Session 08: Advanced Hive

Assignment 3

**Problem Statement**

Link: https://acadgild.com/blog/transactions-in-hive/

Refer the above given link for transactions in Hive and implement the operations given in the blog using

your own sample data set and send us the screenshot.

**Note: Hive 0.14 should be installed to implement the hive transaction property.**

**Solution:-**

## Transactions in Hive

Transactions in Hive are introduced in Hive 0.13, but they only partially fulfill the ACID properties like atomicity, consistency, durability, at the partition level. Here, Isolation can be provided by turning on one of the locking mechanisms available with zookeeper or in memory.

But in Hive 0.14, new API’s have been added to completely fulfill the ACID properties while performing any transaction.

Transactions are provided at the row-level in Hive 0.14. The different row-level transactions available in Hive 0.14 are as follows:

1. Insert
2. Delete
3. Update

**Row-level Transactions Available in Hive 0.14**

Let’s perform some row-level transactions available in Hive 0.14. Before creating a Hive table that supports transactions, the transaction features present in Hive needs to be turned on, as by default they are turned off.

The below properties needs to be set appropriately in ***hive shell***, order-wise to work with transactions in Hive:

**hive>**set hive.support.concurrency = true;

**hive>**set hive.enforce.bucketing = true;

**hive>**set hive.exec.dynamic.partition.mode = nonstrict;

**hive>**set hive.txn.manager = org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;

**hive>**set hive.compactor.initiator.on = true;

**hive>**set hive.compactor.worker.threads = 2;



## Creating a Table That Supports Hive Transactions

**Hive>** CREATE TABLE college(clg\_id int,clg\_name string,clg\_loc string) clustered by (clg\_id) into 5 buckets stored as orc TBLPROPERTIES('transactional'='true');



## Inserting Data into a Hive Table

|  |  |
| --- | --- |
|  | **Hive>** INSERT INTO table college values(1,'IIT-B','MUM'),(2,'NIT-T','TRICHY'),  (3,'NIFT','AHD'),(4,'OXFORD','UK'),(5,'MIT','USA'),(6,'BITS','PILANI'),(7,'IISC','HYD'); |

*Now, we have successfully inserted the data into the Hive table.*

*The contents of the table can be viewed using the command****select \* from college***



From the above image, we can see that the data has been inserted successfully into the table.

Now if we try to re-insert the same data again, it will be appended to the previous data as shown below:





## Updating the Data in Hive Table

|  |  |
| --- | --- |
|  | **Hive>** UPDATE college set clg\_id = 8 where clg\_id = 7; |

The above command is used to update a row in Hive table.



From the above image, we can see that we have received an error message. This means that the Update command is not supported on the columns that are bucketed.

In this table, we have bucketed the ***‘clg\_id’*** column and performing the Update operation on the same column, so we have go the error

**Now let’s perform the update operation on Non bucketed column**

|  |  |
| --- | --- |
|  | **Hive>** UPDATE college set clg\_name = 'IIT' where clg\_id = 6; |



We have successfully updated the data.

The updated data can be checked using the command ***select \* from college.***



We can see that the data has been updated successfully.

Now let’s perform the Delete operation on the same table.

## Deleting a Row from Hive Table

|  |  |
| --- | --- |
| **Hive>** | delete from college where clg\_id=5; |



We have now successfully deleted a row from the Hive table. This can be checked using the command **select \* from college.**



We can see that there is no row with***clg\_id =5***. This means that we have successfully deleted the row from the Hive table.

This is how the transactions or row-wise operations are performed in Hive.