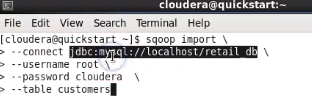
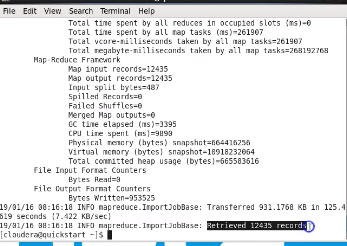


Show tables;

Describe customers; 🡪 will get all details about the customers



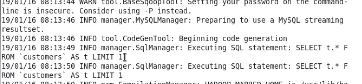


Today 12435 records are available in the table.

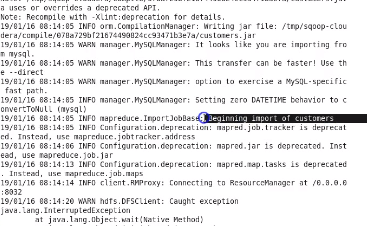
Java classes is invoked. Sqoop supports parallel processing.

Fault tolerant is provided.

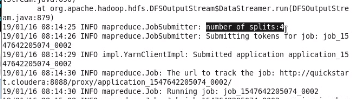
It runs in various threads . I this case 4 threads run in parallel to fetch the data.



First only one column is fetched to get the meta data to use in java class



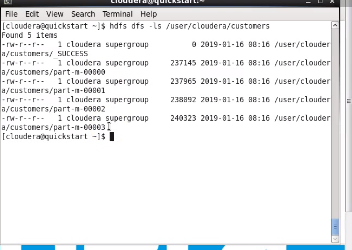
A jar file is created after reading the meta data. After that it will begin the execution of import



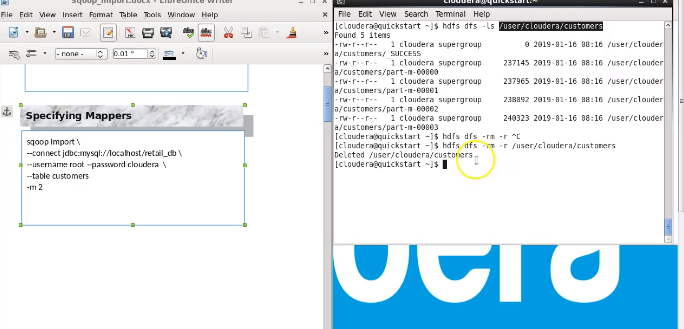
Four parallel processing are running

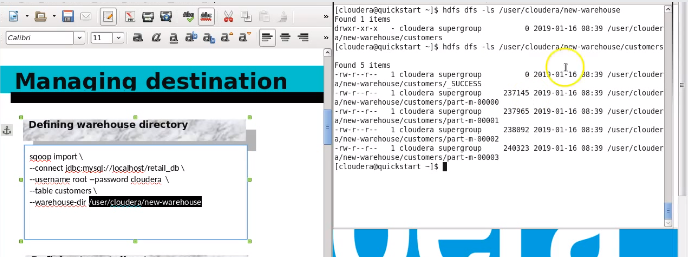
Once the sqoop command executes the data exist in the default warehouse directory

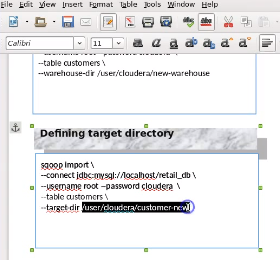
/user/cloudera/customers

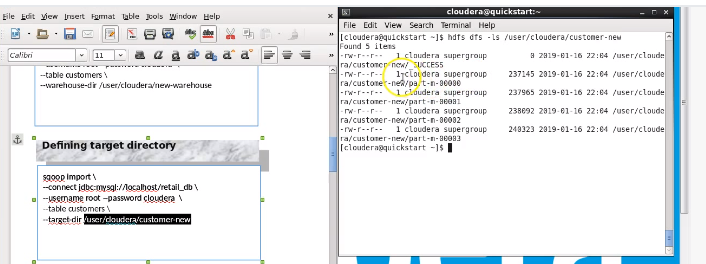


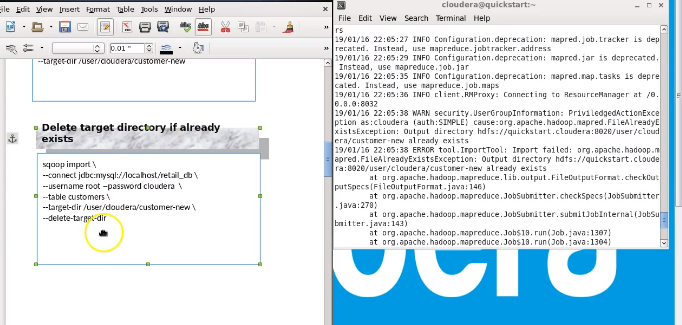
This directory will contain four mappers and so four files are available.







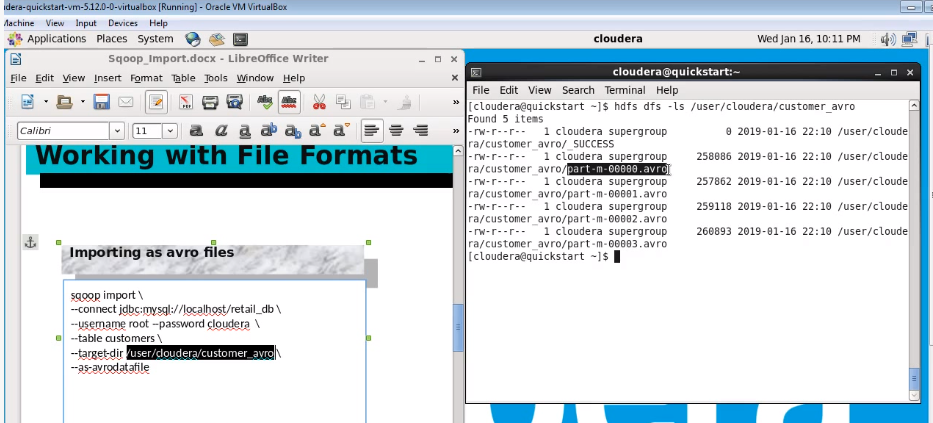




# Working with file formats:

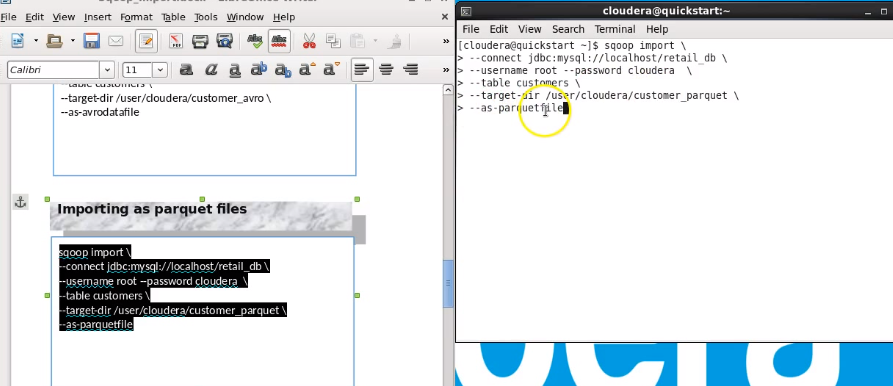
## Avro file

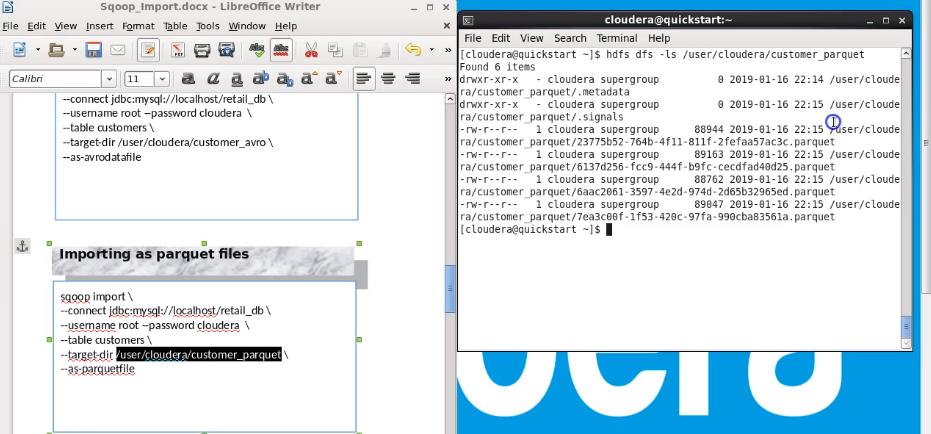
Avro file saves the schema and data together and in binary format. So it is very efficient in Hadoop environment



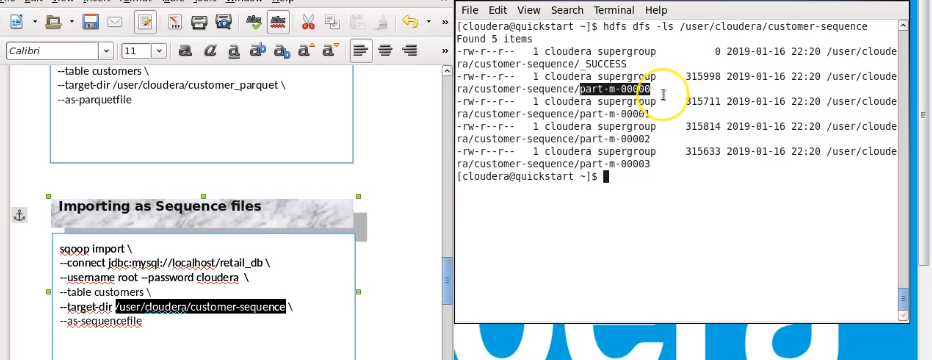
Internally saved in the json format but actually stored/compressed in binary format

## Parquet file

 Apache Parquet is a free and open-source column-oriented data storage format of the Apache Hadoop ecosystem

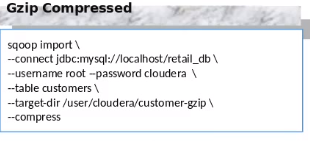


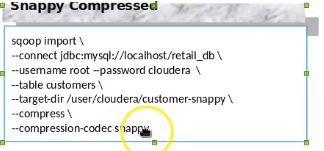
## No extension for sequence file

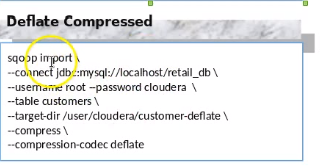


# Working with compression type

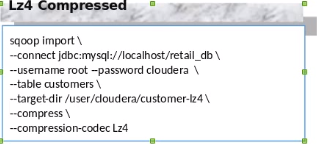
Gzip files are unsplittable and avoided in Hadoop.

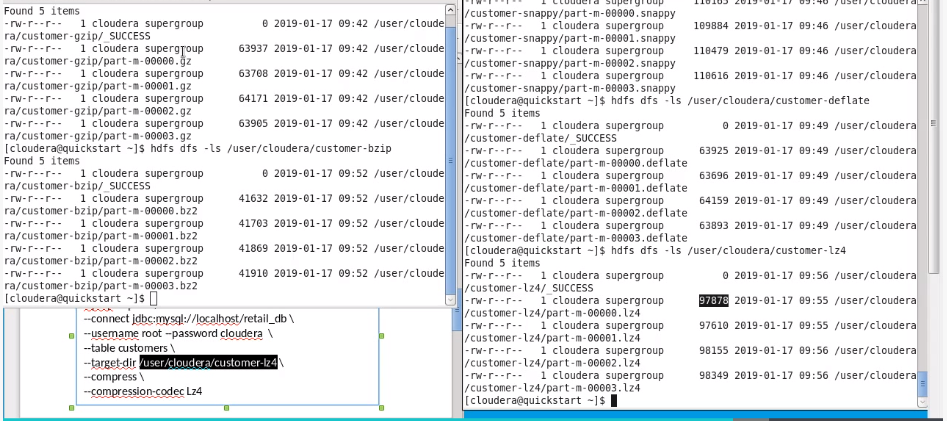






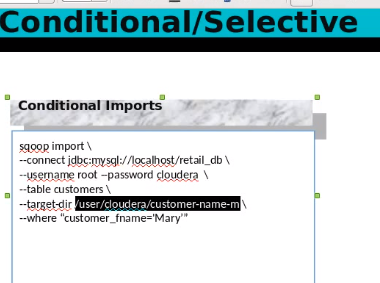


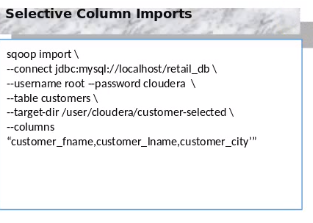




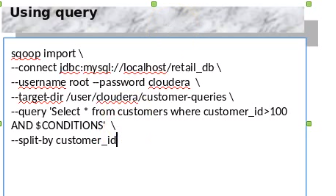
* Bz2 compression is very good. splittable
* Very slow and cpu intensive
* Snappy comp is modest and very fast . splittable.
* Lz2 provides modest and spllittable compression

# Conditional imports:



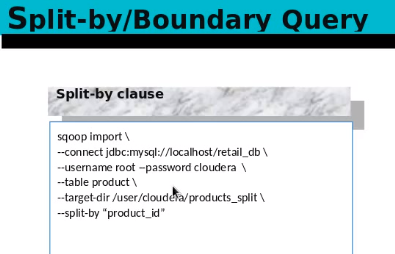


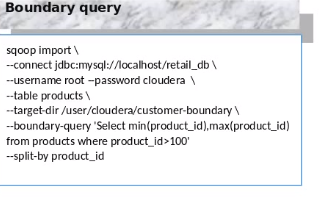
We should not mention the table name explicitly and use conditions which will be replaced by sqoop process



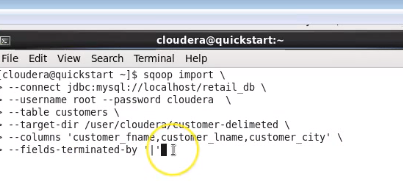
# Splityby and boundary query

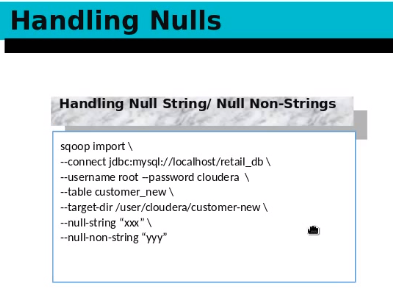
By default the split by happens based on the primary key and if we don’t specify for table not having primary key the sqoop command will fail.



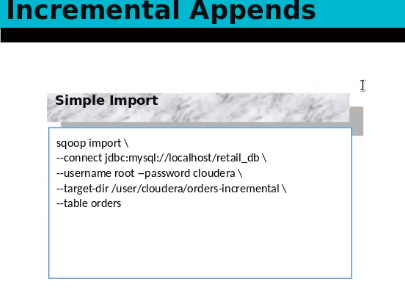


# field delimiters

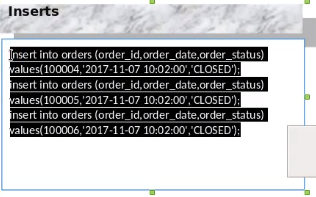


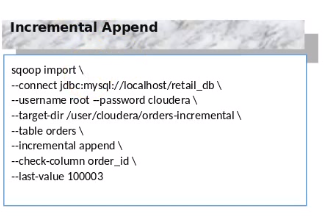


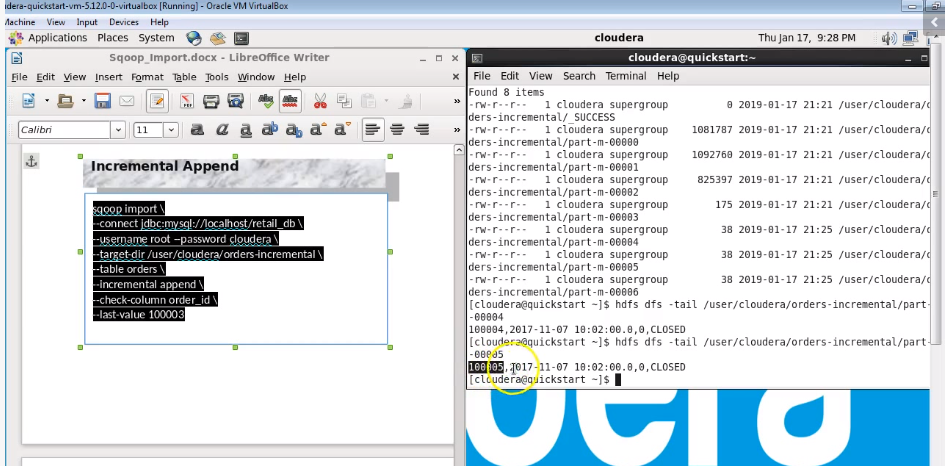
# Incremental appends



Insert the below rows

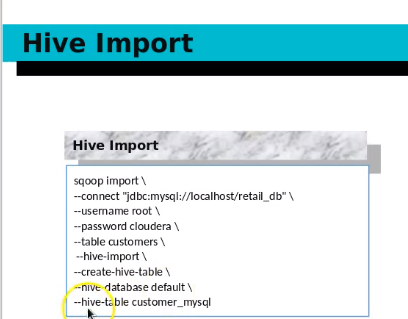






In place of append we can use the last modified if the existing data changes and the column to check for changes in the similar fashion.

# Sqoop hive import

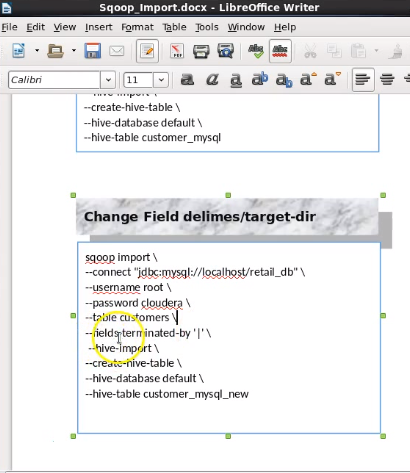


Show tables;

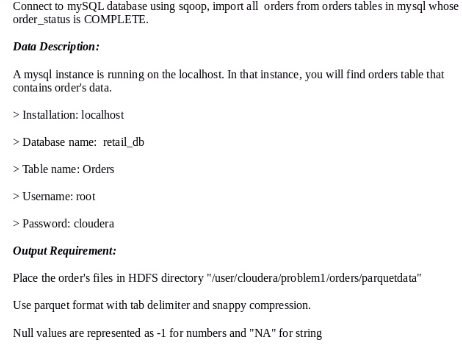
Select \* from customer\_mysql limit 5;

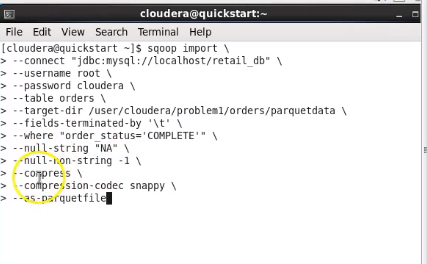
Describe formatted customer\_mysql;

Default delimiter is \n and we can implicity change as show in the below examples

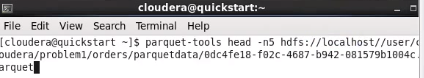


# Sqoop exercise1:

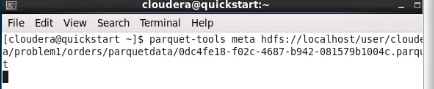




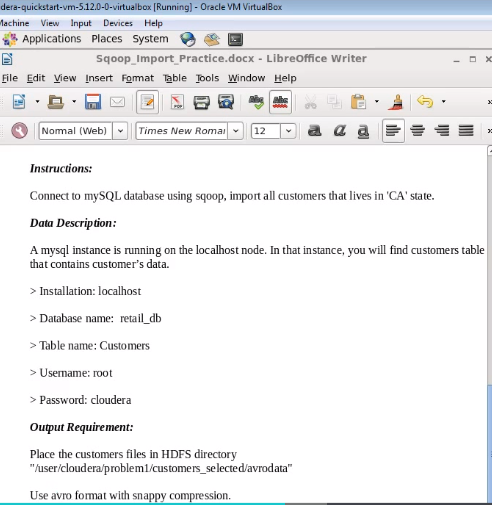
Using parquet tools to view the data

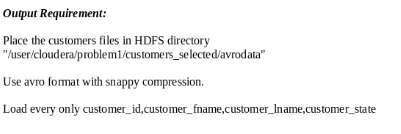


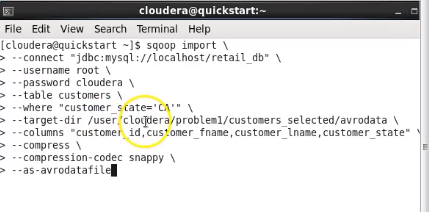
To get the meta structure



# Sqoop exercise2:





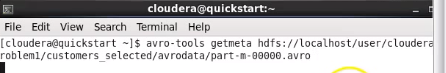


Avlo file is compressed and we can view using avro tools

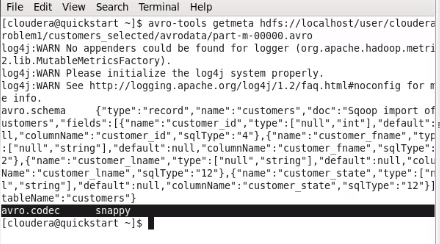




To get the meta information



Compressed in snappy format



# Sqoop exercise3

# 

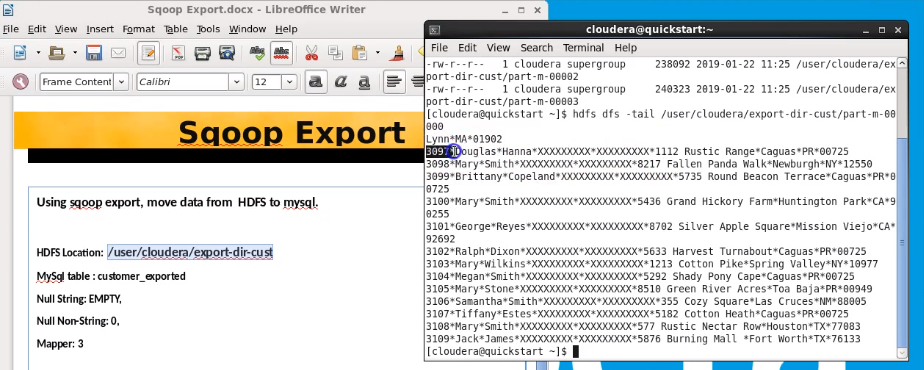
# 

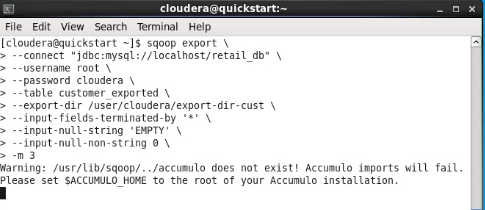
# 

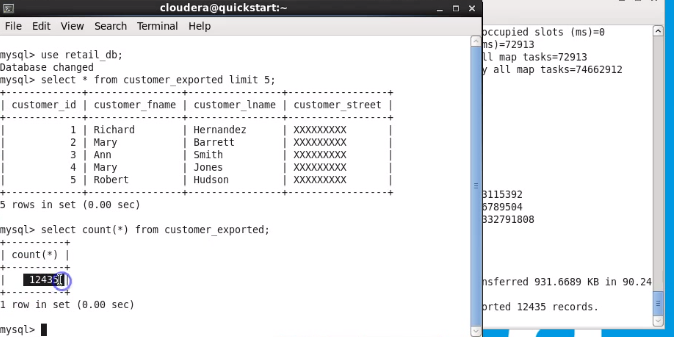
# 

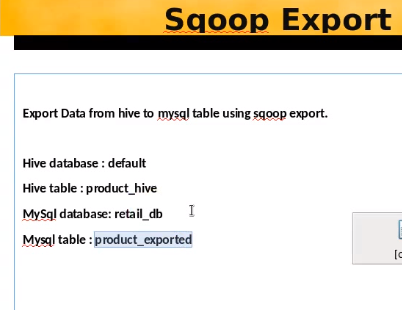
# Sqoop Export:

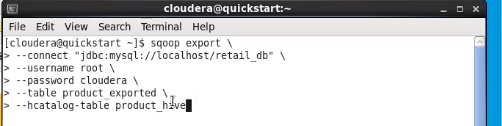


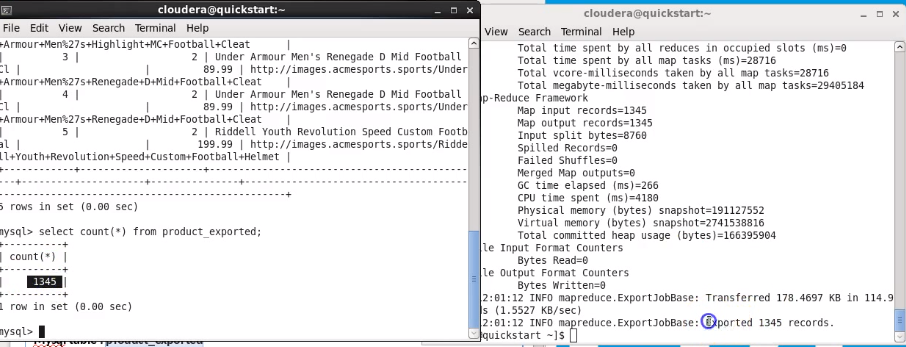












### ****I have around 300 tables in a database. I want to import all the tables from the database except the tables named Table298, Table 123, and Table299. How can I do this without having to import the tables one by one?****

This can be accomplished using the import-all-tables import command in Sqoop and by specifying the exclude-tables option with it as follows-

sqoop import-all-tables

--connect –username –password --exclude-tables Table298, Table 123, Table 299