Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2020, 7(1):41-45

Research Article ISSN: 2394 - 658X

Streamlined Data Queries: Integrating Fixed-Width and CSV Data into Apache Hive

Pankaj Dureja

Pankaj.Dureja@gmail.com

ABSTRACT

This paper discusses loading Fixed-Width and CSV Data along with External Tables Implementation to Efficiently Query It. While comparatively simple and ubiquitous, fixed-width as well as CSV formats are often challenging processing targets - a difficulty only compounded within large-scale data ecosystems. With Apache Hive, which is built on top of the Hadoop ecosystem, handling structural data through a SQL-like interface becomes possible. Hive external tables provide a way to have fixed-width or CSV data, ingested and saved in an organized fashion with an ability to query that from anywhere in the cluster. The benefits of using external tables to manage these data formats are pointed out and specifics about how to configure Hive do such for fixed-width file and CSV files.

Keywords: Apache Hive, Data Ingestion, Fixed-Width, CSV, Data Management, Big Data Analytics, SerDe, LazySimpleSerDe, OpenCSVSerDe

INTRODUCTION

The explosion of data in a multitude of forms - structured, semi-structured or unstructured. Organization requires to process and analyze many datasets cost effectively across the board. Although simple to use, straightforward integration in many applications, and wildly popular formats such as fixed-width or CSV are still some of the least-effort choices in terms of huge amounts processing. And dealing with these formats efficiently in distributed environment becomes critical to unlocking big data technologies.

Apache Hive, a data warehousing solution built atop the Hadoop ecosystem, offers a SQL-like interface that facilitates the querying and analysis of structured data housed in the Hadoop Distributed File System (HDFS) or other compatible storage systems. Initially tailored for structured data types like CSV, Hive has expanded to accommodate a wide range of data formats, enhancing its versatility in big data scenarios.

By contrast, the working of fixed-width and CSV files) in Hive is complicated due to many problems such as aligning data and parsing it without any predefined delimiters minimizing large task with huge dataset. These challenges require advanced tools and methodologies in order to avoid performance bottlenecks and optimize the use of computing resources.

Leveraging Apache Hive for processing fixed width and CSV data offers several benefits. It integrates seamlessly with existing Hadoop infrastructure, supports distributed computing environments, and provides familiar SQL-based querying capabilities. By utilizing Hive's external tables feature, fixed-width and CSV data can be efficiently ingested, stored, and queried, enabling organizations to process and analyze their data effectively alongside other structured data sources. This integration allows enterprises to extract valuable insights from their datasets while maximizing their existing infrastructure investments.

PROBLEM STATEMENT

In my role at an oil and gas company, I am tasked with managing the data loading process for various datasets, including two that are in fixed-width format and one in CSV format, the latter being the most commonly used. To handle these datasets efficiently, we utilize Apache Hive to facilitate the loading and processing.

Sample of fixed Width file

First Column Country is from 1 to 16 Characters

Second Column Product type is from 17 to 32 Characters

Third Column Date is from 33 to 62 Characters
Last Column Volume is from 63 to 69 Characters

```
AUSTRALI        LPGETHANE       JAN1984                         19.488
AUSTRALI        LPGETHANE       FEB1984                         20.432
AUSTRALI        LPGETHANE       MAR1984                         22.856
AUSTRALI        LPGETHANE       APR1984                         25.551
AUSTRALI        LPGETHANE       MAY1984                         25.101
AUSTRALI        LPGETHANE       JUN1984                         28.644
AUSTRALI        LPGETHANE       JUL1984                         36.701
AUSTRALI        LPGETHANE       AUG1984                         34.830
AUSTRALI        LPGETHANE       SEP1984                         28.644
AUSTRALI        LPGETHANE       OCT1984                         33.333
```

Sample of CSV file

The columns are separated by Comma

```
Country,County,Basin,DrillFor,Location,Trajectory,WellDepth,Year,Month,Week,RigCount,State/Province,PublishDate
UNITED STATES,SABINE,Haynesville,Gas,Land,Horizontal,10k-15k,2011,4,13,1,LOUISIANA,4/1/2011 0:00
UNITED STATES,TERREBONNE,Other,Oil,Inland Waters,Directional,10k-15k,2011,4,13,1,LOUISIANA,4/1/2011 0:00
UNITED STATES,VERMILION,Other,Gas,Land,Directional,5k-10k,2011,4,13,1,LOUISIANA,4/1/2011 0:00
UNITED STATES,VERMILION,Other,Gas,Land,Vertical,>15k,2011,4,13,1,LOUISIANA,4/1/2011 0:00
UNITED STATES,EDDY,Permian,Oil,Land,Horizontal,5k-10k,2011,4,13,1,NEW MEXICO,4/1/2011 0:00
UNITED STATES,EDDY,Permian,Oil,Land,Horizontal,10k-15k,2011,4,13,1,NEW MEXICO,4/1/2011 0:00
UNITED STATES,LEA,Permian,Oil,Land,Horizontal,10k-15k,2011,4,13,1,NEW MEXICO,4/1/2011 0:00
UNITED STATES,DIVIDE,Williston,Oil,Land,Horizontal,>15k,2011,4,13,1,NORTH DAKOTA,4/1/2011 0:00
UNITED STATES,DIVIDE,Williston,Oil,Land,Horizontal,>15k,2011,4,13,1,NORTH DAKOTA,4/1/2011 0:00
```

SOLUTION IMPLEMENTED

In a distributed environment it's really helpful for Apache hive to handle external tables which have fixed width and CSV data with open or collector delimiters. To accomplish this, several components need to be configured: Stand up Hive and configure it for these types; define fixed-widths or CSV table schema as appropriate; query the data using similar SQL-like syntax in your Hive. This section provides a detailed walkthrough to optimize external tables for fixed-width and CSV data processing in Apache Hive.

Setup and Configuration

Before processing fixed-width or CSV data in Apache Hive, ensure that all necessary components are properly installed and configured. This typically involves setting up a Hadoop cluster with Hive installed and configuring external storage systems as required (e.g., HDFS, cloud storage). Additionally, ensure that any necessary libraries and dependencies for handling fixed-width and CSV formats in Hive are available.

Once the environment is ready, define an external table in Apache Hive to reference the fixed-width or CSV data. Specify the table schema to match the structure of the data files, including column names and data types. When defining the table, specify the location of the data files, ensuring that they are accessible to the Hive cluster.

For fixed width files, each row can be treated as record, which can be broken down by SQL query.

In this below example, the `LazySimpleSerDe` SerDe (Serializer/Deserializer) is used to parse each row into a record.

```
hive> show create table iea_demand_hoed_fixed_data;
OK
CREATE EXTERNAL TABLE `iea_demand_hoecd_fixed_data`(
  `record` string COMMENT 'from deserializer')
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  'maprfs:/mapr/user/data_load/data/iea/latest/HOECDDE'
TBLPROPERTIES (
  'numFiles'='1',
  'totalSize'='8068511')
Time taken: 0.047 seconds, Fetched: 14 row(s)
```

For CSV files, there 2 SerDe's which can be used as follows:

1. LazySimpleSerDe

```
ROW FORMAT SERDE
 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
WITH SERDEPROPERTIES (
 'field.delim'=',',
 'serialization.format'=',')
```

2. OpenCSVSerDe

```
ROW FORMAT SERDE
 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
 'field.delim'=',',
 'quotechar'='\"', =',')
 'serialization.format'=',')
```

The choice between `LazySimpleSerDe` and `OpenCSVSerDe` depends largely on the specific requirements of the data format and the complexities involved in data handling. `LazySimpleSerDe` offers a generic, flexible solution for simple delimited data, whereas `OpenCSVSerDe` provides specialized handling for CSV files, respecting format nuances such as quotes and escapes.

```
hive> CREATE EXTERNAL TABLE `bhi_rigs_master_raw_data`(
  `country` string,
  `county` string,
  `basin` string,
  `drill_for` string,
  `location` string,
  `trajectory` string,
  `well_depth` string,
  `year` string,
  `month` string,
  `week` string,
  `rig_count` decimal(38,10),
  `state_province` string,
  `publish_date` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
WITH SERDEPROPERTIES (
  'field.delim'=',',
  'serialization.format'=',')
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  'maprfs:/mapr/user/data_load/data/bhi/latest/master_data'
TBLPROPERTIES (
  'numFiles'='1',
  'skip.header.line.count'='1',
  'totalSize'='75238644')
Time taken: 0.053 seconds, Fetched: 30 row(s)
```

Querying Fixed Width Data

Once the external table is set up, querying fixed-width data in Apache Hive can be performed using standard SQL-like syntax. For instance, you can employ SELECT statements to extract specific columns or implement substring conditions on the data. The first query type retrieves the entire record, whereas the second type of query allows for column creation by breaking down the record. This is done using the substring function, which specifies a starting index and the number of characters. The output from the second type of query can be used to create another table, effectively transforming the data from a single record format into a more structured row and column format.

```
hive> select * from iea_demand_hoecd_fixed_data limit 2;
-chgrp: 'domain users' does not match expected pattern for group
Usage: hadoop fs [generic options] -chgrp [-R] GROUP PATH...
OK
iea_demand_hoecd_fixed_data.record
AUSTRALI        LPGETHANE       JAN1984                         19.488
AUSTRALI        LPGETHANE       FEB1984                         20.432
Time taken: 0.068 seconds, Fetched: 2 row(s)
```

```
hive> select substr(record,1,16) country
    > , substr(record,17,16) producttype
    > , substr(record, 33,8) datetime
    > , substr(record,63,6) volume
    > from iea_demand_hoecd_fixed_data limit 2;
-chgrp: 'domain users' does not match expected pattern for group
Usage: hadoop fs [generic options] -chgrp [-R] GROUP PATH...
OK
country producttype     datetime        volume
AUSTRALI                LPGETHANE               JAN1984         19.488
AUSTRALI                LPGETHANE               FEB1984         20.432
Time taken: 0.105 seconds, Fetched: 2 row(s)
```

Querying CSV Data

When querying CSV data using a Serializer/Deserializer like `LazySimpleSerDe` or `OpenCSVSerDe`, the SerDe parses the CSV files to interpret and extract fields based on commas and quotation marks, adhering to CSV standards. This enables standard SQL-like queries to directly interact with the data, facilitating seamless data retrieval and manipulation.

```
hive> SELECT tbl.vessel
    > , tbl.vessel_flag
    > , tbl.vessel_class
    > , tbl.imo
    > , tbl.grade
    > FROM clipper_global_crude_data_csv tbl limit 10;
-chgrp: 'domain users' does not match expected pattern for group
Usage: hadoop fs [generic options] -chgrp [-R] GROUP PATH...
OK
tbl.vessel      tbl.vessel_flag tbl.vessel_class        tbl.imo tbl.grade
Zoya 1  PANAMA  VLCC    9081174 CRUDE
Super Sapphire  LIBERIA Aframax 9422988 SAUDI CRUDE
Ruby    LIBERIA Suezmax 9050278 MURBAN
Pirai   BRAZIL  LR1     8617081 BRAZILIAN CRUDE
Genessa INDIA   LR1     9183647 SOUTH PARS CONDENSATE
Pirai   BRAZIL  LR1     8617081 BRAZILIAN CRUDE
Navarz  IRAN    VLCC    9079078 SOROOSH (CYRUS)
Diamond II      IRAN    VLCC    9218478 SOUTH PARS CONDENSATE
Delta Kanaris   GREECE  Suezmax 9429015 HAMACA
Delta Harmony   GREECE  Suezmax 9408463 HAMACA
Time taken: 0.084 seconds, Fetched: 10 row(s)
```

POTENTIAL EXTENDED USE CASES

1. Data Integration: Utilizing Apache Hive external tables allows for the seamless combination of fixed-width and CSV data with other structured datasets. This integration facilitates a holistic analysis, providing richer insights and enabling a more comprehensive approach to data-driven decision-making.
2. Big Data Analytics: Leveraging the SQL-like interface of Hive, organizations can delve into detailed analytics with fixed-width and CSV datasets. This capacity helps unearth critical insights that are instrumental in shaping strategic decisions and optimizing business processes.
3. Schema Flexibility: External tables in Hive are equipped to handle schema modifications, which is crucial for organizations needing to adapt their data structures to evolving business needs. This flexibility ensures that new and varied fixed-width and CSV formats can be integrated without disrupting existing data systems.
4. Scalable Query Performance: With its robust distributed computing framework, Hive enables efficient and scalable querying of fixed-width and CSV data stored in external tables. This capability is essential for maintaining fast access to data insights, crucial for operational efficiency and timely decision-making in dynamic environments.

IMPACT

The fast processing of fixed-width files and CSV with Apache Hive ensures that organizations can make informed decisions because these insights have strengthened by accessing faster, more accurate data. This makes Hadoop-based implementations significantly less expensive than proprietary data warehousing solutions leveraging existing infrastructure and open-source technology like Apache Hive. Apache Hive is also highly scalable - this allows for the processing of large amounts of fixed-width and CSV data with ease so that you can continue to grow your dataset without having to worry about slow performance. In addition, the fact that external tables can be used with fixed-width and CSV data in conjunction to other data sources also has potential for better use of information leading corporations uncovering their profound insights holistically from multiple types of datasets.

SCOPE

From finance to healthcare, from retail to telecommunications Apache Hive external tables for fixed width and CSV processing are being used across industries highlighting wide use cases. As a result, Apache Hive and its external tables are evolving as well to be more capable of solving complex data problems since ongoing research is continually improving them. External tables can also be fed into new technology workloads, like machine learning and AI for more advanced analytics/predictive modeling against fixed-width or CSV data sets. Data professionals receive comprehensive training and education via the following areas: in using Apache Hive for these data formats fostering an environment of innovative skill development in order to enhance skills applicable within Data Analytics.

CONCLUSION

The Apache Hive external table for fixed-width and CSV data is highly advantageous in distributed computing worlds. Integrating these data formats into the Hive ecosystem provides organizations with a unified interface for querying, analyzing and joining against other structured data sources. With external table, schema evolution is easy and does not require us to modify the underlying data files improving flexibility towards changing data requirements. In this paper, we described the external table configuration for fixed width and CSV data in Apache Hive with demonstration of how well you can query it Using SQL syntax provided by hive like querying format and used some examples to demonstrate power flexibility & scalability built around hive. Furthermore, performance benchmarks and real-world examples demonstrate the efficiency and usefulness of this technique in many use cases. As Apache Hive and associated tooling advance down the path, we expect more ways to process both fixed-width and CSV data with increased capabilities which continues enhancing Hive as a strong platform for providing insights out of your data in order to drive future course-making decisions.

REFERENCES

[1]. Capriolo, E., Wampler, D., & Rutherglen, J. (2012). Programming Hive, O'Reilly Media. Chapter [4]: [HiveQL: Data Definition - External Tables], pp. 56.
[2]. Capriolo, E., Wampler, D., & Rutherglen, J. (2012). Programming Hive, O'Reilly Media. Chapter [5]: [HiveQL: Data Manipulation], pp. 71-76.
[3]. Capriolo, E., Wampler, D., & Rutherglen, J. (2012). Programming Hive, O'Reilly Media. Chapter [6]: [HiveQL: Queries], pp. 79-91.
[4]. Capriolo, E., Wampler, D., & Rutherglen, J. (2012). Programming Hive, O'Reilly Media. Chapter [15]: [Customizing Hive File and Record Formats], pp. 206.
[5]. Hive Language Manual. Available at http://wiki.apache.org/hadoop/Hive/LanguageManual.
[6]. Hive Wiki. Available at https://cwiki.apache.org/confluence/display/Hive/SerDe