Big Data

Dan Grogan
10/20/2016

Hive – A Petabyte Scale Data Warehouse Using Hadoop

- Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Antony, Hao Liu and Raghotham Murthy

A Comparison of Approaches to Large-Scale Data Analysis

- Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden and Michael Stonebraker

One Size Fits All: An Idea Whose Time Has Come And Gone

- Michael Stonebraker

Main Idea Of The Paper I Chose

The main idea of this paper focuses on an open-source data warehouse solution called Hive. Hive was built on top of Hadoop, an open-source MapReduce implementation used to store and process large data sets on commodity hardware. Hadoop however is a very low level programming model and requires developers to write custom programs which are hard to maintain and reuse. This is where Hive comes into use. Hive supports queries expressed in an SQL-like declarative language (creatively named HiveQL) which are compiled into MapReduce jobs that are executed through Hadoop. Hive also has other features, such as the ability to plug in custom MapReduce scripts into queries and a system catalog, the Metastore, that contains schemas and statistics of the data.

How It’s Implemented

Hive structures data into typical database concepts such as tables, columns, rows and partitions. It also supports all major primitive types and even some complex types, such as maps, lists and structs that can be nested arbitrarily in order to create even more complex types. Hive stores data in tables where each table consists of a number of rows and columns, much like a typical relational database. HiveQL allows for anyone who is familiar with SQL to create queries in order to retrieve data from the database.

My Analysis Of The Idea And It’s Implementation

To me it seems like Hive, although not yet a finished product, will make working with big data a lot easier. Instead of getting used to some other program that works with Hadoop (and the syntax that goes along with it), Hive allows anyone with basic SQL knowledge to create queries and manage a database through Hadoop. As they do say in the paper, Hive is still a work in progress and it only recognizes a subset of SQL queries as valid queries as of the time this paper was published, but they are continuously working on it to make that subset larger. This project seems like a good idea with an implementation that works well with the general population that will be working with Hive.

Main Idea Of Comparison Paper

The paper talks about two current paradigms for large scale data analysis. The two paradigms that are compared are MapReduce and Parallel DBMS. MapReduce uses concepts that have been seen in Parallel DBMS for around 20 years. One of the qualities that makes MapReduce attractive is its simplicity. It uses two functions, Map and Reduce. Parallel DBMS has been around since the late 1980s. The system supports standard relational tables and SQL and is stored on multiple machines in order to store all the data. When compared, the authors noticed that while the MapReduce system was considerably faster, the Parallel DBMS system’s performance was better.

How These Ideas Are Implemented

As mentioned in the previous slide, Parallel DBMS systems require data to fall into the relational database paradigm of rows and columns. In contrast, the MapReduce model does not require the data files to fit the relational model. MapReduce users are free to structure their data however they please, including having no structure at all. MapReduce does not have any built in indexes like Parallel DBMS does, so MapReduce users must implement any indexes that they want to speed up access to the data inside of their application. In order to access data in MapReduce systems, you must present an algorithm as opposed to an SQL query for the relational Parallel DBMS model.

My Analysis On Comparison Paper

I feel as though the authors did a good job testing each paradigm. They used the newer versions of each system and thoroughly tested each system’s performance. There is still a lot to improve in each system. It doesn’t surprise me that the system that uses relational tables is slower, as it is harder to go compare and retrieve data when you are working with larger databases. Relational databases require you to go through the entire system in order to make comparisons and retrieve data, which is not ideal for larger databases. I can see reasons why someone would like either of the two systems, but they are both far from perfect. As the authors pointed out, we can consider what works between each of these paradigms and use those things to help us create an even more efficient system that works well with large data.

Comparisons Of The Two Papers

I thought the two papers were sort of different papers in a sense that they were coming from two different points of view. The comparison paper seemed to focus on a much larger picture and that was the general system or paradigm in which you should use when working with big data. The Hive paper only talks about one of those systems, however in my opinion makes that construct seem like it'd be easier or better than what they described that construct as in the comparison paper. It seems that Hive could have helped to fix some problems that the comparison paper had originally had with MapReduce systems.

Main Idea of Stonebraker Talk

There are multiple ways to approach big data and relational data was, for a long time, considered to be a “one size fits all” type of model. What has been realized is that the relational model may not be the best structure for larger databases, in fact it may be one of the worst. One of his main points was that there is much to research in DBMS and that this is a great time to get involved into the research or to get involved in developing a new structure that will work well when dealing with big data.

Advantages And Disadvantages

One advantage of Hive is that it allows people who are familiar with SQL, which is very popular among people in the field, to be able to come in and create queries in order to retrieve data. One downside to this is that Hive uses a relational structure which has proven to be less effective with the more data you have.