Partitioning data in hive

1. Create table called movies_whole with 3 columns (movieid, movie_name, genre)

2. Load action_comedy_thriller file into table

3. Create a table called movies_part with 2 columns (movieid, movie_name) that is partitioned on genre

4. Load each file (action, comedy, and thriller) into a partitions ("Action", "Comedy", and "Thriller")

5. Describe the structure of the table and list the partitions (hint: describe and show partitions command)

6. Navigate to the location of movie_part on HDFS. How does the partitioned table look on HDFS? Write 1 line on what you think is happening when partitioned tables are created. **(2)**

7. Run the following queries on both **movies_part and movies_whole** table and find out the time it takes to execute the query.

-- Substitute *table* with actual table name

(a) Select * from *table* limit 20;
(b) Select count(*) from *table* where genre='Action';
(c) Select count(*) from *table*;
(d) Select t.year, count(*) as count from (Select regexp_extract(movie_name, '([1-2][0-9][0-9][0-9])',1) as year from *table*) t group by year order by count desc limit 5;
(e) Select t.year, count(*) as count from (Select regexp_extract(movie_name, '([1-2][0-9][0-9][0-9])',1) as year from *table* where genre='Thriller') t group by year order by count desc limit 5;

Answer the following two questions for each of the queries above

(7.1) On which table do you think queries should run faster?
(7.2) On which tables (movie_part or movie_whole) do they actually run faster.

8. With some help from the "select" statement in 7(e) -> create a table called movie_year_temp with following columns (movieid, movie_title, movie_year)

Bucketing data in hive

9. Create a table called year_buckets with the same column definitions as movie_year_temp, but with 8 buckets, clustered on movie_year

10. Use insert overwrite table to load the rows in movie_year_temp into year_buckets. (set "hive.enforce.bucketing" to true)

11. Navigate to the location of year_buckets on HDFS. How does the partitioned table look on HDFS?

Apply Histogram function

12. Using the table **movie_year_temp** apply the histogram function (with 5 buckets) on **movie_year** to get get the distribution of year values in the table