# Display Schema Name

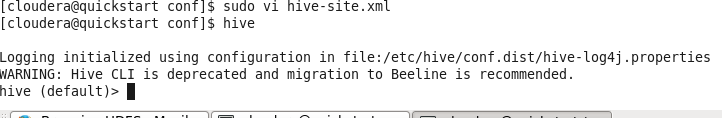
To display the schema name in hive prompt, you can use

set hive.cli.print.current.db=true; but this is applicable for current session as you exit this will lost

To make it permanent we need to edit /etc/hive/conf/hive-site.xml file

Just add below property

|  |
| --- |
| <property>  <name>hive.cli.print.current.db</name>  <value>true</value>  </property> |



You can create .hiverc file in /etc/hive/conf folder and add property

|  |
| --- |
| set hive.cli.print.current.db=true |

Priority => command line set > .hiverc >hive-site.xml

# Print Header

To check the create script of any table

|  |
| --- |
| show create table <table\_name>; |

To print the header value in result ( column name in case of select statement) add below property in hive-site.xml

|  |
| --- |
| <property>  <name>hive.cli.print.header</name>  <value>true</value>  </property> |

# Drop all tables

|  |
| --- |
| hive -e 'use testdb;show tables' | xargs -I '{}' hive -e 'use testdb;drop table {}' |

# Drop database

|  |
| --- |
| DROP DATABASE IF EXISTS testdb CASCADE; |

# Create different file format table from text table

|  |
| --- |
| CREATE TABLE Carrier\_Claims\_Parquet LIKE Carrier\_Claims STORED AS PARQUET; |

|  |
| --- |
| CREATE TABLE Carrier\_Claims\_Parquet LIKE Carrier\_Claims STORED AS ORC; |

|  |
| --- |
| CREATE TABLE Carrier\_Claims\_Parquet LIKE Carrier\_Claims STORED AS RCFILE; |

# Show Table properties

show tblproperties myTableName, It gives below result:

numFiles 12

numRows 1688092

rawDataSize 934923162

totalSize 936611254

# Insert data in partition table from another table

## Insert using where clause (static partition)

Here we need to define the exact partition value and also need to define all columns in select clause except partition column that we need to define in where criteria

|  |
| --- |
| insert into Beneficiary\_Summary\_PARQUET PARTITION (year=2008) select DESYNPUF\_ID,BENE\_BIRTH\_DT,BENE\_DEATH\_DT,BENE\_SEX\_IDENT\_CD,BENE\_RACE\_CD,BENE\_ESRD\_IND,SP\_STATE\_CODE,BENE\_COUNTY\_CD,BENE\_HI\_CVRAGE\_TOT\_MONS,BENE\_SMI\_CVRAGE\_TOT\_MONS,BENE\_HMO\_CVRAGE\_TOT\_MONS,PLAN\_CVRG\_MOS\_NUM,SP\_ALZHDMTA,SP\_CHF,SP\_CHRNKIDN,SP\_CNCR,SP\_COPD,SP\_DEPRESSN,SP\_DIABETES,SP\_ISCHMCHT,SP\_OSTEOPRS,SP\_RA\_OA,SP\_STRKETIA,MEDREIMB\_IP,BENRES\_IP,PPPYMT\_IP,MEDREIMB\_OP,BENRES\_OP,PPPYMT\_OP,MEDREIMB\_CAR,BENRES\_CAR,PPPYMT\_CAR from Beneficiary\_Summary where year=2008; |

## Insert all partition column without where clause (dynamic partition)

Instead of define each partition in multiple insert query we can make a single insert query without defining the exact value, see below query

|  |
| --- |
| set hive.exec.dynamic.partition.mode=nonstrict;  set hive.exec.dynamic.partition = true; |

|  |
| --- |
| insert into Beneficiary\_Summary\_PARQUET PARTITION (year) select \* from Beneficiary\_Summary ; |

Or

|  |
| --- |
| From Beneficiary\_Summary insert into Beneficiary\_Summary\_PARQUET PARTITION (year) |

# Insert into local directory from hive

We can load data into local file system directory from hive table

Syntax

|  |
| --- |
| Standard syntax:  INSERT OVERWRITE [LOCAL] DIRECTORY directory1  [ROW FORMAT row\_format] [STORED AS file\_format] (Note: Only available starting with Hive 0.11.0)  SELECT ... FROM ...    Hive extension (multiple inserts):  FROM from\_statement  INSERT OVERWRITE [LOCAL] DIRECTORY directory1 select\_statement1  [INSERT OVERWRITE [LOCAL] DIRECTORY directory2 select\_statement2] ...      row\_format  : DELIMITED [FIELDS TERMINATED BY char [ESCAPED BY char]] [COLLECTION ITEMS TERMINATED BY char]  [MAP KEYS TERMINATED BY char] [LINES TERMINATED BY char]  [NULL DEFINED AS char] (Note: Only available starting with Hive 0.13) |

Example:

|  |
| --- |
| hive> INSERT OVERWRITE LOCAL DIRECTORY '/root/tempdir' select \* from temptableapplication limit 10;  Query ID = root\_20170222090707\_5ed481dc-4242-4edc-ab3b-4eaac2fa5313  Total jobs = 1  Launching Job 1 out of 1  Number of reduce tasks determined at compile time: 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\_1484894548299\_0047, Tracking URL = http://mac55:8088/proxy/application\_1484894548299\_0047/  Kill Command = /opt/cloudera/parcels/CDH-5.8.2-1.cdh5.8.2.p0.3/lib/hadoop/bin/hadoop job -kill job\_1484894548299\_0047  Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 1  2017-02-22 09:08:07,490 Stage-1 map = 0%, reduce = 0%  2017-02-22 09:08:39,424 Stage-1 map = 33%, reduce = 0%, Cumulative CPU 29.43 sec  2017-02-22 09:08:54,757 Stage-1 map = 67%, reduce = 0%, Cumulative CPU 38.28 sec  2017-02-22 09:09:54,949 Stage-1 map = 67%, reduce = 0%, Cumulative CPU 50.15 sec  2017-02-22 09:09:55,967 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 50.5 sec  2017-02-22 09:10:01,065 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 51.42 sec  MapReduce Total cumulative CPU time: 51 seconds 420 msec  Ended Job = job\_1484894548299\_0047  Copying data to local directory /root/tempdir  MapReduce Jobs Launched:  Stage-Stage-1: Map: 3 Reduce: 1 Cumulative CPU: 51.42 sec HDFS Read: 132172178 HDFS Write: 4500 SUCCESS  Total MapReduce CPU Time Spent: 51 seconds 420 msec  OK  Time taken: 132.949 seconds |

Check if file is loaded into local direcotry

|  |
| --- |
| [root@mac55 ~]# cd /root/tempdir/  [root@mac55 tempdir]# ls -l  total 8  -rw-r--r-- 1 root root 4500 Feb 22 09:10 000000\_0  [root@mac55 tempdir]# cat 000000\_0  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_5mstsc.exeC:\Windows\system32\mstsc.exeMicrosoft® Windows® Operating SystemMicrosoft CorporationRemote Desktop ConnectionRemote Desktop Connection0.02016-05-25 18:01:53.8482016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_2Explorer.EXEC:\Windows\Explorer.EXEMicrosoft® Windows® Operating SystemMicrosoft CorporationWindows ExplorerSearch Pane0.02016-05-25 18:01:43.6132016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_19mstsc.exeC:\Windows\system32\mstsc.exeMicrosoft® Windows® Operating SystemMicrosoft CorporationRemote Desktop ConnectionRemote Desktop Connection0.02016-05-25 18:05:39.9152016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_16mstsc.exeC:\Windows\system32\mstsc.exeMicrosoft® Windows® Operating SystemMicrosoft CorporationRemote Desktop ConnectionRemote Desktop Connection0.02016-05-25 18:03:34.3032016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_13mstsc.exeC:\Windows\system32\mstsc.exeMicrosoft® Windows® Operating SystemMicrosoft CorporationRemote Desktop ConnectionRemote Desktop Connection0.02016-05-25 18:03:05.5372016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_10mstsc.exeC:\Windows\system32\mstsc.exeMicrosoft® Windows® Operating SystemMicrosoft CorporationRemote Desktop ConnectionRemote Desktop Connection0.02016-05-25 18:02:47.3962016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_7mstsc.exeC:\Windows\system32\mstsc.exeMicrosoft® Windows® Operating SystemMicrosoft CorporationRemote Desktop ConnectionRemote Desktop Connection0.02016-05-25 18:02:19.4262016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_4mstsc.exeC:\Windows\system32\mstsc.exeMicrosoft® Windows® Operating SystemMicrosoft CorporationRemote Desktop ConnectionRemote Desktop Connection0.02016-05-25 18:01:53.8012016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_1Explorer.EXEC:\Windows\Explorer.EXEMicrosoft® Windows® Operating SystemMicrosoft CorporationWindows ExplorerStart menu0.02016-05-25 18:01:33.412016-05-25 18:00:002016-05-25  61006D00690074007000610077004300540032002D004100500050002D0039003000390031003000300031004100410030003200370032003900360042003100370032002E00320037002E0036002E00310035003500430059004200410047004500f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_0StartWindowsSwitchStartWindowsSwitchStartWindowsSwitchStartWindowsSwitchStartWindowsSwitchStartWindowsSwitch0.02016-05-25 18:01:33.412016-05-25 18:00:002016-05-25 |

Additionally we can define Row Format and Field separator to load the data in directory

|  |
| --- |
| hive> INSERT OVERWRITE LOCAL DIRECTORY '/root/tempdir'  > ROW FORMAT DELIMITED  > FIELDS TERMINATED BY ','  > SELECT \* FROM temptableapplication LIMIT 10;  Query ID = root\_20170222091414\_bdef1157-15c4-4943-a9be-fa5e0ce5999e  Total jobs = 1  Launching Job 1 out of 1  Number of reduce tasks determined at compile time: 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\_1484894548299\_0049, Tracking URL = http://mac55:8088/proxy/application\_1484894548299\_0049/  Kill Command = /opt/cloudera/parcels/CDH-5.8.2-1.cdh5.8.2.p0.3/lib/hadoop/bin/hadoop job -kill job\_1484894548299\_0049  Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 1  2017-02-22 09:14:55,736 Stage-1 map = 0%, reduce = 0%  2017-02-22 09:15:10,525 Stage-1 map = 33%, reduce = 0%, Cumulative CPU 28.08 sec  2017-02-22 09:15:12,577 Stage-1 map = 67%, reduce = 0%, Cumulative CPU 32.54 sec  2017-02-22 09:15:31,008 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 42.8 sec  2017-02-22 09:15:36,135 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 43.71 sec  MapReduce Total cumulative CPU time: 43 seconds 710 msec  Ended Job = job\_1484894548299\_0049  Copying data to local directory /root/tempdir  MapReduce Jobs Launched:  Stage-Stage-1: Map: 3 Reduce: 1 Cumulative CPU: 43.71 sec HDFS Read: 132172238 HDFS Write: 4500 SUCCESS  Total MapReduce CPU Time Spent: 43 seconds 710 msec  OK  Time taken: 59.292 seconds |

Check data

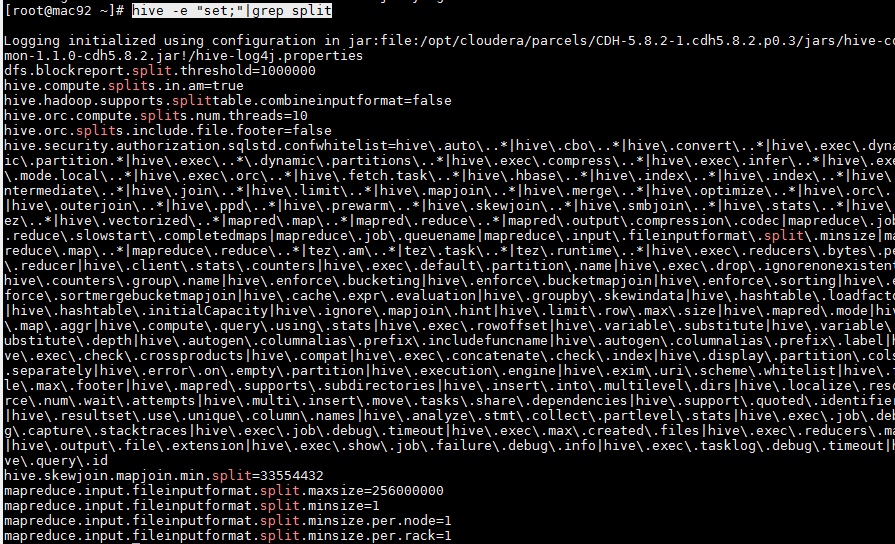
|  |
| --- |
| [root@mac55 tempdir]# cat 000000\_0  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_5,mstsc.exe,C:\Windows\system32\mstsc.exe,Microsoft® Windows® Operating System,Microsoft Corporation,Remote Desktop Connection,Remote Desktop Connection,0.0,2016-05-25 18:01:53.848,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_2,Explorer.EXE,C:\Windows\Explorer.EXE,Microsoft® Windows® Operating System,Microsoft Corporation,Windows Explorer,Search Pane,0.0,2016-05-25 18:01:43.613,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_19,mstsc.exe,C:\Windows\system32\mstsc.exe,Microsoft® Windows® Operating System,Microsoft Corporation,Remote Desktop Connection,Remote Desktop Connection,0.0,2016-05-25 18:05:39.915,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_16,mstsc.exe,C:\Windows\system32\mstsc.exe,Microsoft® Windows® Operating System,Microsoft Corporation,Remote Desktop Connection,Remote Desktop Connection,0.0,2016-05-25 18:03:34.303,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_13,mstsc.exe,C:\Windows\system32\mstsc.exe,Microsoft® Windows® Operating System,Microsoft Corporation,Remote Desktop Connection,Remote Desktop Connection,0.0,2016-05-25 18:03:05.537,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_10,mstsc.exe,C:\Windows\system32\mstsc.exe,Microsoft® Windows® Operating System,Microsoft Corporation,Remote Desktop Connection,Remote Desktop Connection,0.0,2016-05-25 18:02:47.396,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_7,mstsc.exe,C:\Windows\system32\mstsc.exe,Microsoft® Windows® Operating System,Microsoft Corporation,Remote Desktop Connection,Remote Desktop Connection,0.0,2016-05-25 18:02:19.426,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_4,mstsc.exe,C:\Windows\system32\mstsc.exe,Microsoft® Windows® Operating System,Microsoft Corporation,Remote Desktop Connection,Remote Desktop Connection,0.0,2016-05-25 18:01:53.801,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_1,Explorer.EXE,C:\Windows\Explorer.EXE,Microsoft® Windows® Operating System,Microsoft Corporation,Windows Explorer,Start menu,0.0,2016-05-25 18:01:33.41,2016-05-25 18:00:00,2016-05-25  61006D0069007400700061007700,4300540032002D004100500050002D003900300039003100,300030003100410041003000320037003200390036004200,3100370032002E00320037002E0036002E00310035003500,430059004200410047004500,f0b9c075-72aa-46b3-af68-9253df8d66b1\_Client\_0,StartWindowsSwitch,StartWindowsSwitch,StartWindowsSwitch,StartWindowsSwitch,StartWindowsSwitch,StartWindowsSwitch,0.0,2016-05-25 18:01:33.41,2016-05-25 18:00:00,2016-05-25 |

Now data is comma separated

# Check any configuration parameter in hive

From linux file system run below command to check the configuration parameter which contain split

|  |
| --- |
| hive -e "set;"|grep split |



# Connect hive with beeline

Just type beeline on terminal and click enter

#beeline>!connect jdbc:hive2://localhost:10000 <username> <password>

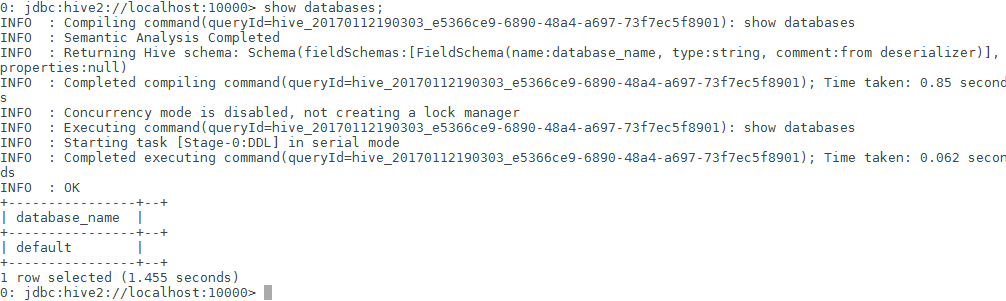
Or we can specify hive metastore’s username and password later

|  |
| --- |
| beeline> !connect jdbc:hive2://localhost:10000  Connecting to jdbc:hive2://localhost:10000  Enter username for jdbc:hive2://localhost:10000: db\_hive  Enter password for jdbc:hive2://localhost:10000: \*\*\*\*\*\*\*\*  Connected to: Apache Hive (version 1.1.0-cdh5.9.0)  Driver: Hive JDBC (version 1.1.0-cdh5.9.0)  Transaction isolation: TRANSACTION\_REPEATABLE\_READ  0: jdbc:hive2://localhost:10000> |

Or to connect without user name and password use below command

|  |
| --- |
| beeline -u jdbc:hive2://mac127:10000 |

Here –u is database URL



!q to quit beeline

# Permission denied at the time of creating External table

|  |
| --- |
| hive (esplus)> CREATE EXTERNAL TABLE ApplicationSwitchData  > (  > json string  > )  > LOCATION '/esplus/tables/applicationswitchdata';  FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask. MetaException(message:java.security.AccessControlException: Permission denied: user=root, access=WRITE, inode="/esplus/tables/applicationswitchdata":hdfs:hdfs:drwxr-xr-x  at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:319)  at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:219)  at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:190)  at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermission(FSDirectory.java:1771)  at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermission(FSDirectory.java:1755)  at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPathAccess(FSDirectory.java:1729)  at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkAccess(FSNamesystem.java:8348)  at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.checkAccess(NameNodeRpcServer.java:1978)  at org.apache.hadoop.hdfs.protocolPB.ClientNamenodeProtocolServerSideTranslatorPB.checkAccess(ClientNamenodeProtocolServerSideTranslatorPB.java:1443)  at org.apache.hadoop.hdfs.protocol.proto.ClientNamenodeProtocolProtos$ClientNamenodeProtocol$2.callBlockingMethod(ClientNamenodeProtocolProtos.java)  at org.apache.hadoop.ipc.ProtobufRpcEngine$Server$ProtoBufRpcInvoker.call(ProtobufRpcEngine.java:616)  at org.apache.hadoop.ipc.RPC$Server.call(RPC.java:969)  at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2151)  at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2147)  at java.security.AccessController.doPrivileged(Native Method)  at javax.security.auth.Subject.doAs(Subject.java:422)  at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1657)  at org.apache.hadoop.ipc.Server$Handler.run(Server.java:2145) |

Use below command to log into hive shell and create table again

|  |
| --- |
| sudo -u hdfs hive |

# File permission issue (No files matching path)

|  |
| --- |
| LOAD DATA LOCAL INPATH '/root/Esplus/newdatasets/13\_07\_2016/4\_1-7-2016\_1-44-1\_06096.txt' INTO TABLE esplus.ApplicationSwitchData;  FAILED: SemanticException Line 1:23 Invalid path ''/root/Esplus/newdatasets/13\_07\_2016/4\_1-7-2016\_1-44-1\_06096.txt'': No files matching path file:/root/Esplus/newdatasets/13\_07\_2016/4\_1-7-2016\_1-44-1\_06096.txt |

The file does exist at the specified location in the Linux file system and all users have read permission.

If its in the root folder only root can access it. Hive error messages can be pretty generic here and not distinguish between access rights and files exists. If you run beeline the read will be executed by the hive server and that one is running under the hive user so only he could access the data.

First give read write permission to folder

|  |
| --- |
| chmod -R 755 newdatasets/ |

Move Esplus folder to root (/)

|  |
| --- |
| mv Esplus / |

Change owner from root to hdfs

chown -R hdfs:hdfs Esplus

|  |
| --- |
| -rw-r--r-- 1 root root 12785 Dec 12 02:19 emp.java  drwxr-xr-x 5 hdfs hdfs 109 Jan 3 00:52 Esplus |

# Hdfs command in hive

|  |
| --- |
| Hive>dfs -ls |

# Unix command in hive

|  |
| --- |
| Hive> **!ls** |

# json serde in hive

<http://thornydev.blogspot.in/2013/07/querying-json-records-via-hive.html>

<https://github.com/rcongiu/Hive-JSON-Serde>

# Move data from json serde table to normal table

Json serde Table use

|  |
| --- |
| CREATE EXTERNAL TABLE JsonTable\_raw (  data array<struct<  start\_date:string,  end\_date:string,  measures:struct<  Visitors:int,  Singlepagevisits:int  ) ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe' |

Non serde table

|  |
| --- |
| CREATE EXTERNAL TABLE JsonTable (  start\_date string,  end\_date string,  Visitors int,  Singlepagevisits int  ) |

Insert query

|  |
| --- |
| INSERT INTO TABLE JsonTable  select cols.start\_date,cols.end\_date,cols.measures.Visitors,  cols.measures.Singlepagevisits  FROM JsonTable\_raw jt lateral view explode(jt.data) collection as cols; |

# Hive - External Table with Partitions

When the data files become huge (number and size) then we might need to use Partition to improve the efficiency of data processing.

|  |
| --- |
| CREATE TABLE user (  userId BIGINT,  type INT,  level TINYINT,  )  COMMENT 'User Infomation'  PARTITIONED BY (date String) |

date String is moved to PARTITIONED BY, when we need to load data into hive, partition must be assigned.

|  |
| --- |
| LOAD INPATH '/user/chris/data/testdata' OVERWRITE INTO TABLE user PARTITION (date='2012-02-22') |

After data is loaded, we can see a new folder named date=2010-02-22 is created inside **/user/chris/warehouse/user/**

So, how can we do it using external table?

Same as before, first declare the external table user, and assign the location.

|  |
| --- |
| CREATE EXTERNAL TABLE user (  userId BIGINT,  type INT,  level TINYINT,  date String  )  COMMENT 'User Infomation'  PARTITIONED BY (date String)  LOCATION '/user/chris/datastore/user/'; |

Then, create the folder **date=2010-02-22** inside **/user/chris/datastore/user/**

At last, put the data files of date 2010-02-22 into the folder, done.

But,

When we executes select \* from user; nothing appears.

Because when external table is declared, default table path is changed to specified location in hive metadata which contains in metastore, but about partition, nothing is changed, so, we must manually add those metadata.

We can check the existing partition in metadata by running below command

|  |
| --- |
| **SHOW PARTITITONS user;** |

You will not get any partition information

So we need to manually add partition

|  |
| --- |
| **ALTER TABLE user ADD PARTITION(date='2010-02-22');** |

Every time a new data=... folder (partition) is created, we must manually alter the table to add partition information.

Or we could set some different folder while adding partition

|  |
| --- |
| ALTER TABLE user ADD PARTITION(date='2010-02-22') LOCATION '/external/test’; |

If new partition data's were added to HDFS (without alter table add partition command execution) then we can sync up the metadata by executing the command 'msck repair'.

Or we can use MSCK (Meta Store Check Command) to add partition in metadata

|  |
| --- |
| MSCK REPAIR TABLE <table\_name> |

Note: Msck does not remove a partition whom directory does not exist in the filesystem. It only notifies which partitions' directories does not exist in the filesystem. You should manually run "alter table drop partition”.

MSCK Repair command only creates partitions when folder is specified in proper format like **year=2017/month=01/day-01**, if you don’t have folder with proper name you need to use ALTER table command

|  |
| --- |
| ALTER TABLE user ADD PARTITION (year='2017',month='01',day='01') location external/test/2017/01/01/' |

# Replace Blank Value with Null

|  |
| --- |
| CREATE EXTERNAL TABLE person  (  name string,  title string,  birth\_year string  )  ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'  LOCATION '/person'  **TBLPROPERTIES('serialization.null.format'='');** |

Set table properties like mentioned above, this will replace all blank values into null while inserting data into table

For more information refer below link

<https://abhijitsureshshingate.wordpress.com/2013/11/16/how-to-use-blank-as-null-in-hive/>

TBLPROPERTIES('serialization.null.format'='') means the following:

* An empty field in the data files will be treated as NULL when you query the table
* When inserting rows to the table, NULL values will be written to the data files as empty fields

You are doing something else -  
You are inserting an empty string to a table from a query.  
It is treated "as is" - an empty string.

**Demo**

**bash**

hdfs dfs -mkdir /user/hive/warehouse/mytable

echo Hello,,World | hdfs dfs -put - /user/hive/warehouse/mytable/data.txt

**hive**

create table mytable (s1 string,s2 string,s3 string)

row format delimited

fields terminated by ','

;

hive> select \* from mytable;

OK

s1 s2 s3

Hello World

hive> alter table mytable set tblproperties ('serialization.null.format'='');

OK

hive> select \* from mytable;

OK

s1 s2 s3

Hello NULL World

# Map phase stuck at 0%

I was trying to launch one query in which we have a join of two tables , hive converted this join to map side join and stuck at 0% and it was launching tasks again and again , we tried to increase container memory but issue persist

Please find below log for the same

|  |
| --- |
| Total jobs = 1  Execution log at: /tmp/root/root\_20170309060707\_2fb18bdd-b054-42b7-9075-c718ff55da4f.log  2017-03-09 06:07:49 Starting to launch local task to process map join; maximum memory = 1908932608  2017-03-09 06:07:55 Dump the side-table for tag: 1 with group count: 15 into file: file:/tmp/root/858bcac1-d9dd-484d-b51f-23c909351983/hive\_2017-03-09\_06-07-36\_752\_4756823141955389275-1/-local-10002/HashTable-Stage-4/MapJoin-mapfile01--.hashtable  2017-03-09 06:07:55 Uploaded 1 File to: file:/tmp/root/858bcac1-d9dd-484d-b51f-23c909351983/hive\_2017-03-09\_06-07-36\_752\_4756823141955389275-1/-local-10002/HashTable-Stage-4/MapJoin-mapfile01--.hashtable (760085 bytes)  2017-03-09 06:07:55 End of local task; Time Taken: 5.587 sec.  Execution completed successfully  MapredLocal task succeeded  Launching Job 1 out of 1  Number of reduce tasks is set to 0 since there's no reduce operator  Starting Job = job\_1488780768629\_0041, Tracking URL = http://c2esplusm1.espluscs1.i6.internal.cloudapp.net:8088/proxy/application\_1488780768629\_0041/  Kill Command = /opt/cloudera/parcels/CDH-5.7.0-1.cdh5.7.0.p0.45/lib/hadoop/bin/hadoop job -kill job\_1488780768629\_0041  Hadoop job information for Stage-4: number of mappers: 1; number of reducers: 0  2017-03-09 06:08:18,025 Stage-4 map = 0%, reduce = 0%  2017-03-09 06:09:18,968 Stage-4 map = 0%, reduce = 0%, Cumulative CPU 34.67 sec |

So to resolve this issue we change below property

set hive.auto.convert.join=false;

By doing so it is not converted into map side join and ran successfully

# Count number of rows in a particular bucket

Assume we have following bucketed table

|  |
| --- |
| CREATE TABLE timeslot\_bucketed(  username string,  modulename string,  timeslot smallint,  date string)  CLUSTERED BY (timeslot) INTO 24 BUCKETS |

If we want to count number of rows in bucket 1 , we can fire below query

|  |
| --- |
| SELECT count(\*) FROM timeslot\_bucketed TABLESAMPLE(BUCKET 1 OUT OF 24 ON timeslot); |

# Sampling Bucketized Table

|  |
| --- |
| table\_sample: TABLESAMPLE (BUCKET x OUT OF y [ON colname]) |

The TABLESAMPLE clause allows the users to write queries for samples of the data instead of the whole table. The TABLESAMPLE clause can be added to any table in the FROM clause. The buckets are numbered starting from 1. colname indicates the column on which to sample each row in the table. colname can be one of the non-partition columns in the table or rand() indicating sampling on the entire row instead of an individual column. The rows of the table are 'bucketed' on the colname randomly into y buckets numbered 1 through y. Rows which belong to bucket x are returned.

In the following example the 3rd bucket out of the 32 buckets of the table source would be selected. 's' is the table alias.

|  |
| --- |
| SELECT \*FROM source TABLESAMPLE(BUCKET 3 OUT OF 32 ON rand()) s; |

**Input pruning:** Typically, TABLESAMPLE will scan the entire table and fetch the sample. But, that is not very efficient. Instead, the table can be created with a CLUSTERED BY clause which indicates the set of columns on which the table is hash-partitioned/clustered on. If the columns specified in the TABLESAMPLE clause match the columns in the CLUSTERED BY clause, TABLESAMPLE scans only the required hash-partitions of the table.

**Example:**

So in the above example, if table 'source' was created with 'CLUSTERED BY id INTO 32 BUCKETS'

|  |
| --- |
| TABLESAMPLE(BUCKET 3 OUT OF 16 ON id) |

Would pick out the 3rd and 19th clusters as each bucket would be composed of (32/16)=2 clusters.

* From first cluster 1 to 16 buckets will be 3rd bucket
* From second cluster 16 to 32 buckets will be 19th bucket

On the other hand the tablesample clause

|  |
| --- |
| TABLESAMPLE(BUCKET 3 OUT OF 64 ON id) |

Would pick out half of the 3rd cluster as each bucket would be composed of (32/64)=1/2 of a cluster.

# Insert multiple rows from single statement

|  |
| --- |
| INSERT INTO TABLE <Table\_name> values (<val11>,(<val12>), (<val21>,(<val22>); |

|  |
| --- |
| insert into table books values ("1","book1"), ("2","book2"); |

# Queried on data as table

|  |
| --- |
| with tbl\_test\_values as  (  select inline  (  array  (  struct ('Number','1','One')  ,struct ('Number','5','Five')  ,struct ('Letter','A','First Letter')  ,struct ('Human','Bob','Dude')  )  ) as (Name, Value, Descript)  )  select \* from tbl\_test\_values; |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **tbl\_test\_values.name** | **tbl\_test\_values.value** | **tbl\_test\_values.descript** |
| 1 | Number | 1 | One |
| 2 | Number | 5 | Five |
| 3 | Letter | A | First Letter |
| 4 | Human | Bob | Dude |

# How to put file in hdfs using echo statement

|  |
| --- |
| hdfs dfs -mkdir /user/hive/warehouse/mytable  echo Hello,,World | hdfs dfs -put - /user/hive/warehouse/mytable/data.txt |

# Group by in hive

|  |
| --- |
| Option 1 select year(from\_unixtime(unix\_timestamp(date,'dd-MM-yyyy'))) as year  ,max(value) as max\_value  from t  group by year(from\_unixtime(unix\_timestamp(date,'dd-MM-yyyy')))  ; Option 2 **pre Hive 2.2.0**  set hive.groupby.orderby.position.alias=true;  **as of Hive 2.2.0**  set hive.groupby.position.alias=true;  select year(from\_unixtime(unix\_timestamp(date,'dd-MM-yyyy'))) as date  ,max(value)  from t  group by 1  ;  +------+-----------+  | year | max\_value |  +------+-----------+  | 1900 | 23 |  | 1901 | 24 |  +------+-----------+ |

# Merge small RC/ORC files

**Alter Table/Partition Concatenate**

|  |
| --- |
| ALTER TABLE table\_name [PARTITION (partition\_key = 'partition\_value' [, ...])] CONCATENATE; |

If the table or partition contains many small RCFiles or ORC files, then the above command will merge them into larger files. In case of RCFile the merge happens at block level whereas for ORC files the merge happens at stripe level thereby avoiding the overhead of decompressing and decoding the data.

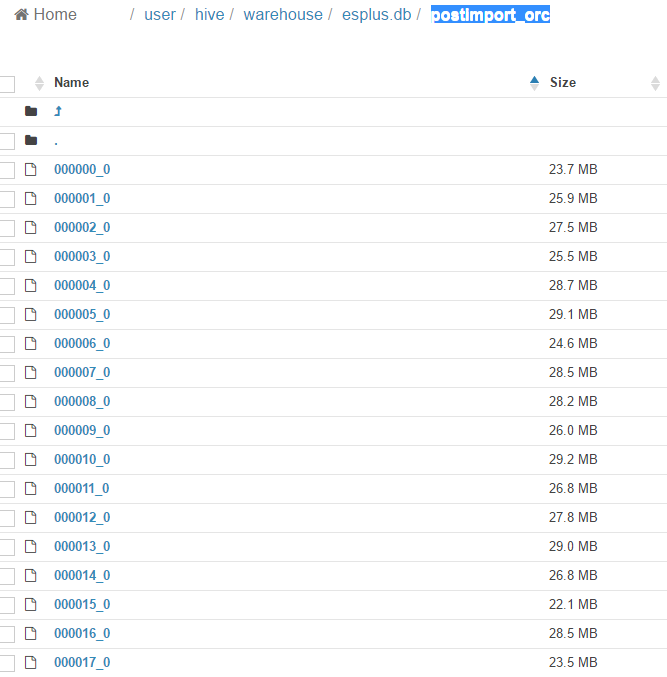


|  |
| --- |
| hive> ALTER TABLE PostImport\_Part PARTITION(date='2016-07-04') CONCATENATE;  Starting Job = job\_1490094830773\_0436, Tracking URL = http://mac55:8088/proxy/application\_1490094830773\_0436/  Kill Command = /opt/cloudera/parcels/CDH-5.8.2-1.cdh5.8.2.p0.3/lib/hadoop/bin/hadoop job -kill job\_1490094830773\_0436  Hadoop job information for null: number of mappers: 1; number of reducers: 0  2017-04-12 06:37:44,991 null map = 0%, reduce = 0%  2017-04-12 06:37:59,905 null map = 57%, reduce = 0%, Cumulative CPU 1.8 sec  2017-04-12 06:38:05,227 null map = 100%, reduce = 0%, Cumulative CPU 2.33 sec  MapReduce Total cumulative CPU time: 2 seconds 330 msec  Ended Job = job\_1490094830773\_0436  Loading data to table esplus.postimport\_part partition (date=2016-07-04)  Partition esplus.postimport\_part{date=2016-07-04} stats: [numFiles=1, numRows=0, totalSize=158624278, rawDataSize=0]  MapReduce Jobs Launched:  Stage-null: Map: 1 Cumulative CPU: 2.33 sec HDFS Read: 158768280 HDFS Write: 158624278 SUCCESS  Total MapReduce CPU Time Spent: 2 seconds 330 msec  OK  Time taken: 29.584 seconds  hive> |



Here we can see that two orc files are merged in one file,

We need to define partition key and values



|  |
| --- |
| hive> ALTER TABLE postimport\_orc CONCATENATE;  Starting Job = job\_1490094830773\_0446, Tracking URL = http://mac55:8088/proxy/application\_1490094830773\_0446/  Kill Command = /opt/cloudera/parcels/CDH-5.8.2-1.cdh5.8.2.p0.3/lib/hadoop/bin/hadoop job -kill job\_1490094830773\_0446  Hadoop job information for null: number of mappers: 8; number of reducers: 0  2017-04-12 07:14:39,164 null map = 0%, reduce = 0%  2017-04-12 07:14:48,952 null map = 13%, reduce = 0%, Cumulative CPU 2.3 sec  2017-04-12 07:14:50,089 null map = 24%, reduce = 0%, Cumulative CPU 11.07 sec  2017-04-12 07:14:52,226 null map = 30%, reduce = 0%, Cumulative CPU 14.31 sec  2017-04-12 07:14:55,470 null map = 31%, reduce = 0%, Cumulative CPU 14.77 sec  2017-04-12 07:14:57,759 null map = 35%, reduce = 0%, Cumulative CPU 15.35 sec  2017-04-12 07:14:58,855 null map = 40%, reduce = 0%, Cumulative CPU 16.24 sec  2017-04-12 07:15:01,143 null map = 46%, reduce = 0%, Cumulative CPU 17.08 sec  2017-04-12 07:15:03,410 null map = 50%, reduce = 0%, Cumulative CPU 17.41 sec  2017-04-12 07:15:04,544 null map = 55%, reduce = 0%, Cumulative CPU 19.12 sec  2017-04-12 07:15:07,057 null map = 58%, reduce = 0%, Cumulative CPU 19.46 sec  2017-04-12 07:15:08,212 null map = 59%, reduce = 0%, Cumulative CPU 19.57 sec  2017-04-12 07:15:09,590 null map = 63%, reduce = 0%, Cumulative CPU 20.1 sec  2017-04-12 07:15:10,952 null map = 64%, reduce = 0%, Cumulative CPU 20.55 sec  2017-04-12 07:15:13,174 null map = 66%, reduce = 0%, Cumulative CPU 21.12 sec  2017-04-12 07:15:16,570 null map = 70%, reduce = 0%, Cumulative CPU 21.76 sec  2017-04-12 07:15:18,823 null map = 72%, reduce = 0%, Cumulative CPU 21.99 sec  2017-04-12 07:15:22,223 null map = 73%, reduce = 0%, Cumulative CPU 22.48 sec  2017-04-12 07:15:23,361 null map = 76%, reduce = 0%, Cumulative CPU 22.78 sec  2017-04-12 07:15:25,636 null map = 80%, reduce = 0%, Cumulative CPU 23.27 sec  2017-04-12 07:15:27,906 null map = 81%, reduce = 0%, Cumulative CPU 23.63 sec  2017-04-12 07:15:32,435 null map = 82%, reduce = 0%, Cumulative CPU 23.9 sec  2017-04-12 07:15:33,571 null map = 83%, reduce = 0%, Cumulative CPU 24.11 sec  2017-04-12 07:15:34,702 null map = 85%, reduce = 0%, Cumulative CPU 24.32 sec  2017-04-12 07:15:35,832 null map = 86%, reduce = 0%, Cumulative CPU 24.55 sec  2017-04-12 07:15:38,101 null map = 87%, reduce = 0%, Cumulative CPU 24.77 sec  2017-04-12 07:15:40,333 null map = 91%, reduce = 0%, Cumulative CPU 25.19 sec  2017-04-12 07:15:41,395 null map = 92%, reduce = 0%, Cumulative CPU 25.36 sec  2017-04-12 07:15:43,505 null map = 94%, reduce = 0%, Cumulative CPU 25.72 sec  2017-04-12 07:15:48,792 null map = 96%, reduce = 0%, Cumulative CPU 25.92 sec  2017-04-12 07:15:49,854 null map = 97%, reduce = 0%, Cumulative CPU 26.18 sec  2017-04-12 07:15:52,971 null map = 100%, reduce = 0%, Cumulative CPU 26.6 sec  MapReduce Total cumulative CPU time: 26 seconds 600 msec  Ended Job = job\_1490094830773\_0446  Loading data to table esplus.postimport\_orc  Table esplus.postimport\_orc stats: [numFiles=8, numRows=0, totalSize=2113581609, rawDataSize=0]  MapReduce Jobs Launched:  Stage-null: Map: 8 Cumulative CPU: 26.6 sec HDFS Read: 2005534482 HDFS Write: 2113581609 SUCCESS  Total MapReduce CPU Time Spent: 26 seconds 600 msec  OK  Time taken: 81.744 seconds  hive> |







# Convert Json using group by sum

Json file

|  |
| --- |
| {"key" : "ke" , "value" : 1 }  {"key" : "ke" , "value" : 2 }  {"key" : "ke1" , "value" : 5 } |

|  |
| --- |
| create table mytable(str string); |

|  |
| --- |
| load data local inpath '/root/json\_string.txt' into table mytable; |

|  |
| --- |
| hive> select concat('{"key":"',jt.key,'","value":',sum(jt.value),'}')  > from mytable t  > lateral view json\_tuple(t.str, 'key', 'value') jt as key,value  > group by jt.key;  Query ID = root\_20170413175151\_8199ad35-07dc-4836-9170-fa6275fae4c6  Total jobs = 1  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\_1491484583384\_0036, Tracking URL = http://mac127:8088/proxy/application\_1491484583384\_0036/  Kill Command = /opt/cloudera/parcels/CDH-5.9.0-1.cdh5.9.0.p0.23/lib/hadoop/bin/hadoop job -kill job\_1491484583384\_0036  Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1  2017-04-13 17:51:44,190 Stage-1 map = 0%, reduce = 0%  2017-04-13 17:51:52,469 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.96 sec  2017-04-13 17:52:01,788 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 4.61 sec  MapReduce Total cumulative CPU time: 4 seconds 610 msec  Ended Job = job\_1491484583384\_0036  MapReduce Jobs Launched:  Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 4.61 sec HDFS Read: 9241 HDFS Write: 51 SUCCESS  Total MapReduce CPU Time Spent: 4 seconds 610 msec  OK  {"key":"ke","value":3.0}  {"key":"ke1","value":5.0}  Time taken: 28.065 seconds, Fetched: 2 row(s) |

# Count number of words in column separated by “|”

Input data is

+----------------------+--------------------------------+

| movie\_name | Genres |

+----------------------+--------------------------------+

| digimon | Adventure|Animation|Children's |

| Slumber\_Party\_Massac | Horror |

+----------------------+--------------------------------+

select \*

,size(split(coalesce(Genres,''),'[^|\\s]+'))-1 as count\_of\_genres

from mytable

Output

+----------------------+--------------------------------+-----------------+

| movie\_name | Genres | count\_of\_genres |

+----------------------+--------------------------------+-----------------+

| digimon | Adventure|Animation|Children's | 3 |

| Slumber\_Party\_Massac | Horror | 1 |

+----------------------+--------------------------------+-----------------+

# Print Beeline output without header

From shell you can execute query like below

beeline --showHeader=false --outputformat=tsv2 -e "select ..."

Within beeline

!set showheader false

!set outputformat tsv2

select ...;