Native SQL Access to Hadoop Data

K Surya Ram Prasad[1]
Assistant Professor
DNR College of Engineering &Technology

P Srikanth[2]
Assistant Professor
Shri Vishnu Engineering College for Women.

***Abstract*:- The NoSQL movement that has been happening over the past few years has taught two important lessons: a) Alternatives to relational databases can be a great help in solving a variety of problems and b) SQL isn't going anywhere. In fact, the NoSQL movement is now being rebranded as New-SQL, as in, "Here's a new technology where you can use SQL!" Even though we've seen a tremendous amount of innovation in the information management field — technologies are now available that can store graphs, documents, and key/value pairs at a massive scale — the IT market is still demanding SQL support for all of it. Hadoop is no exception, and a number of companies are investing heavily to drive open source projects and proprietary solutions for SQL access to Hadoop data.**

Keywords: NoSQL, Hive,Map Reduce, Big SQL, HDFS, DrillBit.

INTRODUCTION:

The IT industry has had 40 years of experience with SQL, since it was first developed by IBM in the early 1970s. With the increase in the adoption of relational databases in the 1980s, SQL has since become a standard skill for most IT professionals. You can easily see why SQL has been so successful: It's relatively easy to learn, and SQL queries are quite readable. This ease can be traced back to a core design point in SQL — the fact that it's a declarative language, as opposed to an imperativelanguage. For a language to be declarative means that your queries deal only with the nature of the data being requested. In other words, all you indicate in SQL is what information you want back from the system, not how to get it. In contrast, with an imperative language (C, for example, or Java, or Python) your code consists of instructions where you define the actions you need the system to execute. When talking about how Hadoop can complement the data warehouse, it's clear that organizations will store structured data in Hadoop. And as a result, they'll run some of their existing application logic against Hadoop. No one wants to pay for applications to be rewritten, so a SQL interface is highly desirable. With the development of SQL interfaces to Hadoop data, an interesting trend is that commercial business analytics and data management tools are almost all jumping on the Hadoop bandwagon, including business intelligence reporting; statistical packages; Extract, Transform, and Load frameworks (ETL); and a variety of other tools. In most cases, the interface to the Hadoop data is Hive.

SQL ACCESS:

SQL accessrelying on a few basic assumptions:

- **Language Standards:**The most important standard, of course, entails the language itself. Many "SQL-like" solutions exist, though they usually don't measure up in certain fundamental ways that would prevent even typical SQL statements from working. The American National Standards Institute (ANSI) established SQL as an official technical standard, and the IT industry accepts the ANSI SQL-92 standard as representing the benchmark for basic SQL compliance.
- **Drivers:**Another key component in a SQL access solution is the driver — the interface for applications to connect and exchange data with the data store. Without a driver, there's no SQL interface for any client applications or tools to connect to for the submission of SQL queries. As such, any SQL on Hadoop solution has to have JDBC and ODBC drivers at the very least, because they're the most commonly used database interface technologies.
- **Real-Time Access:**Until Hadoop 2, MapReduce-based execution was the only available option for analytics against data stored in Hadoop. For relatively simple queries involving a full scan of data in a table, Hadoop was quite fast as compared to a traditional relational database. Keep in mind that this is a batch analysis use case, where fast can mean hours, depending on how much data is involved. But when it came to more complex queries, involving subsets of data, Hadoop did not do well. MapReduce is a batch processing framework, so achieving high performance for real-time queries before Hadoop 2 was architecturally impossible.
- **Mutable Data:**A common question in many discussions around SQL support on Hadoop is "Can we use INSERT, UPDATE, and DELETE statements, as we would be able to do in a typical relational database?" For now, the answer is no, which reflects the nature of HDFS — it's focused on large, immutable files. At the time of this writing, technologies such as Hive offer read-only access to these files. Regardless, work is ongoing in the Hive Apache project to enable INSERT, UPDATE, and DELETE statements.

IBM BIG SQL:

IBM has a long history of working with SQL and database technology, as the introduction to this chapter makes clear. In keeping with this history, IBM's solution for SQL on Hadoop leverages components from its relational database technologies that are ported to run on Hadoop.

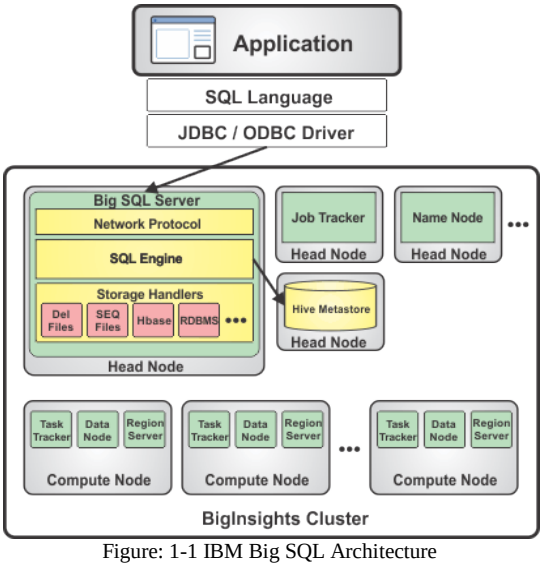

Figure: 1-1 IBM Big SQL Architecture

Big SQL supports JDBC and ODBC client access from both Linux and Windows platforms. That client access means that you can take advantage of your SQL skills, any SQL-based business intelligence applications, and query or reporting tools to query InfoSphere BigInsights data.
Big SQL is not a replacement for relational database management systems (RDBMS) technology. It is designed to compliment and leverage the Hadoop-based infrastructure of InfoSphere BigInsights. Some features common to database management systems are not present in Big SQL. Some Big SQL features are not common to most relational database management systems. Big SQL supports querying data, but INSERT, UPDATE and DELETE statements are not supported.
However, Big SQL tables can contain columns of complex data types, such as flat rows. Big SQL also supports several underlying storage mechanisms stored on either Hadoop Distributed File System (HDFS) or IBM General Parallel File System (GPFS™ FPO), including the following:

- Delimited files (such as comma-separated values)
- Hive tables in sequence file format and RCFile format
- HBase tables

Data administrators can use Big SQL to create tables over existing data using the CREATE EXTERNAL TABLE command. They can create new tables using the CREATE TABLE command and load data to it using the LOAD command. They can also create a table and load data from a query using the CREATE TABLE <name> AS <query> statement. Application developers can use the Standard SQL syntax of Big SQL, along with the SQL extensions that are specific to InfoSphere BigInsights to take advantage of the Hadoop-based technologies. The Big SQL language provides you with familiar SQL syntax to write queries to accomplish joins, unions, grouping, windowing functions, common table expressions.

APACHE HIVE:

Apache Hive is indisputably the most widespread data query interface in the Hadoop community. Originally, the design goals for Hive were not for full SQL compatibility and high performance, but were to provide an easy, somewhat familiar interface for developers needing to issue batch queries against Hadoop. This rather piecemeal approach no longer works, so the demand grows for real SQL support and good performance. Hortonworks responded to this demand by creating the Stinger project, where it invested its developer resources in improving Hive to be faster, to scale at a petabyte level, and to be more compliant to SQL standards. This work was to be delivered in three phases.
In Phases 1 and 2, you saw a number of optimizations for how queries were processed as well as added support for traditional SQL data types; the addition of the ORCFile format for more efficient processing and storage; and integration with YARN for better performance. In Phase 3, the truly significant evolutions take place, which decouple Hive from MapReduce.

MASSIVELY PARALLEL PROCESSING DATABASES:

To provide a better understanding of the SQL on Hadoop alternatives to Hive it would be helpful to provide a primer on massively parallel processing (MPP) databases first. Apache Hive is layered on top of the Hadoop Distributed File System (HDFS) and the MapReduce system and presents an SQL-like programming interface to your data (HiveQL, to be precise). This combination of Hadoop technologies deployed on a cluster is similar to MPP databases that have existed for a while in the IT marketplace. MPP databases usually provide an SQL interface and a relational database management system (RDBMS) running on a cluster of servers networked together by a high-speed interconnect. The following figure shows the components of an RDBMS that are typically included in the SQL on- Hadoop solutions.

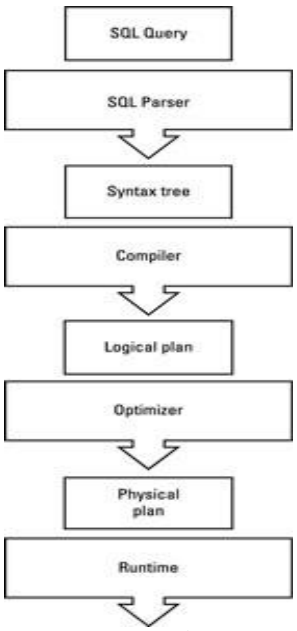

Figure 1-2: Components of RDBMS in MPP

The figure 1-2 shows the flow of a query as it's processed by an RDBMS engine. First, the query text is parsed and understood. Then the syntax tree for the query is compiled into a logical execution plan, which is then optimized to form the final physical execution plan, which is then executed by the runtime. For many of the SQL-on-Hadoop solutions, we're seeing similar components being deployed in Hadoop.

APACHE DRILL:

Drill is a candidate project in the Apache incubator. We don't mean that Apache Drill is especially sickly, though. The Apache Software Foundation (ASF) candidate technologies all begin as incubator projects before becoming official ASF technologies. The performance goal for Drill is to enable SQL queries against a petabyte or more of data distributed across 10,000-plus servers.

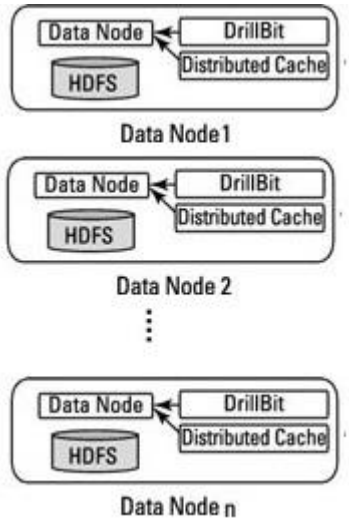

Figure 1-3: Apache Drill Architecture

The following figure 1-3 states that the key to the Drill architecture are the DrillBit servers deployed on each data node. Note that each server includes a query parser, compiler, optimizer, and runtime, but there is a master DrillBit server nominated by Zookeeper servers, which oversees the execution of the queries and looks after the task of pulling together the interim result sets into a single set of output.

CONCLUSION:

Hadoop is often thought of as the one-size-fits-all solution for big data processing problems, the project is limited in its ability to manage large-scale graph processing, stream processing, and scalable processing of structured data. Big SQL, a massively parallel processing SQL engine that is optimized for processing large-scale structured data. We can observe how it compares to other systems that were recently introduced to improve the efficiency of the Hadoop framework for processing large-scale structured data.

REFERENCES:

[1] Apache Hadoop. http://hadoop.apache.org/
[2] Hadoop - dummies - Dummies.com www.dummies.com/programming/big data/hadoop/
[3] P. H. Carns, W. B. Ligon III, R. B. Ross, and R. Thakur. "PVFS: A parallel file system for Linux clusters," in Proc. of 4th Annual Linux Showcase and Conference, 2000, pp. 317–327.
[4] J. Dean, S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," In Proc. of the 6th Symposium on Operating Systems Design and Implementation, San Francisco CA, Dec. 2004.
[5] A. Gates, O. Natkovich, S. Chopra, P. Kamath, S. Narayanam, C. Olston, B. Reed, S. Srinivasan, U. Srivastava. "Building a High-Level Dataflow System on top of MapReduce: The Pig Experience," In Proc. of Very Large Data Bases, vol 2 no. 2, 2009, pp. 1414–1425.
[6] O. O'Malley, A. C. Murthy. Hadoop Sorts a Petabyte in 16.25 Hours and a Terabyte in 62 Seconds. May 2009.
[7] S.Vikram Phaneendra & E.Madhusudhan Reddy "Big Data-solutions for RDBMS problems- A survey" In 12th IEEE/IFIP Network Operations & Management Symposium (NOMS 2010) (Osaka, Japan, Apr 19{23 2013).
[8] Jimmy Lin "MapReduce Is Good Enough?" The control project. IEEE Computer 32 (2013).
[9] Ahmed Eldawy, Mohamed F. Mokbel "A Demonstration of SpatialHadoop:An Efficient MapReduce Framework for Spatial Data" Proceedings of the VLDB Endowment, Vol. 6, No. 12 Copyright 2013 VLDB Endowment 21508097/13/10.
[10] Mrigank Mridul, Akashdeep Khajuria, Snehasish Dutta, Kumar N " Analysis of Bidgata using Apache Hadoop and Map Reduce" Volume 4, Issue 5, May 2014" 27