Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2018, 5(12):1132-1136

Research Article ISSN: 2394-658X

Streamlining Oracle to Hive Data Transfer with Apache Sqoop & Spark: Addressing Empty Directories, Sticky Bit Issue and De-duplication

Pankaj Dureja

Email id – Pankaj.Dureja@gmail.com

ABSTRACT

This paper explores the optimization of data transfer from Oracle to Hive using Apache Sqoop and Spark, focusing on common challenges such as empty directories, sticky bit issues, and data deduplication. Through detailed analysis and practical solutions, we aim to streamline the data migration process, enhance data integrity, and improve overall efficiency. The findings are supported by case studies and practical implementations, showcasing the potential for extended applications in big data environments.

Key words: Oracle, Hive, Apache Sqoop, Apache Spark, Data Transfer, Empty Directories, Sticky Bit, Data Deduplication, Big Data, ETL

INTRODUCTION

In the realm of big data, Apache Sqoop, Spark, and Hive are pivotal tools that facilitate efficient data transfer and processing.

A. What is Apache Sqoop:

Sqoop is a framework for transferring data from/to hadoop (HDFS, HBase and Hive) to/from relational database management systems. It is a data import and export tool based on command-line interface. Sqoop offers parallelized data transfer for moving the data from source to destination as quickly as possible. Sqoop is used for importing data from relational databases such as MySQL, Oracle, PostgreSQL, etc to hadoop and export data from hadoop file system to relational databases using specialized connectors [1].

Sqoop Command for exporting data from oracle:

sqoop import --connect jdbc:oracle:thin:@oracle.example.com:1521/ORACLE --username SQOOP --password sqoop --table schema_name.table_name

B. What is Apache Spark:

Apache Spark is an open-source unified analytics engine for large-scale data processing, with built-in modules for streaming, SQL, machine learning, and graph processing. Spark's in-memory computation capabilities make it significantly faster than traditional disk-based processing frameworks like Hadoop MapReduce. Spark provides a comprehensive set of APIs centered around data processing and real-time stream processing, making it highly efficient and flexible for a variety of big data applications.

C. What is Apache Hive:

Apache Hive is a data warehouse infrastructure built on top of Hadoop for providing data summarization, query, and analysis. It allows SQL-like queries (HiveQL) to be executed on large datasets stored in HDFS, making it easier for users to manage and analyze big data without extensive programming knowledge.

These tools, when integrated, provide a powerful solution for managing data migration and processing tasks, ensuring data integrity, and optimizing performance in big data environments.

PROBLEM STATEMENT

I work for an oil and gas company where we use Apache Sqoop and Spark for daily data loads from Oracle to MemSQL Database. Apache Hive serves as an external table [3] for the data copied through Sqoop and Spark utilities onto the MapR file system. We utilize Apache Spark to load tables that are smaller in size and have fewer columns, with record counts less than 500K. For tables with record counts exceeding 500K, we use Apache Sqoop because it handles parallelism efficiently for tables with millions of records. Each of these utilities creates a folder in Hive.

We encountered several issues during the data loads, including:

A. **Empty Directories:** When a Sqoop job failed or even in case of non-failure, there were instances sqoop load resulted in an empty directory on the MapR file system, causing the corresponding table in MemSQL to be empty.

B. **Sticky Bit Issues:** There were instances where the folders on the MapR file system had the sticky bit set, which also resulted in the tables being empty.

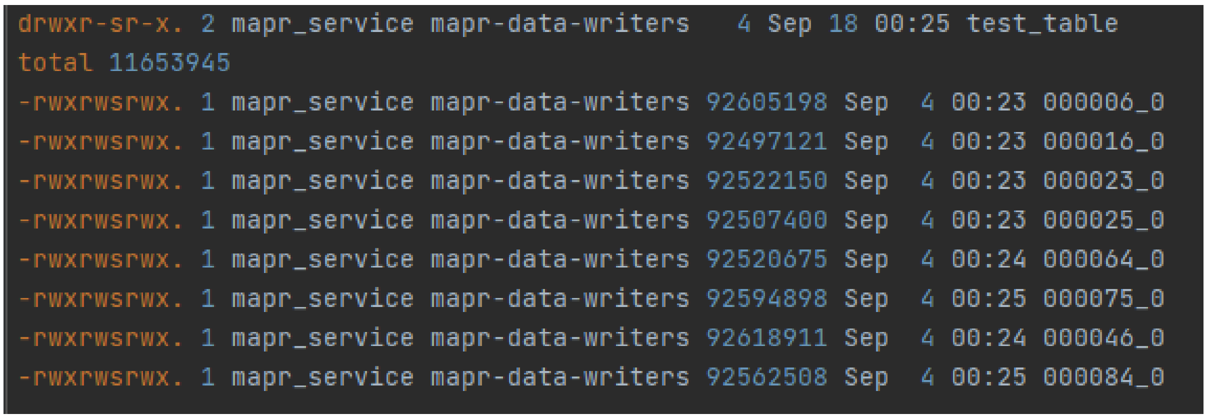
C. **Data Discrepancies:** After upgrading our Oracle source system to an Exadata environment, the MapR file system began to contain more data than what was actually read from the source system, leading to inconsistencies.

SOLUTION IMPLEMENTED

After thorough analysis, the following solutions were implemented to address each of the reported issues.

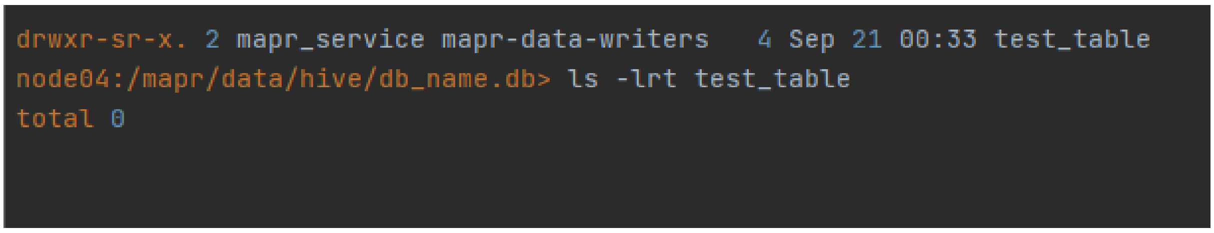
A. Empty Directories:

When a Sqoop load completes successfully, all the data files for the table being loaded are present in the directory. Refer to the screenshot below:



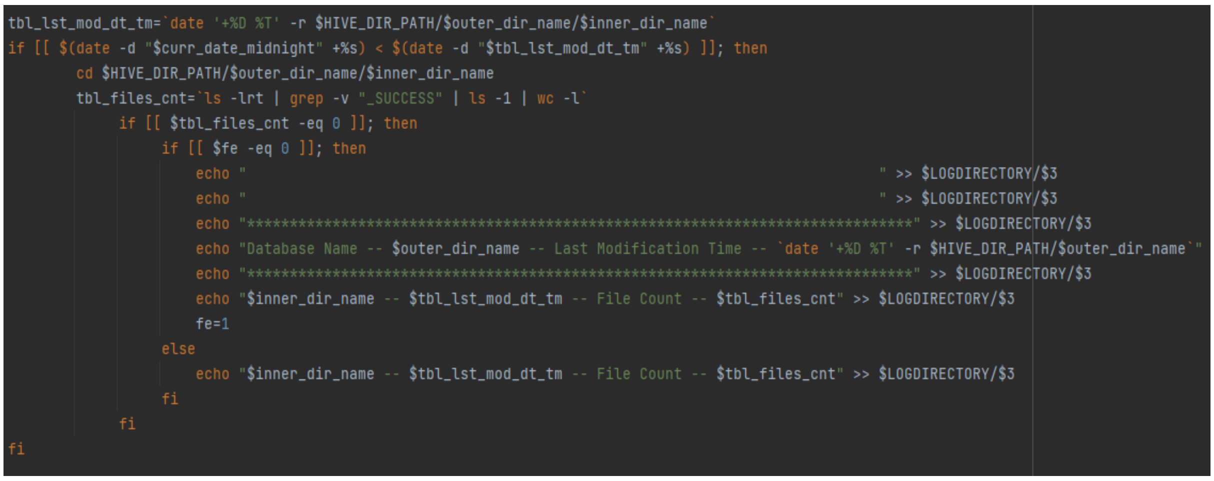
```
drwxr-sr-x. 2 mapr_service mapr-data-writers    4 Sep 18 00:25 test_table
total 11653945
-rwxrwsrwx. 1 mapr_service mapr-data-writers 92605198 Sep  4 00:23 000006_0
-rwxrwsrwx. 1 mapr_service mapr-data-writers 92497121 Sep  4 00:23 000016_0
-rwxrwsrwx. 1 mapr_service mapr-data-writers 92522150 Sep  4 00:23 000023_0
-rwxrwsrwx. 1 mapr_service mapr-data-writers 92507400 Sep  4 00:23 000025_0
-rwxrwsrwx. 1 mapr_service mapr-data-writers 92520675 Sep  4 00:24 000064_0
-rwxrwsrwx. 1 mapr_service mapr-data-writers 92594898 Sep  4 00:25 000075_0
-rwxrwsrwx. 1 mapr_service mapr-data-writers 92618911 Sep  4 00:24 000046_0
-rwxrwsrwx. 1 mapr_service mapr-data-writers 92562508 Sep  4 00:25 000084_0
```

When a Sqoop load fails or other issues arise, the directory may become empty, as shown in the screenshot below:



```
drwxr-sr-x. 2 mapr_service mapr-data-writers    4 Sep 21 00:33 test_table
node04:/mapr/data/hive/db_name.db> ls -lrt test_table
total 0
```

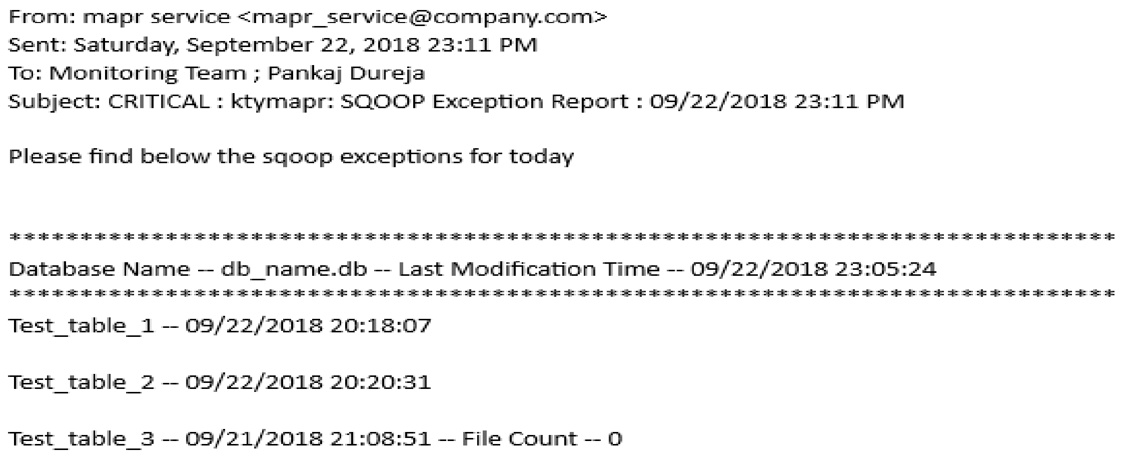
A shell script was created to run after the Sqoop loads. This script checks if any of the database directories are empty. If they are not empty, it then loops [5] through each table to verify whether it is empty. A consolidated report is generated and sent via email to identify any Sqoop exceptions.



```
tbl_lst_mod_dt_tm=`date '+%D %T' -r $HIVE_DIR_PATH/$outer_dir_name/$inner_dir_name`
if [[ $(date -d "$curr_date_midnight" +%s) < $(date -d "$tbl_lst_mod_dt_tm" +%s) ]]; then
        cd $HIVE_DIR_PATH/$outer_dir_name/$inner_dir_name
        tbl_files_cnt=`ls -lrt | grep -v "_SUCCESS" | ls -1 | wc -l`
            if [[ $tbl_files_cnt -eq 0 ]]; then
                if [[ $fe -eq 0 ]]; then
                    echo "                                                                          " >> $LOGDIRECTORY/$3
                    echo "                                                                          " >> $LOGDIRECTORY/$3
                    echo "******************************************************************************" >> $LOGDIRECTORY/$3
                    echo "Database Name -- $outer_dir_name -- Last Modification Time -- `date '+%D %T' -r $HIVE_DIR_PATH/$outer_dir_name`"
                    echo "******************************************************************************" >> $LOGDIRECTORY/$3
                    echo "$inner_dir_name -- $tbl_lst_mod_dt_tm -- File Count -- $tbl_files_cnt" >> $LOGDIRECTORY/$3
                    fe=1
                else
                    echo "$inner_dir_name -- $tbl_lst_mod_dt_tm -- File Count -- $tbl_files_cnt" >> $LOGDIRECTORY/$3
                fi
            fi
fi
```

Email notifications are sent to the support team to address any Sqoop exceptions that occur. The email specifies which tables within the database were not refreshed today based on the file count or the last modification time. If the last modification time is more than 24 hours old, it indicates that the table has not been refreshed.

[1]. Sqoop Exception Email



From: mapr service <mapr_service@company.com>
Sent: Saturday, September 22, 2018 23:11 PM
To: Monitoring Team ; Pankaj Dureja
Subject: CRITICAL : ktymapr: SQOOP Exception Report : 09/22/2018 23:11 PM

Please find below the sqoop exceptions for today

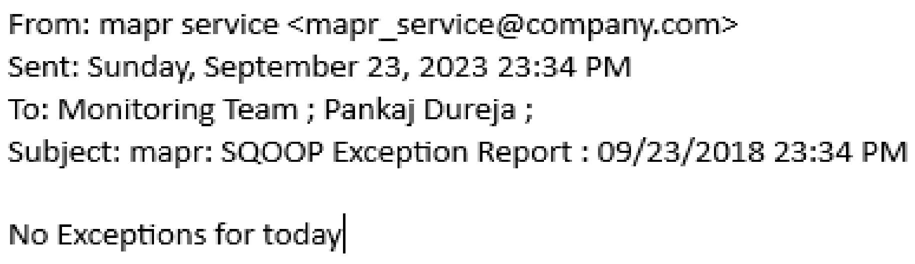
```
*****************************************************************************
Database Name -- db_name.db -- Last Modification Time -- 09/22/2018 23:05:24
*****************************************************************************
Test_table_1 -- 09/22/2018 20:18:07

Test_table_2 -- 09/22/2018 20:20:31

Test_table_3 -- 09/21/2018 21:08:51 -- File Count -- 0
```

If there are no exceptions, it means that all tables have been successfully refreshed.

[2]. Sqoop No Exception Email



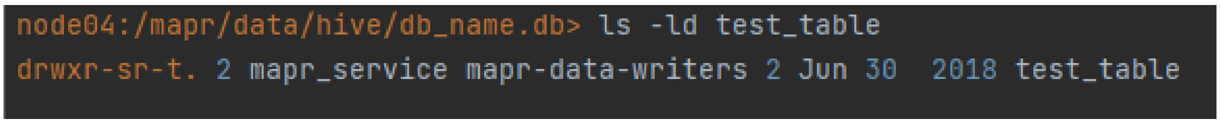
From: mapr service <mapr_service@company.com>
Sent: Sunday, September 23, 2023 23:34 PM
To: Monitoring Team ; Pankaj Dureja ;
Subject: mapr: SQOOP Exception Report : 09/23/2018 23:34 PM

No Exceptions for today

B. Sticky Bit Issues:

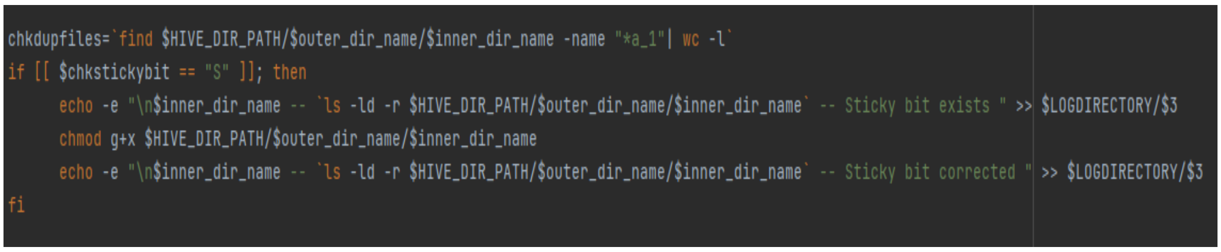
Check if a folder has a sticky bit set. This can be done using the following shell command, which is included in the script to correct and fix it. If the sticky bit [5] is present, it is removed with the command shown below.

[1]. Existence of sticky bit



```
node04:/mapr/data/hive/db_name.db> ls -ld test_table
drwxr-sr-t. 2 mapr_service mapr-data-writers 2 Jun 30  2018 test_table
```

[2]. Code to remove sticky bit

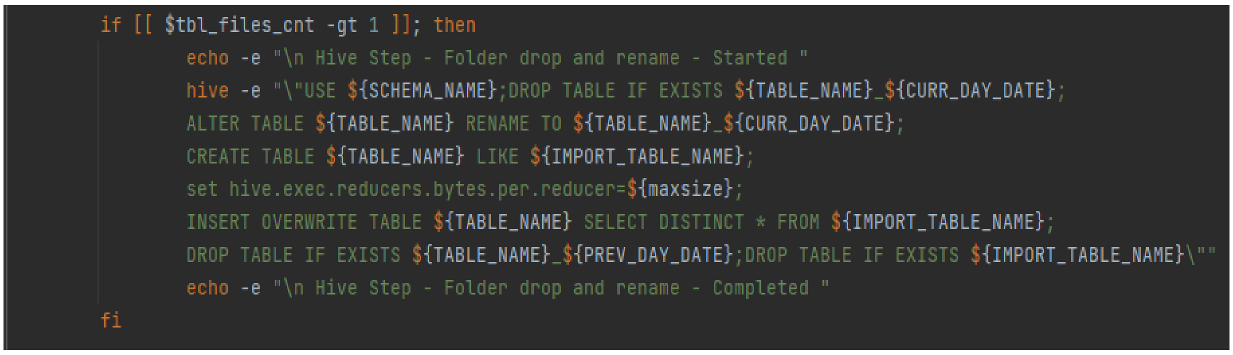


```
chkdupfiles=`find $HIVE_DIR_PATH/$outer_dir_name/$inner_dir_name -name "*a_1"| wc -l`
if [[ $chkstickybit == "S" ]]; then
      echo -e "\n$inner_dir_name -- `ls -ld -r $HIVE_DIR_PATH/$outer_dir_name/$inner_dir_name` -- Sticky bit exists " >> $LOGDIRECTORY/$3
      chmod g+x $HIVE_DIR_PATH/$outer_dir_name/$inner_dir_name
      echo -e "\n$inner_dir_name -- `ls -ld -r $HIVE_DIR_PATH/$outer_dir_name/$inner_dir_name` -- Sticky bit corrected " >> $LOGDIRECTORY/$3
fi
```

C. Data Discrepancies:

After upgrading the Oracle source system to the Exadata version, the table in Hive began showing higher row counts than the source system table. To address this, a de-duplication process was implemented using the Hive command `**INSERT OVERWRITE** `[4]

In Hive, `**INSERT OVERWRITE**`[4] is a command that overwrites the existing data in a table or partition with the new data resulting from the query and the duplicates are removed by **DISTINCT *** in select clause. Please refer to the screen shot below:



```
if [[ $tbl_files_cnt -gt 1 ]]; then
        echo -e "\n Hive Step - Folder drop and rename - Started "
        hive -e "\"USE ${SCHEMA_NAME};DROP TABLE IF EXISTS ${TABLE_NAME}_${CURR_DAY_DATE};
        ALTER TABLE ${TABLE_NAME} RENAME TO ${TABLE_NAME}_${CURR_DAY_DATE};
        CREATE TABLE ${TABLE_NAME} LIKE ${IMPORT_TABLE_NAME};
        set hive.exec.reducers.bytes.per.reducer=${maxsize};
        INSERT OVERWRITE TABLE ${TABLE_NAME} SELECT DISTINCT * FROM ${IMPORT_TABLE_NAME};
        DROP TABLE IF EXISTS ${TABLE_NAME}_${PREV_DAY_DATE};DROP TABLE IF EXISTS ${IMPORT_TABLE_NAME}\""
        echo -e "\n Hive Step - Folder drop and rename - Completed "
fi
```

POTENTIAL EXTENDED USE CASES

The solutions implemented to address issues in the data transfer process between Oracle and Hive using Apache Sqoop and Spark can be extended to various other scenarios:

[1]. **Cross-Industry Data Migration:** The methodology can be applied to other industries such as finance, healthcare, and retail, where large-scale data migrations from relational databases to big data platforms are common. Similar scripts and error handling mechanisms can be adapted to ensure data integrity during migration.

[2]. **Real-time Data Processing:** Implementing real-time data processing using Spark Streaming can enhance the system's ability to handle real-time data loads alongside batch processing. This extension would allow for continuous monitoring and real-time analytics, crucial for time-sensitive decision-making.

[3]. **Advanced Analytics and Machine Learning:** The cleaned and deduplicated data in Hive can serve as a reliable source for machine learning models and advanced analytics.

[4]. **Cloud Migration:** The process can be adapted for cloud environments, leveraging cloud storage solutions like AWS S3. Using cloud-based big data services such as Amazon EMR can further optimize the data processing pipeline.

IMPACT

[1]. **Improved Data Integrity:** By addressing empty directories and sticky bit issues, the integrity of data loaded into Hive and MemSQL is significantly enhanced. Consistent data ensures accurate analytics and reporting, crucial for business decisions.

[2]. **Operational Efficiency:** Automated scripts and email notifications reduce manual intervention, allowing the support team to focus on critical issues. The system's reliability increases, reducing downtime and improving overall operational efficiency.

[3]. **Scalability:** The ability to handle large datasets efficiently using Sqoop and Spark ensures the system can scale as data volumes grow. The framework is adaptable to new data sources and larger datasets, supporting the company's future growth.

[4]. **Cost Savings:** Reduced manual intervention and improved data integrity lead to cost savings by minimizing errors and operational overhead. Efficient data processing reduces resource utilization, further contributing to cost savings.

SCOPE

The scope of the current implementation includes - Automating the data transfer process from Oracle to Hive and MemSQL, ensuring regular and reliable data updates. Continuous monitoring and error handling through automated scripts and notifications.

CONCLUSION

The successful implementation of the described solutions for data transfer from Oracle to Hive using Apache Sqoop and Spark demonstrates a robust approach to managing big data workflows. By addressing issues such as empty directories, sticky bit problems, and data discrepancies, the process ensures data integrity, operational efficiency, and scalability. These solutions can be extended to various other use cases and industries, highlighting their versatility and effectiveness. Future enhancements could include real-time data processing and cloud integration, further optimizing the data pipeline and supporting advanced analytics initiatives.

REFERENCES

[1]. Kathleen Ting, Jarek Jarcec Cecho (2013). Chapter [7] Apache Sqoop Cookbook: Specialized Connectors: Importing from Oracle, pp. 63-70.
[2]. Muhammad Asif Abbasi. Learning Apache Spark 2: Chapter [1] Architecture and Installation: Apache spark architecture overview, pp. 9.
[3]. Edward Capriolo, Dean Wampler & Jason Rutherglen. (Second Edition 2015). Programming Hive. O'REILLY. Chapter [4]: [HiveQL: Data Definition] External Tables, pp. 56.
[4]. Edward Capriolo, Dean Wampler & Jason Rutherglen. (Second Edition 2015). Programming Hive. O'REILLY. Chapter [4]: [HiveQL: Data Manipulation] Inserting Data into Tables from Queries, pp. 73.
[5]. Mokhtar Ebrahim, Andrew Mallet. (Second Edition 2018). Mastering Linux Shell Scripting. PACKT. Chapter [6]: [Iterating with loops] for loops and Counting Directories and files, pp. 103-108.
[6]. Linux sticky bit available at https://www.thegeekstuff.com/2013/02/sticky-bit/
[7]. Sqoop User Guide. Available at
[8]. https://sqoop.apache.org/docs/1.4.6/SqoopUserGuide.html#_introduction
[9]. Apache Spark 2.1.1. Available at https://spark.apache.org/docs/2.1.1/
[10]. Hive Language Manual. Available at http://wiki.apache.org/hadoop/Hive/LanguageManual.