Big SQL 3.0 Functionality:

A comprehensive approach to SQL-on-Hadoop

Anjali Khatri
IBM USA
Big Data Specialist
Durham, USA

Maria N. Schewenger
IBM USA
Program Manager
Atlanta, USA

Neal Finkelstein
IBM USA
Big Data Portfolio Specialist
Piscataway, USA

Abstract- **Undoubtedly, Apache Hadoop™ (Hadoop) has received huge recognition in the Big Data market space because of the great flexibility in processing heterogeneous data structures. However, for adding greater value, it is critical for Hadoop to integrate with various analytical tools that are widely used by different enterprises. Most of these tools utilize standard Structure Query Language (SQL) and SQL procedural functionality. It is also important to allow the enterprises to utilize their existing SQL expertise, for which they have been building skills for years.**

In the last few years, this notion brought to life multiple SQL-on-Hadoop solutions developed by different Apache Hadoop distribution vendors. The purpose of IBM InfoSphere® BigInsights™ (BigInsights) - Big SQL 3.0 is to provide its users with a SQL interface built upon the Hadoop infrastructure that is consistent with the SQL interfaces they are using in the relational database world. Big SQL 3.0 is a significant enhancement to the SQL-on-Hadoop technologies as it eliminates the limitations imposed by running SQL queries as MapReduce jobs.

Big SQL 3.0 achieves several important objectives including comprehensive support of ANSI SQL 2011, application integration and portability, query federation (SQL integration between Hadoop and non–Hadoop data stores), and enterprise capabilities such as performance, security, monitoring. This article introduces the Big SQL 3.0 functionality as part of the BigInsights platform and provides examples of its capabilities, usage, and effectiveness.

I. INTRODUCTION

Big Data is derived from multiple sources. It involves not only traditional data, but all paradigms of structured, unstructured, or semi-structured data that are continuously growing at a significant rate. For instance, machine-derived data multiplies quickly and contains rich, diverse content that needs to be discovered, parsed, and analyzed. Another example, human-generated data from social media is highly textual, and the valuable insight is often overloaded with many possible meanings. Trying to extract meaningful insight from such data sources quickly and easily is challenging. Many professionals believe that the Hadoop technology is the best answer to those challenges. [5]

Hadoop is an open source platform that processes large data sets across a cluster using computing models, which many vendors try to extend in search of robust additional functionality.[3] BigInsights is IBM's approach to enrich open source Hadoop with enterprise class capabilities. The latest release of BigInsights - 3.0, provides a single platform to manage all data types, and obtain insights by integrating the usual silos of data.[2] With such integration, BigInsights is a viable platform for storage, analysis and visualization of a wide variety of data sources.[5] Several new features of BigInsights 3.0 concentrate on integration and workload optimizations that bring improvements for storing, processing and querying data. Big SQL 3.0 is one of those features.

BigInsights – Big SQL 3.0 is a SQL-on-Hadoop solution that breaks the dependency from the Hadoop's traditional MapReduce (MR) framework by implementing the database engine of IBM® DB2® for Linux, UNIX and Windows (DB2 LUW).[1] on top of the Hadoop Distributed File System (HDFS or DFS). It delivers a new, more comprehensive way to rich SQL support that includes enterprise level performance, security, management and monitoring - features that are traditionally found in the DB2 LUW and other relational database management systems. (Fig. 1)

Big SQL 3.0 runs distributed DB2 LUW database processes directly on the data nodes of the Hadoop cluster.[11] The data storage does not change since the data remains in the DFS. The Big SQL 3.0 tables are essentially logical views of the existing data. It also provides an easy and well known programming language for analysis of data in Hadoop.

The idea of SQL-on-Hadoop is not new to the industry. In addition to the open source Apache Hive, several vendors like Cloudera Impala®, Teradata-Aster®, Presto®, Pivotol Hawq®, Hadapt®, Stinger® and others are present in the SQL-on-Hadoop landscape today.[1] With SQL querying, the vendors try to extend the traditional MapReduce functionality built for

large scale and complex data processing. However, up until now, it has been especially difficult to provide high performance, comprehensive ANSI SQL 2011 compliance and enterprise level administrative support (monitoring, security) that executes over the heterogeneous Hadoop ecosystem. [1]

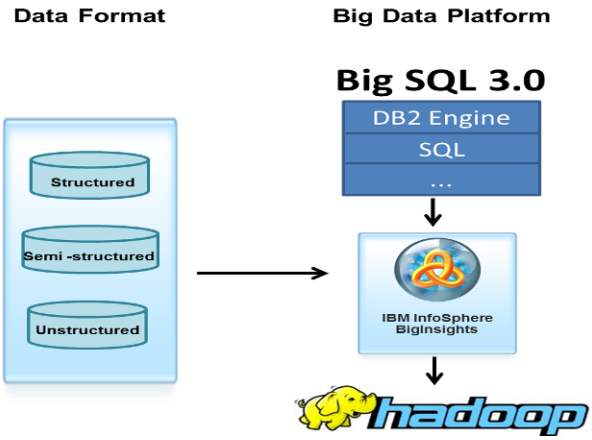

Fig. 1. Logical view of Hadoop ecosystem with Big SQL 3.0

The rest of the paper is organized as follows: Section II describes the functionality of BigInsights – Big SQL 3.0; Section III concludes the paper with a brief evaluation summary of the business and technical values delivered by Big SQL 3.0 SQL.

II. Overview of the Big SQL 3.0 Functionality

Big SQL 3.0 achieves comprehensive SQL capability by leveraging IBM DB2's relational SQL expertise and runtime engine in a shared-nothing, low-latency, parallel processing database architecture.[1] It has the advantage to deploy directly to the Hadoop cluster data nodes and accesses the underlying data natively while still allowing the data to remain in its original format on the HDFS.[3] It provides a logical view of the data and ability to access custom data storage formats through a Java I/O module.

This article provides an overview of five main Big SQL 3.0 functionality features that run upon the Hadoop cluster: rich SQL language support, query federation, application portability and integration, performance and enterprise capabilities.

A. Rich SQL language Support

Big SQL 3.0 has rich SQL support that leverages IBM's SQL capability layer to provide comprehensive ANSI SQL 2011 coverage.[17] It stretches over wide variety of traditional SQL functionality that provides support for: variety of data types (tinyint, smallint, int, bigint, boolean, float, double, string, varchar, and others); standard APIs and common client drivers (Java Database Connectivity (JDBC) or Open Database Connectivity (ODBC)); procedural language structures such as DB2 SQL PL or user defined functions in Java or C++. It has built-in statistical, analytical, and business intelligence functions. The Big SQL3.0 language is designed for large-scale analytic aggregation and supports some of the main OLAP functions like LEAD, LAG, RANK, DENSE_RANK, CORRELATION, STDDEV, REGR_AVGX, and others. Big SQL 3.0 is equipped with an extensive library of more than 250 built-in functions and 25 aggregates.[1]

Big SQL 3.0 is a simple, but comprehensive way to apply SQL to existing data processed by Hadoop.[8] It provides a logical view of the data that uses Hive Catalog (HCatalog) for table definitions, metadata, location, storage format and encoding of input files.[15] When creating a table, HCatalog definition and Hive APIs get stored in hive/warehouse directory within its designated schema on HDFS. Figure 2 below shows how the files for the table GOSALES_DW. PRODUCT_DIM are stored in the hive subdirectory of the BigInsights directory. This is a view from the DFS files tab of the BigInsights Web console.

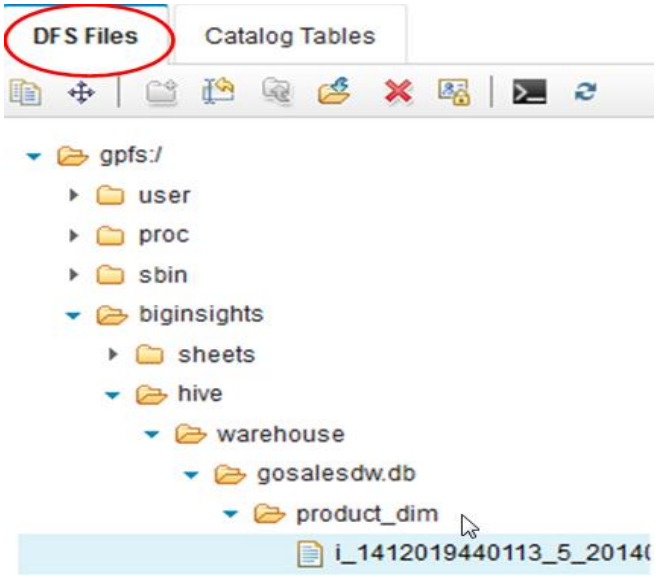

Fig. 2a. Big SQL 3.0 tables placement - DFS Files tab view

From the Catalog Tables tab, the same table appears in the "gosalesdw" schema as show in Fig. 2b.

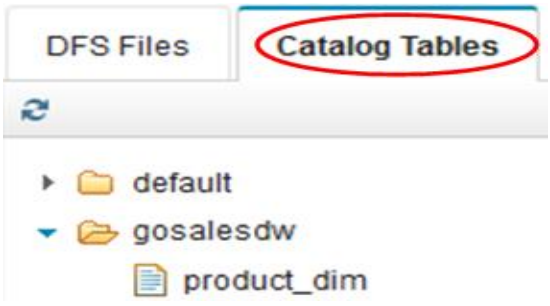

Fig. 2b. Big SQL 3.0 tables placement – Catalog Tables tab view

Hive's metastore catalogs table definitions could be read and/or write, and they are shared across the Hadoop ecosystem - they can be accessed by other components (such as Pig, Sqoop, etc.) throughout the Hadoop cluster.[2] The BigInsights Web console also provides a view of the table definition and the data inside of the table. (Fig. 2c)

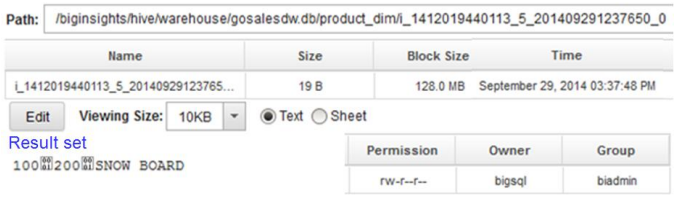

Fig. 2c. Big SQL 3.0 tables definition in the BigInsights Web console

The data inserted in the Big SQL 3.0 tables can be also displayed in tabular format in the BigInsights Web console as shown in Figure 2d below.

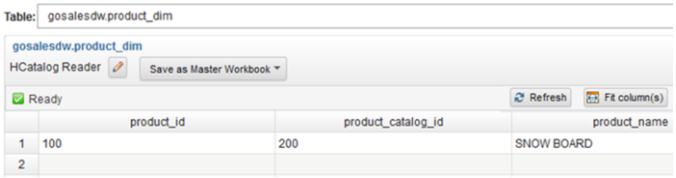

Fig. 2d. Big SQL 3.0 tables – row inserted in the BigInsights Web console

To facilitate query execution, HCatalog stores some of its metadata locally on each node. When in the parallel processing mode, the Big SQL server distributes the query to each data node. Each data node executes the query and generates a result set from the data stored within that specific node of the cluster.[4] The query execution also stores intermediate data into the Big SQL 3.0 engine memory for any future or additional processing. (See Fig. 3)

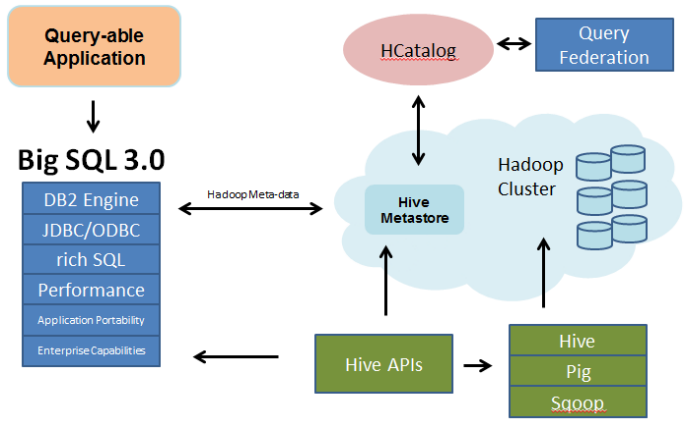

Fig. 3. Hadoop components interaction with Big SQL 3.0

Although the rich SQL support is the essence of the Big SQL 3.0, there are other very important features that compliment and raise the value of the SQL functionality.

B. Query Federation

Query federation is an advanced integration capability that allows sending of distributed SQL requests to multiple different data sources with a single SQL statement. These data sources include different relational database management systems like DB2, Teradata, Oracle®, Netezza®. Big SQL federation collects metadata and statistics from the remote source when federated objects are created. As a result, SQL sent to each source is optimized based on the capabilities and statistical profiles of the federated sources.
Since the federated data will remain in the source relational database, it not necessary to duplicate this data by transferring it locally to the nodes of the Hadoop cluster. Using federation makes it appear as though all the data is in a single database (not distributed between structured and unstructured sources). Using federation can save time and resources.

C. Application Portability & Integration

Within BigInsights – Big SQL 3.0, application portability and integration allows both data sharing and code sharing amongst the Hadoop ecosystem and external applications such as BI tools, and custom code that uses the SQL language.[7] SQL applications developed to use a relational database, can be easily ported to use Big SQL 3.0 and data in Hadoop. In some meaning, this integration is the ultimate driving force behind the Big SQL 3.0 feature.[16] Another aspect of application portability is that Big SQL 3.0 allows third party SQL application tools to use common client drivers like JDBC and ODBC when connecting to Big SQL 3.0 databases. The benefit is that existing queries can be run with little to zero modifications. Big SQL 3.0 tables can be accessed using the DB2 data server client as well as the Big SQL JDBC driver. This is the same client that is used to access DB2 LUW. This makes it possible to easily access Big SQL tables with many popular API's including PERL, Python and Ruby. The Big SQL node and database are cataloged using the same commands as for a DB2 LUW database. Big SQL 3.0 SQL functionality is consistent with the SQL language support in DB2 LUW.[10] In this illustration (Fig. 4), a query is run to access a table in Big SQL 3.0 on a Hadoop cluster from the DB2 client command window on Windows operating system.[19]

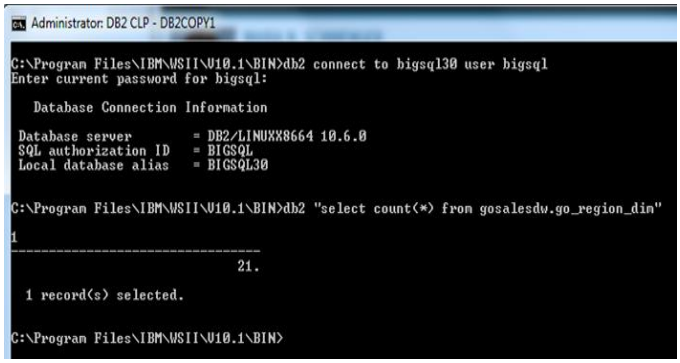

Fig. 4. Access a Big SQL 3.0 table from DB2 CLP on Windows OS

Big Insights also includes an IBM's Big R feature, which allows the R algorithms to execute across all nodes of the Hadoop cluster. R includes a JDBC package called RJDBC that can be installed from the R console from the free RStudio client shown in Figure 5a. This driver can utilize the DB2 driver to access data in Big SQL 3.0 tables. Once the data is returned to the R instance, you can create data frames to do further analysis and visualizations.

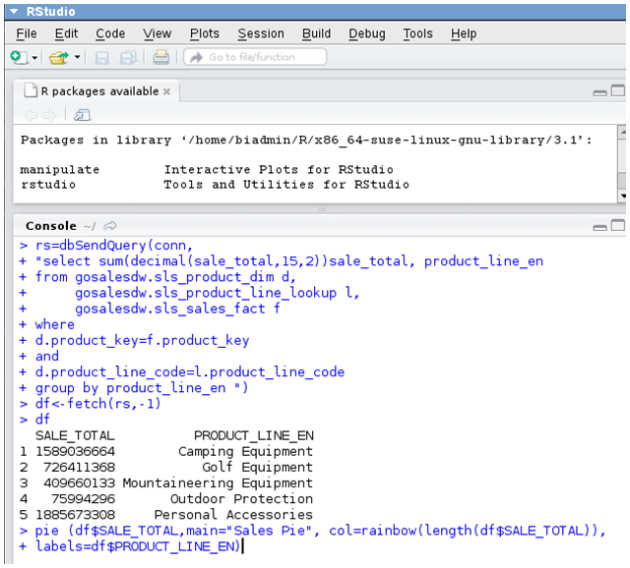

Fig. 5a. Big R Studio executing a query on Hadoop cluster

In Figure 5b below we can see the connection information, result set, and the graph generated by the RStudio client.

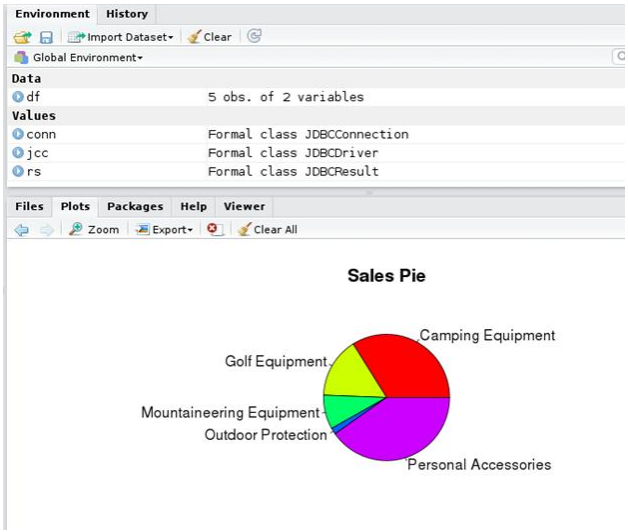

Fig. 5b. Big R Studio executing a graph on Hadoop cluster

The command line interface to Big SQL in the Hadoop cluster is called jsqsh and it is available to run from the $BIGINSIGHTS_HOME/jsqsh/bin directory. When you start jsqsh, you can configure connections using a step-by-step dialog. Once connected, you can enter SQL statements and SQL commands to SHOW TABLES and DESCRIBE connections. You retrieve data from system tables and views as shown below in Figure 6a.

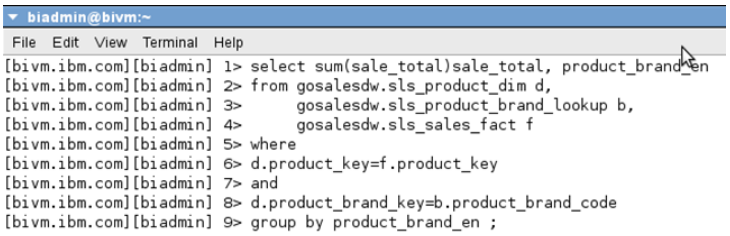

Fig. 6a. Running query in jsqsh

SALE_TOTAL	PRODUCT_BRAND_EN
184559636.65000	Alpha
142417087.53000	Antoni
57131718.12000	Blue Steel
19232849.47000	BugShield
289639036.22000	Canyon Mule
100566897.13000	Course Pro
80454510.84000	Edge
346978986.80000	Epoch
62036795.49000	EverGlow
355808650.26000	Extreme
90016502.11000	Firefly
82911001.67000	Glacier
167465751.05000	Granite
568712752.64000	Hailstorm
229215938.70000	Hibernator
130597495.53000	Husky
38252598.17000	Mountain Man
67164541.58000	Polar
302113144.55000	Relax
12429699.12000	Relief
61883515.16000	Seeker
528221728.02000	Star
26741754.61000	Sun
272835984.18000	TrailChef
323364742.75000	Trakker
146022450.50000	Xray

Fig. 6b. Query result in jsqsh

Being able to use common query tools (with visual environment), which most big data professionals are accustomed to when doing analytics, is another benefit expected from the SQL-on-Hadoop solutions. In Big Insights, from the Welcome tab of the web console (Quick Links pane), you can open a browser window, from which you can run queries against Big SQL tables. Query history for the session can be retrieved from the drop down menu above the SQL entry window. Results are shown below the SQL text. (Fig.7b)

Figures 7a and 7b present the query and the results set as displayed in the Web Browser interface in BigInsights - Big SQL 3.0 Web console.

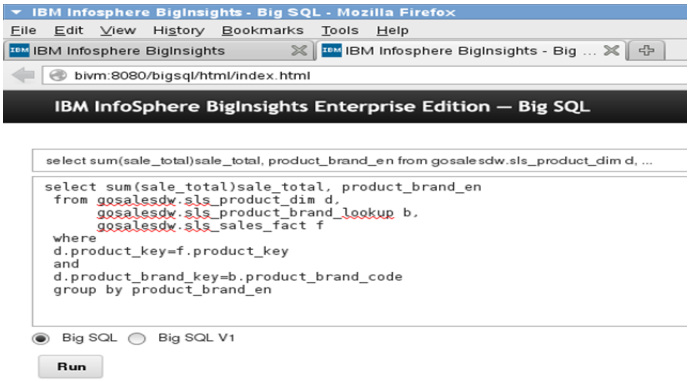

Fig. 7a. Web Browser Interface in Big Insights - Big SQL 3.0 query

Fig. 7b. Web Browser Interface in Big Insights - Big SQL 3.0 result set

D. Performance

One of Big SQL 3.0's major benefits is performance and performance monitoring. It facilitates monitoring of resources and software management.[12] The performance tooling is built from the ground up to surpass low latency and optimize query execution. Its operations run in memory, and its daemons live directly on the Hadoop cluster.[1] As a result, there is less overhead and reduced latency to retrieve data. There is no need to initiate MR jobs on the cluster where the data is located. Experimental works show that the SQL performance is significantly improved when the SQL engine is not dependent on MapReduce.[10]

Big SQL 3.0 query optimization has the following forms: query rewrites, query plan optimization, predicate pushdowns and statistics-driven optimization. Applying query rewrites is an ideal way to execute queries with minimal resources. Query plan optimization includes many sub query optimizations such as in joins. It is a sub query amongst data that is spread out throughout the Hadoop cluster. Shared aggregation and distinct elimination leverage key definitions in a consolidated rewrite. For example, Big SQL 3.0 distinct elimination does its query writes by eliminating the use of the DISTINCT operation if the key values are already unique.[1] Predicate pushdown processes data reads based on static variable definitions. Finally, statistics-driven optimization stores its column data in the query optimizer to determine an ideal query execution. These transformations are based upon years of IBM's relational database experience and greatly improve statistics and heuristics.

E. Enterprise Capabilities

The enterprise level capabilities of Big SQL 3.0 include functionalities like monitoring, management and security of the SQL engine running over the Hadoop system. Resource, event, memory and workload management enhance the Hadoop functionality for stored data in BigInsights.[9] The security capabilities include secure user authentication, authorization, auditing capabilities, and fine grained row/column access control.

User authentication is the verification of a person's identity through Kerberos or Lightweight Directory Access Protocol (LDAP). Big SQL 3.0 comes with a built-in module that verifies that access and determines if they have authorization to different data sets and its specific resources.[18] To validate such access, the audit facility creates access control lists, which limits unwanted behavior or data access. One of those lists is the row access control list that limits access to certain parts of the data at row or column level.

III. Evaluation of Big SQL 3.0

In attempt to evaluate the Big SQL 3.0 functionality, we examined multiple use cases and chose to measure the efficiency of the new functionality by creating a complex reporting structure over a Hadoop cluster. We selected two distinct use cases:

1) Migrating an existing report from a custom developed reporting system running against a relational database management product (Oracle® database);

2) Creating a new report directly on the Hadoop cluster

From a scheduling perspective, the project manager provided us with a 4 weeks' time range. She considered that as more than enough time to create a report. The major steps we followed in our evaluation are:

a) Design of the reports (Identifying the report content): This is usually the simplest step, which took approximately the same time when working against the relational database and against the Hadoop cluster. However, there is one important distinction: We did not evaluate the time required to identify the data set needed for the new report written directly against the Hadoop cluster without the usage of a visual environment, in this case IBM® Data Studio (Data Studio). Using a query tool like Data Studio to sample data (which was possible, of course, based on the integration capabilities provided by Big SQL 3.0) greatly simplified the process and shortened the time to design the new report. It allowed the analysts to "see" the data set they wanted to collect and model into the report. The same task performed without this visual environment would have been tedious to complete as it would have required developing a MapReduce job to sample the data. The estimated completion time for this situation was too high to test ranging from 70 to 80 hours. (Fig. 8a)

b) Creating and generating the reports: As expected, writing the queries that will return the data set for the reports was the most time consuming step. When designing the report, the analyst had to determine the proper syntax to extract the desired data set using Big SQL 3.0 and against the Hadoop cluster. In the case when an existing report was migrated to the Hadoop cluster, this step showed significant difference in the timing needed to complete the task. The time was significantly shorter when Big SQL 3.0 was used, generally limiting the effort of running the queries from the existing report directly against the data stored in the Hadoop cluster.[14] The only setup task was to connect their custom reporting management user interface to the Hadoop cluster via ODBC. The same task proved to be much more complicated when a new MapReduce job had to be created. In this case, the analysts were not able to complete the task by themselves and they had to call for development/Java support.

We encountered even greater complications when attempted to create a new report working directly against the Hadoop cluster. The analysts did not have the necessary skills to even initialize the development process and we had to wait for a Java developer to do the coding of the new report. The analyst had to spend additional time explaining to the Java developer the intent of the report, what the final result should be, some additional design considerations such as the report layout, title fields, summary groups, etc.

The opposite was the situation when the analyst attempted to create a new report utilizing the Big SQL 3.0 functionality.[13] The analyst approached the process in his usual manner working in a familiar reporting environment. The fact that the queries run against data stored directly on the Hadoop cluster was seamless to the analyst. The analyst was using the same SQL and statistical skills he used when creating report against the relational database management system.[12]

c) Testing the reports: Once the analyst finished implementing the report, he ran it on a sample data set to ensure correct reporting. Often the query may require changes if it was not properly completed by the first try and the design process may require re-iteration. We encountered greater simplicity when Big SQL 3.0 was in use for obvious reasons. When writing MapReduce jobs, a majority of the time was spent on making changes to the Java code and redeploying the application.

Of course, the time estimated to design and implement the two reports would vary based on the skill set and experience level of the report analyst and the Java programmer. Our observations were based on the existing skills set by the team that had estimated the medium to advance level skill set (between 4 and 8 years of professional experience).

Figure 8a presents the average time needed to deliver the reports we described above. It includes the time for direct exploration of the data located on the Hadoop cluster we did not include in the official estimations in the design phase. The numbers confirmed our expectations that the utilization of Big

SQL 3.0 significantly simplifies the amount of work required to migrate an existing report from running on a relational database management system to a Hadoop cluster.

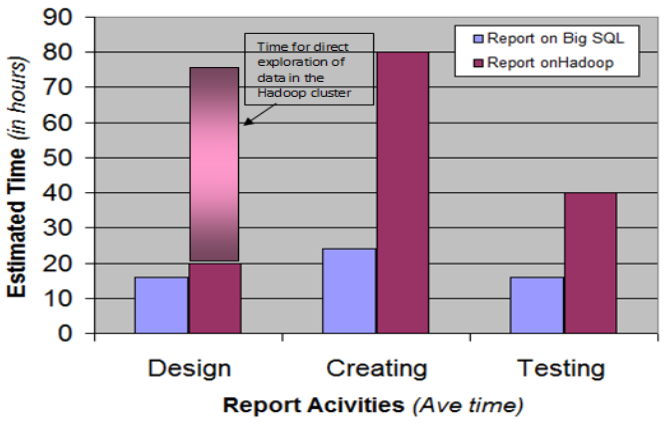

Fig. 8a. Total amount of time to complete the reports

We also evaluated the time spent for the delivery of each report. We recorded the time used in each step of the two use cases described above. A breakdown of the total time spent in each step expressed as a percentage is illustrated in Figure 8b. Using these numbers conservatively, we estimated the time savings by subtracting the time used for the creation of each report from the four-week period provided as a deadline by the business owner of the reports.

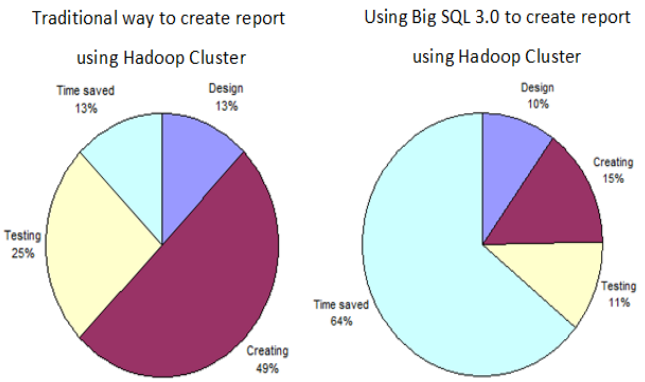

Fig. 8b. Breakdown of time spent in each step of the report creation

Figure 8b shows the significant time savings of 64% when the report was developed utilizing Big SQL 3.0. Another interesting comparison is the time required to create the report directly on the Hadoop cluster, which is almost 50% of the total effort. The same process takes an average of 15% when Big SQL 3.0 is utilized. Similar conclusion can be drawn about the testing efforts for each report.

IV. Conclusion

This paper provides an overview of Big SQL 3.0 functionality with concentration on the simple application integration that demonstrates how the data stored on HDFS can be easily accessed by many existing applications. By providing comprehensive SQL language support, Big SQL 3.0 allows applications developed for relational databases to be easily extended to leverage the Hadoop environment. The federation layer of Big SQL 3.0 also assists applications running on the BigInsights cluster to integrate queries through multiple data sources (structured, not structured, and semi-structured).

The short feasibility evaluation shows that the time required to create complex business intelligence reports is significantly reduced when using Big SQL 3.0 functionality instead of MapReduce programming. The benefit of Big SQL 3.0 is that it provides a robust SQL solution with very few limitations that improves user's productivity and experience by utilizing existing skills and applications.

As outlined above, the functionality of BigInsights – Big SQL 3.0 provides rich SQL support and enhanced processing of SQL queries over data stored in Hadoop. It supports a combination of features that guarantee enterprise level performance, security, management and monitoring.

Acknowledgment

The authors would like to acknowledge a number of colleagues and friends who helped them to build their knowledge and understanding of the Hadoop and Big Insights – Big SQL 3.0 technology and the software products described here.

Disclaimer

The views and opinions expressed in this article are solely of the authors: they are not necessarily those of IBM or any other company or institution mentioned in this paper. All examples and sample code are provided "AS IS", only for illustrative purposes. The authors grant you a nonexclusive copyright license to use all programming code examples from which you can generate similar function tailored to your own specific needs.

Trademarks

IBM, the IBM logo, ibm.com, and DB2 are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and

REFERENCES

[1] S. Gray, F. Ozcan and H. Pereyra. "SQL-on-Hadoop without compromise. How Big SQL 3.0 from IBM represents an important leap forward for speed, portability and robust functionality in SQL-on-Hadoop solutions." IBM Software Group. IBM.com. Retrieved from: http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=SA&subtype=WH&htmlfid=SWW14019USEN#loaded. pp. 2-19. April 2014.

[2] "IBM InfoSphere BigInsights Version 3.0. IBM® Big SQL. IBM Knowledge Center." IBM.com. Retrieved from: http://www-01.ibm.com/support/knowledgecenter/SSPT3X_3.0.0/com.ibm.swg.im.infosphere.biginsights.product.doc/doc/bi_sql_access.html. pp. 1-3. 2014

[3] .T. White. "Hadoop: The Definitive Guide 3rd Edition." Sebastopol, CA: O'Reilly Media Inc. pp. 10-14. 2012

[4] D. DeWitt, R. Nehme, and J. Gramling. "Split Query Processing in Polybase." In ACM SIGMOD, pages 1255–1266, 2013.

[5] "InfoSphere BigInsights: Bringing the power of Hadoop to the Enterprise." IBM.com: Retrieved from: www.ibm.com/software/data/infosphere/biginsights pp 1-3.

[6] "IBM Big Data and Information Management. Now known as IBM Watson Foundations." IBM.com: Retreived from: http://www-01.ibm.com/software/data/bigdata/ pp. 1-3.

[7] M. Stonebraker, D. Abadi and D. DeWitt. "MapReduce and parallel DBMSs: Friends or Foes?" CACM, 53(1):64–71, 2010.

[8] "IBM InfoSphere BigInsights Version 2.1. Big SQL Reference." IBM. Com Retreived from: http://www-01.ibm.com/support/knowledgecenter/api/content/SSPT3X_2.1.1/com.ibm.swg.im.infosphere.biginsights.bigsql.doc/doc/bsql_reference.html. pp 1.

[9] "Streaming SQL for Hadoop. Make the Elephant Fly. Real-time Data with SQLstream." SQL Stream. http://www.sqlstream.com/solutions/streaming-sql-for-hadoop/. pp 1-3.

[10] R. Chong, C. Liu. "DB2 Essentials: Understanding DB2 in a Big Data World (3rd Edition)". IBM. pp: 2014.

[11] R. Lee, T. Luo, and X. Zhang. "YSmart: Yet Another SQL-to-MapReduce Translator." In ICDCS, pages 25–36, 2011.

[12] W. Hu, N. Kaabouch. "Big Data Management, Technologies, and Applications" IGI Global. pp: 1-509. 2014

[13] "Yahoo! Launches World's Largest Hadoop Production Application,". Yahoo Inc. Retrieved from: *http://developer.yahoo.net/blogs/hadoop/2008/02/yahoo-worlds-largest-production-hadoop.html* pp. 1-3. 2014.

[14] A. Floratou, U. Minhas and F. Ozcan. "SQL-on-Hadoop: Full Circle Back to Shared-Nothing Database Architectures." Retreived from from: http://pages.cs.wisc.edu/~floratou/SQLOnHadoop.pdf pp. 1-12.

[15] C. Prokopp. "The Free Hive Book". Semantikoz.com. Retreived from: http://www.semantikoz.com/blog/the-free-apache-hive-book/ Chapter 7. May 2013.

[16] G. Allouche. "Hadoop Happenings: Back to the Basics and SQL-on-Hadoop Frenzy." Qubole. Retreived from: http://www.qubole.com/hadoop-back-to-basics/ pp. 1-3. July 2014.

[17] M. Rathbone. "8 SQL-on-Hadoop frameworks worth checking out". Retreived from: http://blog.matthewrathbone.com/2014/06/08/sql-engines-for-hadoop.html pp. 1-10. June 2014.

[18] C. Saracco, U. Jain. "What's the big deal about Big SQL?" IBM Developers Works. Retreived from: http://www.ibm.com/developerworks/library/bd-bigsql/ pp. 1-11. June 2013.

[19] P. Zikopoulous, S. Lightstone, M. Huras. "DB2 10.5 with BLU Acceleration: New Dynamic In-Memory Analytics for the Era of Big Data" McGraw-Hill Osborne Media 1st Edition. Pp. 35-78.