Case_Study_-_II Customer_Transaction

Case Study Description

Let us take up the CUSTOMER and TRANSACTIONS table we have created in the

Let’s Do Together section. Let us solve the following use cases using these tables -

1. Find out the number of transaction done by each customer (These should be taken up in module 8 itself)

2. Create a new table called TRANSACTIONS_COUNT. This table should have fields - custid, fname and count. (Again to be done in module 8)

3. Now write a hive query in such a way that the query populates the data obtained in Step 1 above and populate the table in step 2 above. (This has to be done in module 9).

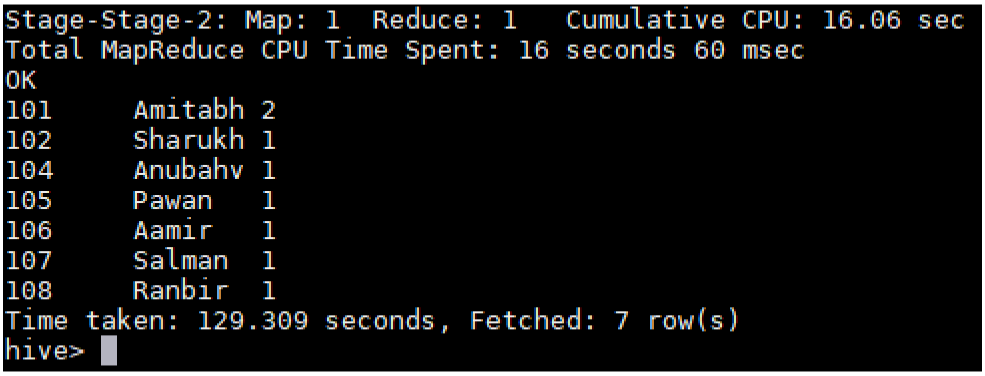
4. Now lets make the TRANSACTIONS_COUNT table Hbase complaint. In the sense, use Ser Des and Storage handler features of hive to change the TRANSACTIONS_COUNT table to be able to create a TRANSACTIONS table in Hbase. (This has to be done in module 10)

5. Now insert the data in TRANSACTIONS_COUNT table using the query in step 3 again, this should populate the Hbase TRANSACTIONS table automatically (This has to be done in module 10)

6. Now from the Hbase level, write the Hbase java API code to access and scan the TRANSACTIONS table data from java level.

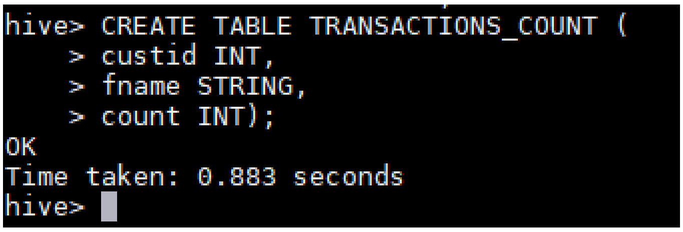
SOLUTION –

1. **select a.custid, a.fname, count(txnno) from CUSTOMER a join TRANSACTIONS b on a.custid=b.custno group by a.custid,a.fname;**



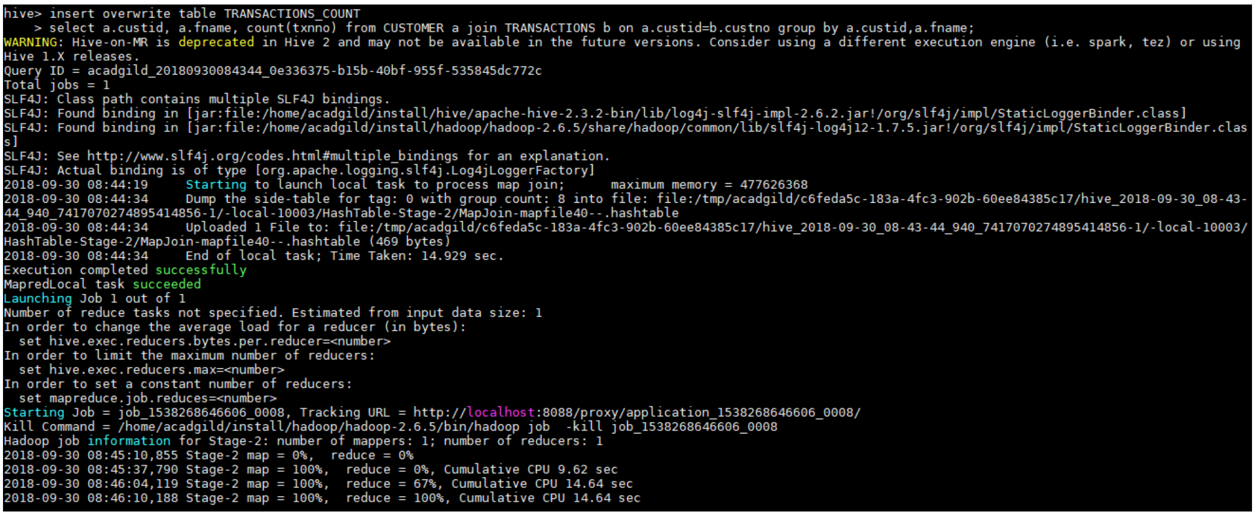
```
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 16.06 sec
Total MapReduce CPU Time Spent: 16 seconds 60 msec
OK
101     Amitabh 2
102     Sharukh 1
104     Anubahv 1
105     Pawan   1
106     Aamir   1
107     Salman  1
108     Ranbir  1
Time taken: 129.309 seconds, Fetched: 7 row(s)
hive>
```

2. **CREATE TABLE TRANSACTIONS_COUNT (**
   custid INT, fname STRING, count INT);

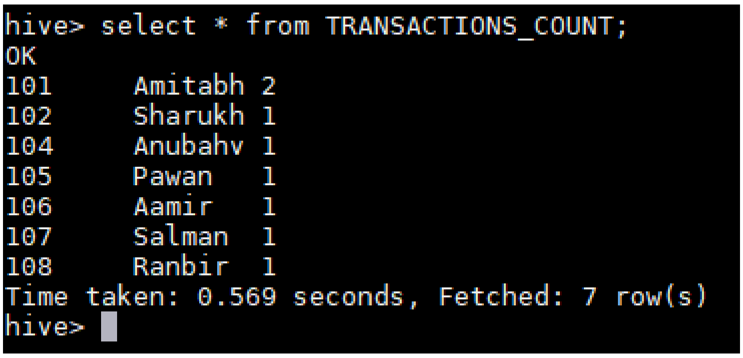


```
hive> CREATE TABLE TRANSACTIONS_COUNT (
    > custid INT,
    > fname STRING,
    > count INT);
OK
Time taken: 0.883 seconds
hive> 
```

3. **INSERT OVERWRITE table TRANSACTIONS_COUNT**
   select a.custid, a.fname, count(txnno) from CUSTOMER a join TRANSACTIONS b on a.custid=b.custno group by a.custid,a.fname;

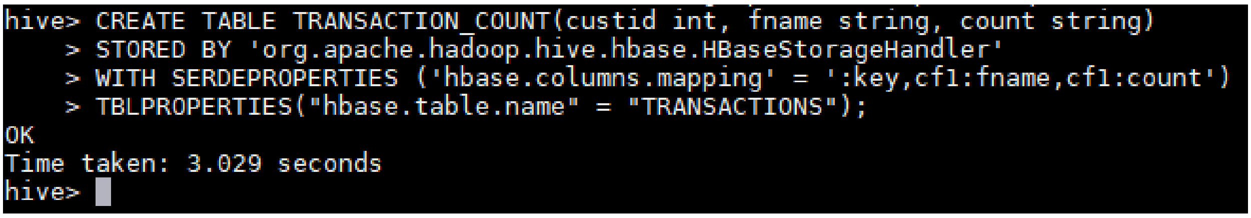


```
hive> insert overwrite table TRANSACTIONS_COUNT
    > select a.custid, a.fname, count(txnno) from CUSTOMER a join TRANSACTIONS b on a.custid=b.custno group by a.custid,a.fname;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using
Hive 1.X releases.
Query ID = acadgild_20180930084344_0e336375-b15b-40bf-955f-535845dc772c
Total jobs = 1
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/acadgild/install/hive/apache-hive-2.3.2-bin/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/acadgild/install/hadoop/hadoop-2.6.5/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.clas
s]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
2018-09-30 08:44:19     Starting to launch local task to process map join;      maximum memory = 477626368
2018-09-30 08:44:34     Dump the side-table for tag: 0 with group count: 8 into file: file:/tmp/acadgild/c6feda5c-183a-4fc3-902b-60ee84385c17/hive_2018-09-30_08-43-
44_940_7417070274895414856-1/-local-10003/HashTable-Stage-2/MapJoin-mapfile40--.hashtable
2018-09-30 08:44:34     Uploaded 1 File to: file:/tmp/acadgild/c6feda5c-183a-4fc3-902b-60ee84385c17/hive_2018-09-30_08-43-44_940_7417070274895414856-1/-local-10003/
HashTable-Stage-2/MapJoin-mapfile40--.hashtable (469 bytes)
2018-09-30 08:44:34     End of local task; Time Taken: 14.929 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1538268646606_0008, Tracking URL = http://localhost:8088/proxy/application_1538268646606_0008/
Kill Command = /home/acadgild/install/hadoop/hadoop-2.6.5/bin/hadoop job  -kill job_1538268646606_0008
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2018-09-30 08:45:10,855 Stage-2 map = 0%,  reduce = 0%
2018-09-30 08:45:37,790 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 9.62 sec
2018-09-30 08:46:04,119 Stage-2 map = 100%,  reduce = 67%, Cumulative CPU 14.64 sec
2018-09-30 08:46:10,188 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 14.64 sec
```

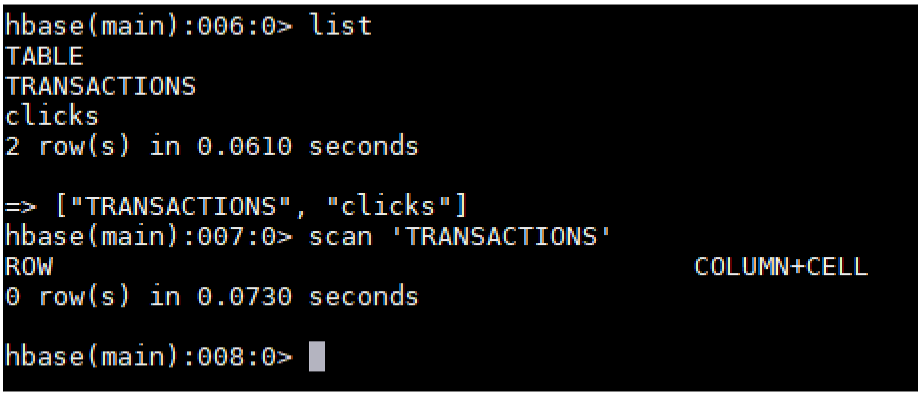


```
hive> select * from TRANSACTIONS_COUNT;
OK
101     Amitabh 2
102     Sharukh 1
104     Anubahv 1
105     Pawan   1
106     Aamir   1
107     Salman  1
108     Ranbir  1
Time taken: 0.569 seconds, Fetched: 7 row(s)
hive> 
```

4. **CREATE TABLE TRANSACTION_COUNT(custid int, fname string, count string)**
   STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
   WITH SERDEPROPERTIES ('hbase.columns.mapping' = ':key,cf1:fname,cf1:count')
   TBLPROPERTIES("hbase.table.name" = "TRANSACTIONS");

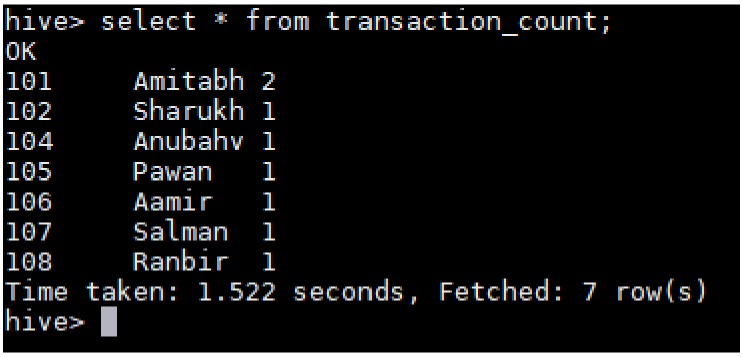


```
hive> CREATE TABLE TRANSACTION_COUNT(custid int, fname string, count string)
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ('hbase.columns.mapping' = ':key,cf1:fname,cf1:count')
    > TBLPROPERTIES("hbase.table.name" = "TRANSACTIONS");
OK
Time taken: 3.029 seconds
hive>
```

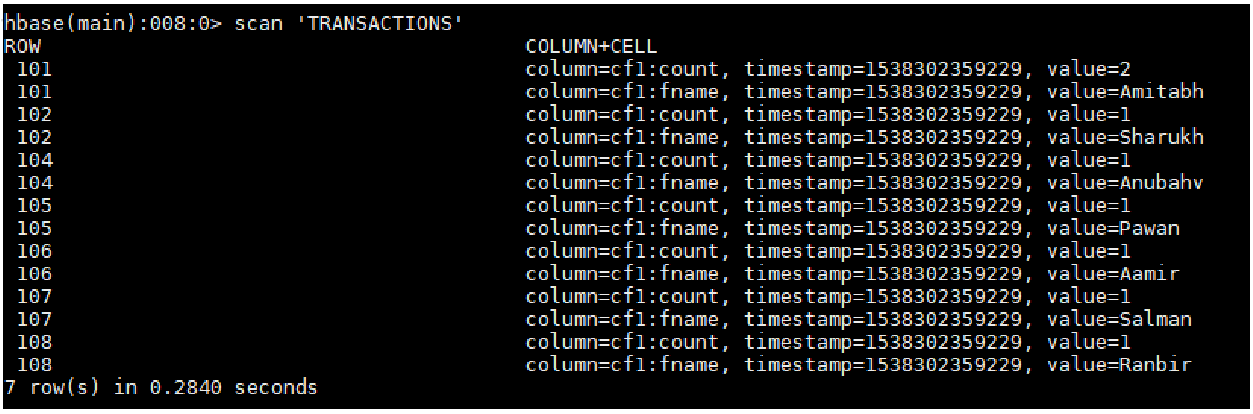


```
hbase(main):006:0> list
TABLE
TRANSACTIONS
clicks
2 row(s) in 0.0610 seconds

=> ["TRANSACTIONS", "clicks"]
hbase(main):007:0> scan 'TRANSACTIONS'
ROW                                      COLUMN+CELL
0 row(s) in 0.0730 seconds

hbase(main):008:0>
```

5. **INSERT OVERWRITE table TRANSACTION_COUNT**
   select a.custid, a.fname, count(txnno) from CUSTOMER a join TRANSACTIONS b on a.custid=b.custno group by a.custid,a.fname;

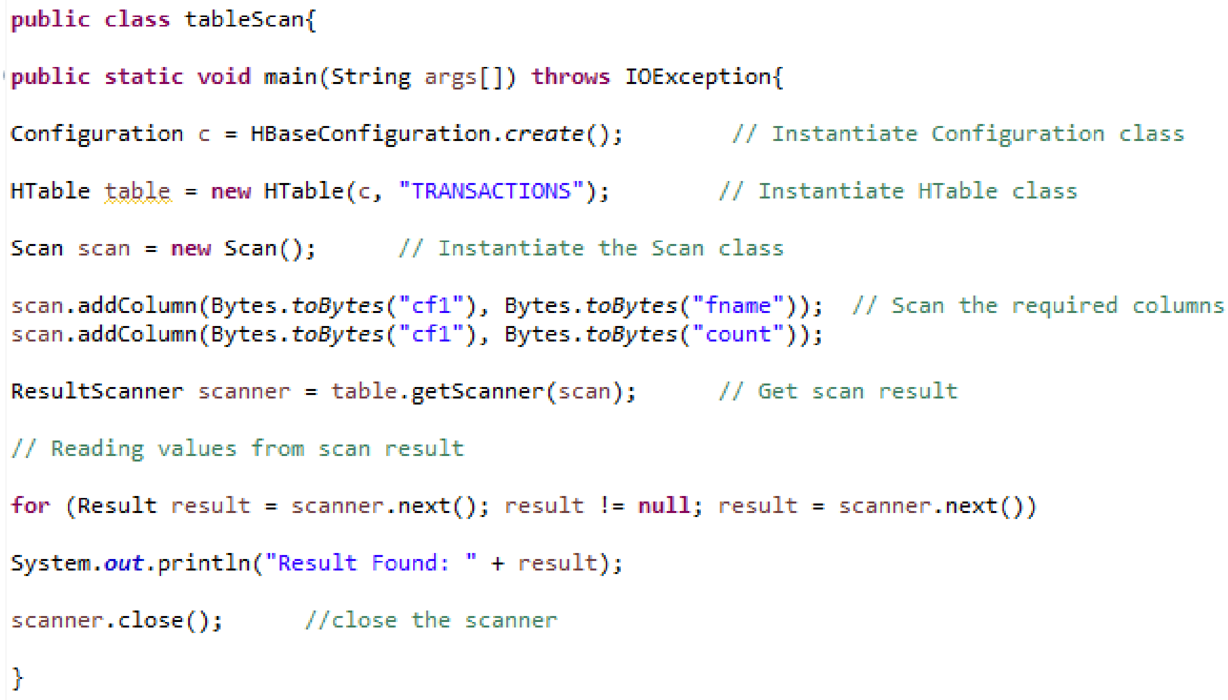


```
hive> select * from transaction_count;
OK
101     Amitabh 2
102     Sharukh 1
104     Anubahv 1
105     Pawan   1
106     Aamir   1
107     Salman  1
108     Ranbir  1
Time taken: 1.522 seconds, Fetched: 7 row(s)
hive>
```



```
hbase(main):008:0> scan 'TRANSACTIONS'
ROW                                        COLUMN+CELL
 101                                       column=cf1:count, timestamp=1538302359229, value=2
 101                                       column=cf1:fname, timestamp=1538302359229, value=Amitabh
 102                                       column=cf1:count, timestamp=1538302359229, value=1
 102                                       column=cf1:fname, timestamp=1538302359229, value=Sharukh
 104                                       column=cf1:count, timestamp=1538302359229, value=1
 104                                       column=cf1:fname, timestamp=1538302359229, value=Anubahv
 105                                       column=cf1:count, timestamp=1538302359229, value=1
 105                                       column=cf1:fname, timestamp=1538302359229, value=Pawan
 106                                       column=cf1:count, timestamp=1538302359229, value=1
 106                                       column=cf1:fname, timestamp=1538302359229, value=Aamir
 107                                       column=cf1:count, timestamp=1538302359229, value=1
 107                                       column=cf1:fname, timestamp=1538302359229, value=Salman
 108                                       column=cf1:count, timestamp=1538302359229, value=1
 108                                       column=cf1:fname, timestamp=1538302359229, value=Ranbir
7 row(s) in 0.2840 seconds
```

6.



```
public class tableScan{

public static void main(String args[]) throws IOException{

Configuration c = HBaseConfiguration.create();        // Instantiate Configuration class

HTable table = new HTable(c, "TRANSACTIONS");        // Instantiate HTable class

Scan scan = new Scan();      // Instantiate the Scan class

scan.addColumn(Bytes.toBytes("cf1"), Bytes.toBytes("fname"));  // Scan the required columns
scan.addColumn(Bytes.toBytes("cf1"), Bytes.toBytes("count"));

ResultScanner scanner = table.getScanner(scan);      // Get scan result

// Reading values from scan result

for (Result result = scanner.next(); result != null; result = scanner.next())

System.out.println("Result Found: " + result);

scanner.close();      //close the scanner

}
```