SQOOP and Hive Usecases 2023-CASPEX

Hive Sqoop Integration usecases:

1. Start mysql service

2. Login to mysql using root

3. Execute the sql file given below in Mysql to load the customer data
Note: The required files are attached in the links provided in the top of this page:

```
source /home/hduser/custpayments_ORIG.sql
```

4. Write sqoop command to import data from customers and products table with 2 mappers, with enclosed by " (As we have ',' in the data itself we are importing in sqoop using enclosed by option).

```
sqoop import --connect jdbc:mysql://localhost/custpayments --username root --password root -table customers -m 2 --split-by customernumber --target-dir /user/hduser/custdata/ --delete-target-dir --enclosed-by '\"';
```

5. Create a hive external table and load the sqoop imported data to the hive table called custmaster. As we have ',' in the data itself we are using quotedchar option below with the csv serde option as given below as example, create the table with all columns.

```
create external table custmaster (customerNumber int,customername string,contactlastname string,contactfirstname string)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
 "separatorChar" = ",",
 "quoteChar" = "\"")
LOCATION '/user/hduser/custdata/';
```

6. Copy the payments.txt into hdfs location /user/hduser/paymentsdata/

```
hadoop fs -mkdir -p /user/hduser/paymentsdata
hadoop fs -put payments.txt /user/hduser/paymentsdata
```

7. Create an external table to point the imported payments data location /user/hduser/paymentsdata/.

8. Create another external table called cust_payments in avro format and load using insert select option from the custmaster.

```
create external table cust_payments(custno int,contactfirstname string,contactlastname string)
row format delimited fields terminated by '~'
stored as avro
location '/user/hduser/paymentsavro';
```

```
insert into table cust_payments select customernumber,contactfirstname,contactlastname from custmaster;
```

9. Create a view called custpayments_vw to only display customernumber,concatenated contactfirst and contactlastname,creditlimit, amount.

```
create view custpayments as select custno,upper(concat(contactfirstname,' ',contactlastname)) from cust_payments;
```

==========

1. Create a usecase dir

mkdir /home/hduser/hiveusecase

Please copy the custpayments_ORIG.sql and payments.txt into /home/hduser/hiveusecase

2. Ensure Hadoop Daemons, MYSQL, Hive remote metastore is running.

Usecase 1: DML operation in Hive, ETL/ELT with hive & Benchmarking in Hive

ETL Using Hive Queries and few functions

Hive can do ETL and ELT, lets explore how can we achieve several business logics in hive by loading staging tables, de-normalized tables with joins, concatenation, summation, aggregations, analytical queries etc.,

1. **Data Ingestion:**

Create a database called custdb.
Go inside custdb.
Create the below table in custdb and load customer data

hive>

```
Create database custdb;
use custdb;
create table customer(custno string, firstname string, lastname string, age int,profession
string)
row format delimited fields terminated by ',';

load data local inpath '/home/hduser/hive/data/custs' into table customer;
```

2. **Data Curation:**

Create cust transaction table and load the customer and transaction data by joining the 2 tables located in multiple databases such as retail and custdb prefixing schema name, here we are

denormalizing the tables into a single fat table with added ETL operations and stored as external table partitioned based on current date

```
create external table ext_cust_txn_part (custno int,fullname string,age int,profession
string,amount double,product string,spendby string,agecat varchar(100),modifiedamout float)
partitioned by (datadt date)
row format delimited fields terminated by ','
location '/user/hduser/custtxn';

insert into table ext_cust_txn_part partition(datadt)
select a.custno,upper(concat(a.firstname, ' ',a.lastname)),
a.age,a.profession,b.amount,b.product,b.spendby,
case when age<30 then 'low'
when age>=30 and age < 50 then 'middle'
when age>=50 then 'old'
else 'others' end as agecat,
case when spendby= 'credit' then b.amount+(b.amount*0.05) else b.amount end as
modifiedamount,current_date
from custdb.customer a JOIN retail.txnrecords b
ON a.custno = b.custno;


select * from ext_cust_txn_part limit 10;
```

3. Data Visualization/Analytics/Aggregation/Reporting/Discovery

Creating aggregation tables that will be considered as cube or deriving **KPIs** (Key Performance Indicator) or deriving Metrics and used for quick reporting purposes. Here we are creating a sequence number or a surrogate key column using olap functions like rownumber over and aggregating the age and amount in multiple dimensions with the addition of current_date also.

```
create external table cust_trxn_aggr1 (seqno int,product string,profession string,level string,sumamt
double, avgamount double,maxamt double,avgage int,currentdate date)
row format delimited fields terminated by ',';

insert overwrite table cust_trxn_aggr1
select row_number() over(),product,profession, agecat,
sum(amount),avg(amount),max(amount),avg(age),current_date()
from ext_cust_txn_part
where datadt=current_date
group by product,profession, agecat, current_date();


create external table cust_trxn_aggr2 (seqno int, profession string,level string,sumamt double,
avgamount double,maxamt double,avgage int,currentdate date)
row format delimited fields terminated by ',';
```

```
insert overwrite table cust_trxn_aggr2
select row_number() over(),profession, agecat,
sum(amount),avg(amount),max(amount),avg(age),current_date()
from ext_cust_txn_part
where datadt=current_date
group by profession, agecat, current_date();
```

Performing DML statements in Hive

Question is , can we do DML/ACID transactions in hive as HDFS doesn't support changes??
Ans: Hive is not for DML/ACID, but it supports in case if we need, but it is cosly to apply.

Try to update/delete the customer table created above, it won't work in general..

```
update customer
set profession='IT'
where custno= 4000001;

delete from customerdml
where custno= 4000002;
```

Solution :

1.Create another hive table with buckets, stored in orc format and transactional table properties true for the DML support

```
create table customerdml(custno string, firstname string, lastname string, age
int,profession string)
clustered by (custno) into 3 buckets
stored as orc
TBLPROPERTIES ('transactional'='true');

set hive.support.concurrency=true;
set hive.txn.manager=org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
```

3. Copy the data from customer to the customerdml table.

```
insert into customerdml select * from customer;
```

tablelocation/delta_1/3 buckets

4. Update

```
Select * from customerdml where custno=4000001;

update customerdml
set profession='IT'
where custno= 4000001;
```

tablelocation/delta_1/0 bucket (only updated data)

5. Delete

```
delete from customerdml where custno= 4000002;

Select * from customerdml where custno in (4000001,4000002);
```

Admin's Scope: schedule to run in off peak time

```
alter table customerdml compact 'major';
```

minor -> 20 delta to 2 delta

major -> 2 delta to 1 delta (final)

Benchmarking Hive using different file format storage:

The purpose of doing ***benchmarking*** is to identify the best functionality or the feature to be used by iterating with different options, here we are going to create textfile, orc, avro and parquet format tables to check the performance between all these tables and the data size it occupied.

```
create table staging_txn(txnno INT, txndate STRING, custno INT, amount DOUBLE, category STRING,
product STRING, city STRING, state STRING, spendby STRING)
row format delimited fields terminated by ',' lines terminated by '\n'
stored as textfile;
```

Note: the below load command is only possible for textfile table and we can't use load command for any other serialized format like orc/parquet/avro/json.

```
LOAD DATA LOCAL INPATH '/home/hduser/hive/data/txns' OVERWRITE INTO TABLE staging_txn;
```

Parquet file table:

```
create table txn_parquet(txnno INT, txndate STRING, custno INT, amount DOUBLE,category STRING,
product STRING, city STRING, state STRING, spendby STRING)
row format delimited fields terminated by ',' lines terminated by '\n'
stored as parquetfile;

Insert into table txn_parquet select txnno,txndate,custno,amount,category,
product,city,state,spendby from staging_txn;
```

Orc file table:

```
create table txn_orc(txnno INT, txndate STRING, custno INT, amount DOUBLE,
category STRING, product STRING, city STRING, state STRING, spendby STRING)
row format delimited fields terminated by ',' lines terminated by '\n'
stored as orcfile;

Insert into table txn_orc select txnno,txndate,custno,amount,category, product,city,state,spendby
from staging_txn;
```

Usecase 2: CSV Serde

Requirement:
Source provider is providing structured data from a DB or FS source, but the data may contain the delimiter what we are going to use.

Answer – By using string/varchar type defined for the whole data set we can handle this.

1. Login to Mysql and execute the sql file to load the custpayments table:

```
mysql> source /home/hduser/hiveusecases/custpayments_ORIG.sql
```

2. Write sqoop command to import data from customerpayments table with 2 mappers, with enclosed by " (As we have ',' in the data itself we are importing in sqoop using --enclosed-by option into the location /user/hduser/custpayments).

```
sqoop import --connect jdbc:mysql://localhost/custpayments --username root --password Root123$ -table
customers -m 2 \
--split-by customernumber --target-dir /user/hduser/custpayments --delete-target-dir --enclosed-by '\"';
```

3. Create a hive external table and load the sqoop imported data to the hive table called custpayments. As we have ',' in the data itself we are using quoted char option below with the csv serde option as given below as example, create the table with all columns.

```
create external table custmaster (customerNumber int,customername string,contactlastname
string,contactfirstname string)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar"     = "\"")
LOCATION '/user/hduser/custpayments/';
```

4. Copy the payments.txt into hdfs location /user/hduser/paymentsdata/ and Create an external table namely payments with customernumber, checknumber, paymentdate, amount columns to point the imported payments data.

```
hadoop fs -mkdir -p /user/hduser/paymentsdata/
```

hadoop fs -put -f /home/hduser/hiveusecases/payments.txt /user/hduser/paymentsdata/

5. Create an external table called cust_payments in avro format and load data by doing inner join of custmaster and payments tables, using insert select customernumber, contactfirstname,contactlastname,phone, creditlimit from custmaster and paymentdate, amount columns from payments table

Csv serde table join payments external table -> cust_payments (avro)

6. Create a **view** called custpayments_vw to only display customernumber,creditlimit,paymentdate and amount selected from cust_payments.

View Benefits:

1. View doesn't contains data, rather it stores only query.
2. Reducing the complexity of writing queries, because view stores complex queries.
3. Performance benefit – when create the view apply the performance accordingly (eg: use partition column in the filter)
4. Security – Enable only the columns/rows to the intended users.

7. Extract only customernumber,creditlimit,paymentdate and amount columns either using the above view/cust_payments table into hdfs location /user/hduser/custpaymentsexport with '|' delimiter.

Note: Achieve the above scenario using insert overwrite directory option or one more option to achieve this, try that out and let me know what is that?

Select customernumber,creditlimit,paymentdate and amount from cust_payments (hive table) > /user/hduser/custpaymentsexport with '|' delimiter

8. Export the data from the /user/hduser/custpaymentsexport location to mysql table called cust_payments using sqoop export with staging table option using records per statement 100 and mappers 3.