**Task 1**

**Data:** Dataset1

**Instructions:**

* Create database targetdb at HDFS location /data/target/targetdb

hadoop fs -mkdir /data/target

hadoop fs -mkdir /data/target/targetdb

CREATE DATABASE targetdb LOCATION '/data/target/targetdb';

use targetdb;

* Create tables: movies, ratings, tags in targetdb database

create table movies (movieId int, title string, genres string) row format delimited fields terminated by ',';

create table ratings (userId int,movieId int,rating double,t\_stamp decimal(12,0)) row format delimited fields terminated by ',';

create table tags (userId int,movieId int,tag string,t\_stamp decimal(12,0)) row format delimited fields terminated by ',';

* Load the relevant data files to those tables

LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/exercise\_6/Hive/Dataset1/movies.csv' INTO TABLE movies;

LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/exercise\_6/Hive/Dataset1/ratings.csv' INTO TABLE ratings;

LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/exercise\_6/Hive/Dataset1/tags.csv' INTO TABLE tags;

* Prepare a SQL to load movie\_stats table.

Following is the column detail for movie\_stats table.

movieid, movie\_name,avg\_rating,hash\_tag,hash\_tag\_cnt

**Note**: Choose appropriate column data type as per your knowledge.

create table movie\_stats (movieId int, movie\_name string, avg\_rating double, hash\_tag string, hash\_tag\_cnt int) row format delimited fields terminated by ',';

select m.movieid, m.title, avg(r.rating) as avg\_rating, max(t.tag) as hash\_tag, count(t.tag) as hash\_tag\_cnt

from movies m

left join ratings r

on m.movieid = r.movieid

left join tags t

on r.movieid = t.movieid and r.userid = t.userid

group by m.movieid, m.title;

=======================================================================

**Task 2 --- not possible in hive**

Let’s apply some integrity constraints to our table

* Modify movies table to make column movieId UPI and NOT NULL

ALTER TABLE movies CHANGE movieId movieId int not null

CREATE INDEX IDX\_MUPI on table movies(movieid) as 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler' with deferred rebuild ;

=======================================================================

**Task 3**

**Data:** Dataset2

**Instructions:**

* Create new database stagedb at location /data/staging/stagedb

hadoop fs -mkdir /data/staging

hadoop fs -mkdir /data/staging/stagedb

CREATE DATABASE stagedb LOCATION '/data/staging/stagedb';

use stagedb;

* Create tables movies, ratings, tags under database stagedb with same details as targetdb tables

create table movies (movieId int, title string, genres string) row format delimited fields terminated by ',';

create table ratings (userId int,movieId int,rating double,t\_stamp decimal(12,0)) row format delimited fields terminated by ',';

create table tags (userId int,movieId int,tag string,t\_stamp decimal(12,0)) row format delimited fields terminated by ',';

* Prepare three different scripts, one for each to load stage tables with given data files considering all possible integrity checks as
  + Movie ID is unique and not null field. Hence have to check for any unique key as well as not null violation
  + Rating should be for a valid movie. Hence movieId in rating data file should be present in Movie table
  + If there is any exact duplicate record then we must discard one.

=======================================================================

**Task 4**

**Instructions:**

* Prepare full refresh load script to refresh movie\_stats table

=======================================================================

**Task 5**

**Data:** Dataset 3

**Instructions:**

* Movies table is Slowly Changing Dimension (SCD) type 2 table.
* Generally, we receive Insert/Update/Delete flag with source file to load Slowly Changing Dimensions (SCD)
* Modify the movies table load script to implement IUD logic

=======================================================================

**Task 6**

**Instructions:**

We have some clients using MySQL database to generate reports. Hence we need to push our movie\_stats table to that MySQL database.