AVOID DATA CONVERSION IN HADOOP USING CUSTOM SERDES

Shreyas Kothavade, Rohan Oswal, Rohit Khirid, Anurag Mane
Department of Computer Engineering,
VIIT Pune, Pune University, Pune City, Maharashtra, India

ABSTRACT
The recent advancements in technology and the wide spreading internet connectivity have resulted in generation of large amount of data. In fact, according to the survey about 2.5 quintillion bytes of data is created each day. Various operations need to be performed on this data to extract useful information from it. Performing these operations manually is practically impossible and will require a huge amount of manpower. Many techniques have been used to reduce the processing time required to perform these operations. In this paper we analyse one of the present strategies that is storing the data of various data types such as CSV, EBCDIC, Oracle Dumps, Positional, Delimited etc. by converting it in ASCII format and retrieving the data by again converting the ASCII data into its original format. We propose a new strategy based on the concept of SerDe (Serializer Deserializer) which helps the information storage and retrieval of Big Data in minimum amount of time using Hadoop platform. With the help of this concept, we aim at reducing the time consumption required to extract the information.

KEYWORDS: *SerDe, HDFS, EBCDIC, CSV, ETL, Big Data, Hadoop, Serializer and Deserializer.*

I. INTRODUCTION

Big data is a term that describes any voluminous amount of structured, semi-structured and unstructured data that can be mined to get useful information. [2] Big data is a leading trend in today's industry. It is the data which is beyond storing capacity and processing power of the systems we use traditionally. Special systems are needed to mine process this voluminous amount of data and mine useful information from it. This data can be used to extract important information using predictive analysis. Examples: data from Social Networking sites, Airports, Defense data etc.
Namely there are 4 V's in Big Data: Volume, Velocity, Variety, and Veracity. [2]
Volume: Scale of Data.
Velocity: Analysis of streaming data.
Variety: Different forms of Data.
Veracity: Uncertainty of Data.
Big Data can be processed with the help of Hadoop.
Hadoop:
Hadoop is an open source framework in which processing of huge data sets is done in distributed computing environment. [4] Hadoop is an Apache open source framework written in Java. The processing is done across clusters of computers with the help of simple programming models. It is designed in a view to scale up from single machines to thousands of machines, offering local computation and storage. Hadoop is open source software. [4] It is a framework. Hadoop has massive amount of storage. Hadoop processes voluminous amounts of data using multiple low-cost computers for fast results. Hadoop is Fault tolerant, low cost as it is open source and scalable.
Hadoop has two components:

HDFS (Hadoop Distributed File System).
MapReduce
HDFS:
The Hadoop Distributed File System is based on the Google FileSystem. It provides a distributed file system that is designed to run on large clusters of small computer machines in a reliable, fault-tolerant manner. [5] HDFS consist of master-slave architecture. Master consists of single name node that manages the file system metadata and Data nodes that store the actual data.
MapReduce:
Hadoop MapReduce is used to process large amount of data stored in the HDFS.A MapReduce job splits the input data-set into independent chunks and are processed by map tasks in a completely parallel manner. [5] The framework sorts the outputs of the maps, which are then input to the reduce tasks.

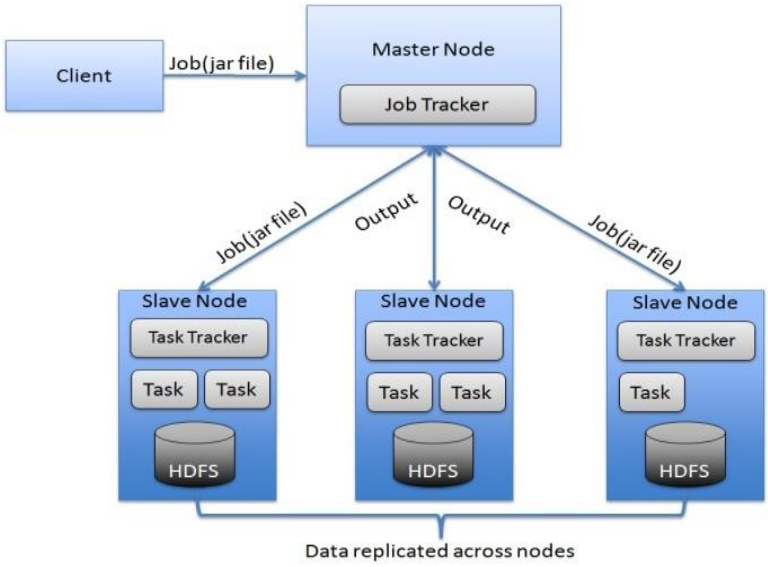

Fig 1: Architecture of Hadoope:

Apache Hive is a data warehouse infrastructure built on top of Hadoop for providing data summarization, query, and analysis. Hive allows SQL developers to write Hive Query Language (HQL) statements that are similar to Standard SQL statements. HQL statements are broken down by the Hive service into Map Reduce jobs and executed across a Hadoop cluster.
Benefits of Hadoop:
One of the top reasons that organizations turn to Hadoop is its ability to store and process huge amounts of data – any kind of data – quickly. With data volumes and varieties constantly increasing, especially from social media and the Internet of Things, that's a key consideration.
Also other benefits include:
1. Computing power
2. Flexibility
3. Fault Tolerance
4. Low Cost
5. Scalability

II. PRELIMINARIES

2.1 CSV

A comma-separated values(CSV) file stores tabular data in plain text. Each line of the file is a basically data record. Each record has one or more fields which is separated by commas. The use of the comma as a field separator is the source for the name of this file format.[13]

There is no official standard for the CSV file format. CSV format may denote some closely related delimiter-separated formats, which uses different field delimiters. These do include tab-separated values and space-separated values and such files are often even given a .csv extension, though a different field separator than the comma is used. This format are best used to represent sets or sequences of records which have each record as an identical list of fields.CSV can be considered a common data exchange format that is widely supported by consumer, business, and scientific applications. Its major applications include moving tabular data between programs that operate on incompatible formats.[13]

2.2 EBCDIC

EBCDIC (Extended Binary Coded Decimal Interchange Code) is used by IBM in their mainframes and is a character encoding set.[12] It is an eight-bit character encoding used mainly on IBM mainframe and IBM midrange computer operating systems. EBCDIC has descended from the code used with punched cards with the corresponding six bit binary-coded decimal codes. EBCDIC makes use of the full 8 bits available to it, so it cannot make use of parity on an 8 bit system.[12] EBCDIC provides a wider range of control characters than ASCII. The encoding scheme is based on Binary Coded Decimal (BCD) being an extension of BCD .It forms contiguous characters in the alphanumeric range in blocks of up to 10 from 0000 binary to 1001 binary. EBCDIC contains four main blocks in its code page: 0000 0000 to 0011 1111(00-3F) is reserved for control characters; 0100 0000 to 0111 1111(40-7F) are for punctuation; 1000 0000 to 1011 1111(80-B7) for lowercase characters and 1100 0000 to 1111 1111(C0-FF) for uppercase characters and numbers. EBCDIC puts lowercase letters before uppercase letters and letters before numbers, exactly the opposite of that in ASCII. Due to EBCDIC's lack of codes for several symbols (such as the brace characters) which are commonly used in programming and in network communications software portability and data exchange are hindered. The gaps present between some letters made simple constructions that worked in ASCII fail on EBCDIC systems. For example, ‘Z' minus 'A' was 40, not 25. This would usually cause problems when porting software from ASCII systems.[12]

2.3 ASCII

ASCII stands for American Standard Code for Information Interchange is the most commonly used character-encoding scheme. ASCII codes represent text in computers, gadgets, communications equipment, and other devices that use text.[14] Most of the modern character-encoding schemes are based on ASCII, though they support many additional characters. ASCII is of two types ASCII 7 and ASCII 8. Originally based on the English alphabet, ASCII 7 encodes 128 specified characters into seven-bit integers. The different characters which are encoded are numbers 0 to 9, lowercase letters a to z, uppercase letters A to Z, basic symbols, control codes and a space.[14] For example, lowercase k would become binary 1101011 and decimal 107.ASCII 8 uses 8 bits to represent a character and can represent 256 different characters. ASCII like other encoding schemes specifies a correspondence between digital bit patterns and character symbols i.e. graphemes and control characters which allows the digital devices to communicate with each other and to process, store, and communicate character-oriented information such as written language.[14]

2.4 Delimited Files

Files in which delimiter-separated values are used to store two-dimensional arrays of data by separating the values in each row with specific delimiter characters are called Delimited Files. Most of the database or spreadsheet programs are able to read or save data in a delimited format. In a delimited text file is a text file which used to store data, in which each a line represents single object, company, or some other thing, and each line has fields separated by the delimiter. The delimited file has the advantage of allowing field values of any length compared to that of a flat file that uses spaces to make every field to same width. Here any character may be used to separate the values, but the mostly common delimiters are comma, tab, and colon. The vertical bar also called as pipe and space are also sometimes used. For example in a comma-separated values (CSV) file the data items are separated using commas as the delimiter, while in a tab-separated values (TSV) file, the data items are separated using tabs as the delimiter. Column headers are usually included as the first line, and are followed by rows of data. Each of these lines are separated by newlines. There are two

types of Delimiters : Field and record delimiters(in CSV file a comma as the delimiter between fields, and an end-of-line indicator as the delimiter between records) and Bracket delimiters(mark start and end of a region of text , example '( )','{ }','[ ]','<>' etc.

III. CURRENT TECHNOLOGY

Some of the technologies that are currently used for data storage and retrieval are mentioned below.

3.1 ETL Tool

Currently, industries deal with a variety of data such as EBCDIC, CSV, Oracle Dumps, Positional, Delimited etc. They use an ETL (Extract Transform Load) tool. Extract, Transform and Load (ETL) refers to a process in database usage and especially in data warehousing that:
a. Extracts data from homogeneous or heterogeneous data sources.
b. Transforms the data for storing it in the proper format or structure for the purposes of querying and analysis.
c. Loads it into the final target (database, more specifically, operational data store, data mart, or data warehouse)
Using this tool, this data is loaded into the HDFS by first converting it into ASCII. While retrieving the data from HDFS, this ASCII data is again converted into the original data and is then displayed to the user.
The drawback of this approach is that the process consumes a lot of time as conversion of data from one format to the other takes a huge amount of time.
Working of Existing system is as follows :

Fig 2: Process of Serialization and Deserialization

3.2 Talend Tool

Talend is a software vendor specializing in Big Data Integration. The company provides big data, Cloud, data integration, data management, Master Data Management, data quality and enterprise application integration software and services.
Though, this technique is faster as compared to the ETL process, the time required for processing large files is much more.

IV. PROPOSED SYSTEM

We propose a concept called as SerDe which will help to retrieve data in fastest possible way. SerDe, as you can see is a combination of two words Serializer and Deserializer.

4.1 Serialization

Serialization is the process of converting an object into a stream of bits. [11] This process is mainly used for two purposes. Firstly, for transmission of data over a network. This transmission takes place with the help of inter process communication. As computers are internally represented with the help of a binary format, it is not possible to transmit data as it is over a network. The data is first serialized and is obtained in a binary format. This data is then transmitted. Secondly, Serialization is also used to load data to a persistent storage. This process is also called as input processing and it is performed during data loading.

4.2 Deserialization

Deserialization is the exact opposite process of serialization. In deserialization, we convert the stream of bits back to its original format. This process is also called as output processing and is performed during data fetching. The primary function of Deserializer is to take data from HDFS and convert it into an object that can be manipulated by hive.
The process of Serialization and Deserialization is as shown below.

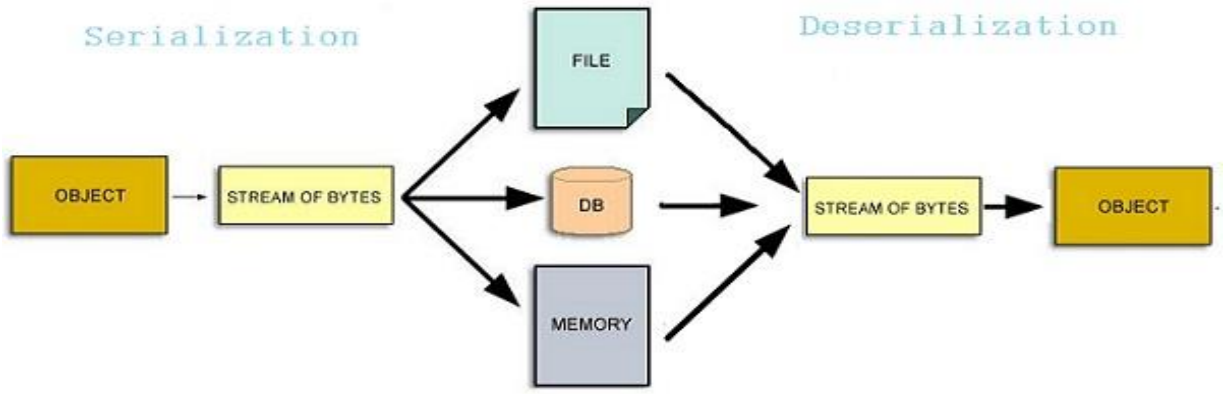

Fig 3: Process of Serialization and Deserialization

4.3 Working of Proposed System

The existing system needs the data to be converted twice i.e before the loading operation and after the fetching operation. This consumes a lot of time which can be avoided with the use of the proposed system. In this system, while storing the data, the data is serialized without converting it to ASCII and then it is stored in original format. During fetching operation, the data is deserialized. The deserialized data is obtained in the original format. Proposed system will eliminate the process of conversion of data format from one format to another data format. Instead it will store the data in its original format and while fetching it will create an abstract view of data obtained using Hive. The user of the system can then obtain the required data by executing queries using Hive.
The steps followed are:

Fig4: Working of proposed system

V. EXPECTED OUTCOMES

After the completion of the project, we expect the data to be dumped in the HDFS in the format it has been received. This will avoid the time required for conversion of files into ASCII. When the user enters a query using HIVE query editor, the data in the HDFS will be processed using MapReduce on different nodes and only the required data will be shown to the user in ASCII format. Here, the time required for conversion of complete data is avoided. We create an abstract view of the data on HIVE editor in which the required data is temporarily converted into ASCII. Thus, we aim at eliminating 3 traditional steps involved in processing the data:
1. Manual pre-processing of data.

2. Conversion of data while loading to HDFS.
3. Conversion of complete data while fetching from HDFS.

VI. FUTURE WORK

After the implementation of the proposed project, the companies generating data by IBM mainframes can view the EBCDIC, Delimited, CSV and ASCII data and can process it in minimum amount of time. This will help the organizations to employ less manpower to convert the data from various data formats to ASCII format.
Our future work includes researching on various traditional data formats and writing of SerDes for these data formats which consume a lot of time converting into ASCII. This will help more and more organizations to come forward and use the product.

VII. CONCLUSIONS

Thus, by implementing the above method of SerDe we aim at reducing significant amount of time required for processing the data. We avoid conversion of various data formats to ASCII during storage as well as during retrieval of data. We create an abstract view of the data to be displayed so as to reduce the time required for conversion.

ACKNOWLEDGEMENTS

We hereby thank Prof. S.B. Tatale (Department of Computer Engineering, VIIT,Pune) for his valuable inputs on Big Data and Hadoop. We also thank Mr. Vinayak Shinde (Director PRGX,India) and Mr. Ashay Dhavale (PRGX, India)for their inputs on concepts of Serialization and EBCDIC files.

REFERENCES

[1] Olston, G. Chiou, L. Chitnis, F. Liu, Y. Han, M. Larsson, A. Neumann, V. B. N. Rao, V. Sankarasubramanian, S. Seth, C. Tian, T. ZiCornell, and X. Wang, Nova: Continuous pig/hadoop workflows, in Proc. of SIGMOD'2011, New York, NY, USA, 2011
[2] Big Data, http://www.ibmbigdatahub.com/infographic/four-vs-big-data
[3] Big Data, http://www.dummies.com/how-to/content/the-4-vs-of-big-data.html
[4] Hadoop, http://www.tutorialspoint.com/hadoop/hadoop_introduction.html
[5] HDFS, http://hortonworks.com/hadoop/hdfs
[6] J. Gantz and D. Reinsel, ''The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east,'' in Proc. IDC iView, IDC Anal. Future, 2012.
[7] J. Manyika et al., Big data: The Next Frontier for Innovation, Competition, and Productivity. San Francisco, CA, USA: McKinsey Global Institute, 2011, pp. 1–137.
[8] K. Cukier, ''Data, data everywhere,'' Economist, vol. 394, no. 8671, pp. 3–16, 2010.
[9] T. economist. (2011, Nov.) Drowning in Numbers—Digital Data Will Flood the Planet- and Help us Understand it Better [Online]. Available: http://www.economist.com/blogs/dailychart/2011/11/bigdata-0
[10] S. Lohr. (2012). The age of big data. New York Times [Online].11.Available: http://www.nytimes.com/2012/02/12/sunday-review/big-datasimpact-in-the-world.html?pagewanted=all&r=0
[11] Serialization, https://en.wikipedia.org/wiki/Serialization
[12] EBCDIC, http://search400.techtarget.com/definition/EBCDIC
[13] CSV, https://en.wikipedia.org/wiki/Comma-separated_values
[14] ASCII, https://en.wikipedia.org/wiki/ASCII

AUTHORS

Shreyas Kothavade Born: Thane, Maharashtra, India Currently pursuing Final Year of Bachelors of Computer Engineering From VIIT, Pune Research Interest: Hadoop, BigData, DBMS.

Rohan Oswal Born: Pune, Maharashtra, India Currently pursuing Final Year of Bachelors of Computer Engineering From VIIT, Pune Research Interest: Hadoop, BigData, Compilers.

Rohit Khirid Born: Pune, Maharashtra, India Currently pursuing Final Year of Bachelors of Computer Engineering From VIIT, Pune Research Interest: Hadoop, BigData, Android Application Development.

Anurag Mane Born: Thane, Maharashtra, India Currently pursuing Final Year of Bachelors of Computer Engineering From VIIT, Pune Research Interest: Hadoop, BigData.