory as well. While Spark performs in-memory analytics in a shared-nothing cluster, Myria has a shared-nothing distributed query processing engine where most operators process all data in-memory but some operations can be pushed into PostgreSQL that could spill data to disk.

In this class project, we aim to compare Myria, Spark, and Vertica from a perspective of a user who has a modern business oriented workload. While these systems are known to be highly performant on the workloads of their choice, we aim to compare them over a standard database benchmark (TPC-H). We first identify three dimensions on which we examine the three systems -- data ingestion time, query runtime, and usability. We then describe our experimental setup, assumptions, and methodology for usability evaluation. We then present our results of experiments and usability evaluation.

## 2. Comparison Metric

We compared Myria, Spark and Vertica on three metrics. First, we compared data ingestion time, which we define as time required to load data into the cluster and get to a state when we can run queries on it. Second, we compared performance (in terms of runtime) of the systems on a wide-range of queries containing joins, filters, and aggregates operations, and subqueries. Third, we compared the systems on the usability aspects. Since these systems also target users other than database experts, the usability of these systems could be an important distinction for non-expert big data users.

## 3. Experimental Setup and Methodology

To examine data ingestion time and query performance, we generated 10GB, 100GB and 300GB of uniform data using the TPC-H benchmark. TPC-H is a decision support benchmark that supports business oriented ad-hoc queries with a high degree of complexity, and examine large volumes of data to have broader industry-wide relevance. All data files were hosted on AWS S3.

To run the experiments, we used a 3 node cluster containing c4.4xlarge nodes on Amazon Web

Services. The nodes had 16 virtual cores of Intel Xeon E5-2666 v3, 30GB of RAM, and 500GB of EBS. We used a three node cluster since Vertica's free license only supports up to 3 nodes. For examining query performance, we removed five of the twenty-two queries generated by TPC-H benchmark since Myria does not support ORDER BY clause. Default configurations were used for all three systems, we did not attempt to tune the deployment to the target workload.

To conduct a usability evaluation of the three systems, we used a modified version of the NASA Task Load Index (TLX) workload assessment. By a group brainstorming session, we identified the following four dimensions that affect the usability of a big data management and analytics system: 1) setting up the cluster, 2) loading data in the cluster, 3) writing queries to analyze data, and 4) debugging query performance. For each of these four tasks, using a modified NASA TLX, we assess Mental Demand (i.e. How mentally demanding was the task?), Performance (i.e. How successful you were in accomplishing the task?), Effort (i.e. How hard you have to work to accomplish your level of performance on the task?) and Frustration (i.e. How insecure, discouraged, irritated, stressed or annoyed were you because of the task?) on a 7-point scale. To examine the usefulness of these systems, we also asked some subjective questions. We incorporated these questions as a survey and sent it to the class Slack channel. The survey participants could respond to the questions for any subset of the three systems depending on their prior experience and comfort with the systems. We now present the results of our experiments and evaluations.

## 4. Results

In this section, we present the results of our experiments.

**Data Ingestion**

Our first goal was to measure the data ingestion performance for each of the three systems. For our experiments, we define data ingestion as the time required to get to the point where we can run queries about the data. For Vertica and Myria, this meant loading the data into the database from S3. While Spark allows executing queries directly on the data source files (therefore zero ingestion cost), to have a more meaningful comparison we used Spark to load data into persistent tables by loading the data into the HiveMetastore.

Figure 1 illustrates the results of our experiments. Spark demonstrated the best overall ingestion speeds across all datasets, with Myria being the slowest. Since vertica performs more pre-processing than Spark (in a bid to improve query performance), it lagged behind in ingestion speeds. For Myria, we used the default LOAD operator which is not parallel. Myria also offers a parallel loading operator which is likely to improve the performance significantly. Unfortunately, we were not able to put this to test.

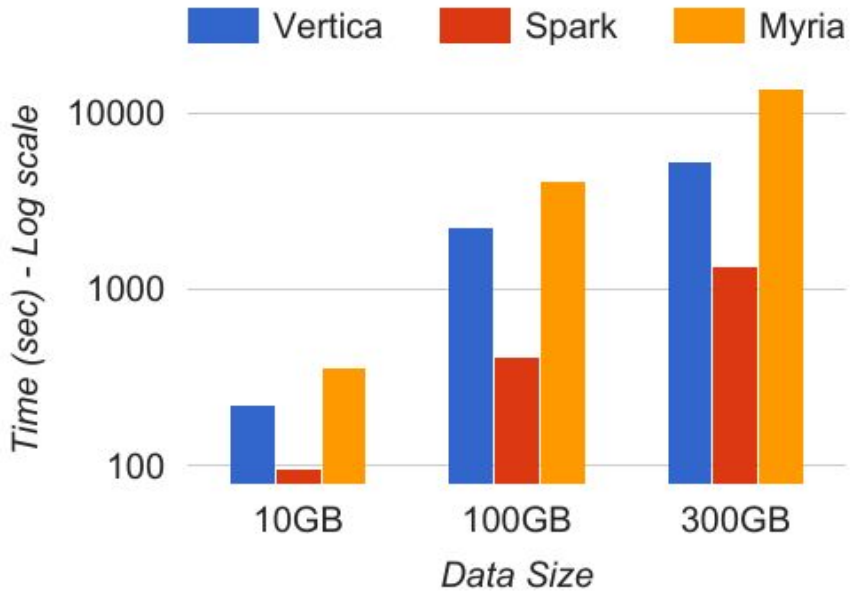


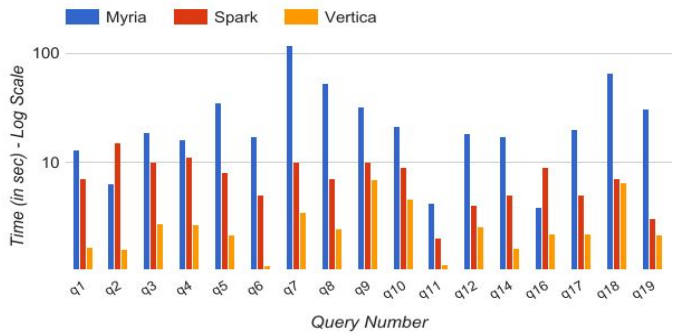

**Figure 1.** The figure presents the time taken (in seconds) for each of the three systems to ingest three TPCH datasets.

**Query performance**

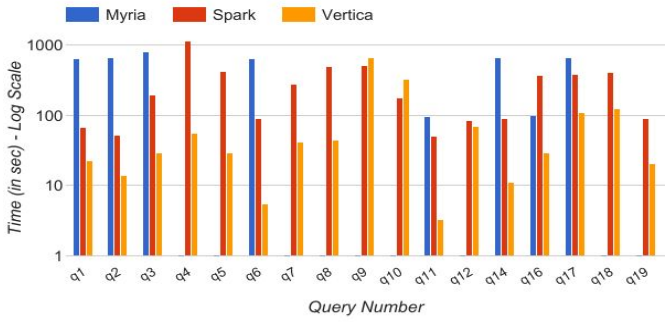
Data ingestion performance, while important (esp for running queries on unstructured raw data), paints an incomplete picture. Some systems sacrifice having shorter ingestion times for much better query performance. In this section, we evaluate the time it takes each system to answer a set of standard queries for each dataset, assuming the data has already been loaded into the system from S3.

Figure 2 presents the outcome of our experiments for 10GB, 100GB and 300GB datasets. Overall, Vertica was the clear winner in terms of performance, with Spark second best. This was rather unsurprising due to Vertica's column oriented storage. Furthermore, since both Spark and Myria are much more memory intensive systems, a three node cluster proved insufficient to handle the larger datasets (even causing some OOM errors in Myria).

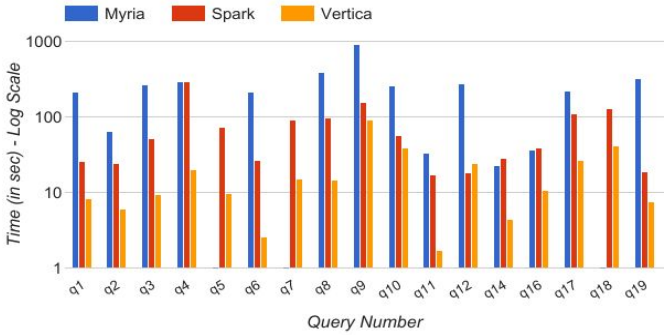
Since unlike Vertica, Spark and Myria do not require licenses, scaling these deployments was inexpensive. To get a sense of how much resources these systems require to match Vertica's performance, we executed all queries again using a 10 node Spark cluster. Figure 2 (d) compares of a 10 Node spark cluster's performance to a 3 Node Vertica cluster's performance on a 100GB dataset. Spark not only matches, but also outperforms Vertica. This suggests that while Vertica may offer the best performance for a given set of resources, it may not offer the best value for money.



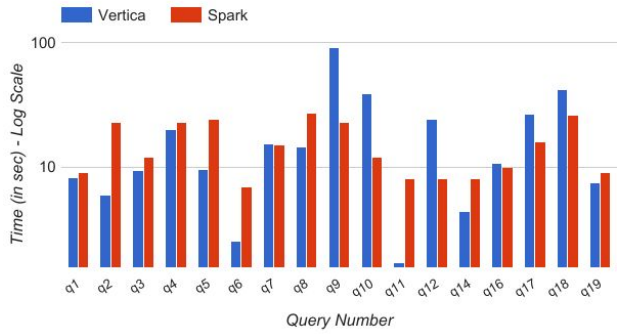

(a) 10GB dataset




(b) 100GB dataset




(a) 300GB dataset




(d) 10 Nodes Spark vs 3 Nodes Vertica

**Figure 2.** The figure presents the time taken (in seconds) for each of the three systems to execute a set of 17 TPCH datasets.

**Usability Study**

Five users completed the usability study. All of them used the three systems. However, only two users stated to have a reasonable understanding of all three systems, one user stated to have a reasonable understanding of Spark and Vertica, and the rest of them considered themselves as novice users. Since only five users finished the survey, we were unable to run significance tests on the usability scores. We also request readers to take the findings in the subsection with a grain of salt.

Figure 3 depicts the scores given to Myria, Spark and Vertica on a 7-point scale on the usability of setting-up the cluster, loading data, writing queries and debugging query performance. It is evident from Figure 3-A that Myria is the most usable system when it comes to setting up the cluster since it resulted in the least mental demand, highest performance. Spark is a close second since it yielded lower effort and frustration. Similarly, Figure 3-B depicts that Myria scored lowest in mental demand and frustration, and highest in performance, suggesting that it is most usable to load data in the clusters. This is not surprising since Myria is designed to be easy to use

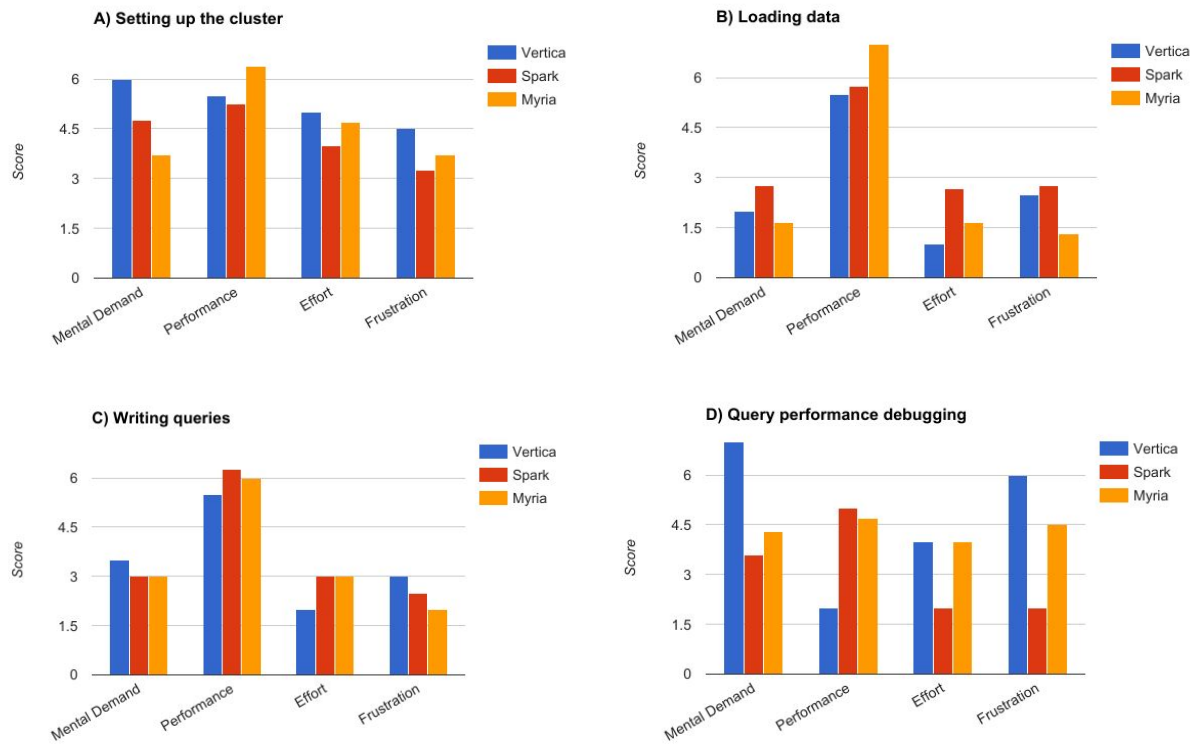



**Figure 3.** The figure presents a usability evaluation of Vertica, Spark and Myria on NASA TLX usability parameters on four system dimensions: A) Setting up the cluster, B) Loading data, C) Writing queries, and D) Query performance debugging.

and it is quite easy to load data using Myria Web interface. The usability scores for measuring the ease of writing queries (see Figure 3-C) showed a mixed picture. All three systems appeared comparable in terms of the ease of writing queries. This is also expected since all three systems support SQL or its variant (like MyriaL). Finally, Spark dominated the other two systems on query performance debugging (see Figure 3-D); it had the lowest score for mental demand, effort and frustration, and highest score for performance. This could be attributed to excellent visualizations on query plan and query execution provided by Spark. Giving an equal weight score to the four factors, our analysis reveals that Myria has lowest scores for mental demand and frustration, and highest score for performance, making it the most usable system among the three systems.

We also asked users, which system they prefer and why. Two participants preferred Myria over the others, primarily because of its simple user interface while two others preferred Spark. A participant who preferred Myria stated:

*"Myria - the web UI was nice and simple. Granted, all I used it for was the in class demo. Spark on Databricks is also easy to use - but setting up Spark from scratch is hard."*

There are some caveats to our usability evaluation. First, the number of participants who completed the survey is too less to generalize the findings. Second, we did not factor the self-reported level of expertise of participants in their scores. Third, there are several ways to access these systems. For instance, Myria could be used through Myria Web and also through Jupyter Notebooks. Similarly, the setup of Spark clusters is very simple using Databricks or using pre-configured AWS EC2 machines, but quite challenging when setting it up from scratch. In future iterations of the survey, we will attempt to overcome these weaknesses by sending the survey to large audiences, factoring their level of expertise in their scores, and including specific questions to tease apart different ways in which these systems could be used.

## 5. Conclusion

In this project, we tested three popular big data management systems (Spark, Myria and Vertica) on the TPCH query workload to see how they compare. Based on our results, Vertica is the fastest system and offers the best query performance. Spark on the other hand offers the best performance for value, especially for unstructured data due to the low ingestion time. Myria offers the most usability and is most suitable for non-expert users. Comparing such systems is a challenging problem and can involve bias caused by different levels of expertise for different systems and the hardware choice. An interesting project for the future would be to test the systems for different workloads and to test the systems on different cluster sizes and types for a more complete analysis.