COMMANDS:

1) **INFILE ACCESS:**

```
mysql -u myuser -p --local-infile pTest(Database Name);

SET GLOBAL local_infile = true;
```

2) create schema for the table and load the data using this command

Exporting the SQL FILE:

```
/usr/local/mysql/bin/mysqldump -u root -p pTest > my_movie.sql
```

Importing the SQL FILE:

```
/usr/local/mysql/bin/mysql -u root -p newData < my_movie.sql
```

SCHEMA:

----------MOVIE DATASET----------

```
CREATE TABLE movie
(
     movieID VARCHAR(10),
     primaryTitle VARCHAR(50),
     originalTitle VARCHAR(50),
     isAdult BOOLEAN,
     releaseYear INT(4),
     runtimeMinutes INT,
     averageRating FLOAT,
     numVotes INT,
     PRIMARY KEY(movieID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/movie.tsv" INTO TABLE
movie IGNORE 1 ROWS;
```

----------SERIES DATASET----------

```
CREATE TABLE series
(
     seriesID VARCHAR(10),
     primaryTitle VARCHAR(50),
```

```
      originalTitle VARCHAR(50),
      isAdult BOOLEAN,
      startYear INT,
      averageRating FLOAT,
      numVotes INT,
      PRIMARY KEY(seriesID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/Series.tsv" INTO TABLE
series IGNORE 1 ROWS;
```

----------EPISODE DATASET----------

```
CREATE TABLE episode
(
      episodeID VARCHAR(10),
      seriesID VARCHAR(10),
      seasonNumber INT(4),
      episodeNumber INT(4),
      primaryTitle VARCHAR(50),
      originalTitle VARCHAR(50),
      isAdult BOOLEAN,
      startYear INT(4),
      PRIMARY KEY(episodeID),
      FOREIGN KEY (seriesID) REFERENCES series(seriesID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/Episode.tsv" INTO TABLE
episode IGNORE 1 ROWS;
```

----------EPISODE_ALIAS----------

```
CREATE TABLE episode_alias
(
      episodeID VARCHAR(10),
      ordering INT,
      title VARCHAR(50),
      region VARCHAR(10),
```

```
      isOriginalTitle VARCHAR(50),
      PRIMARY KEY(episodeID,ordering),
      FOREIGN KEY (episodeID) REFERENCES episode(episodeID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/series_alias.tsv" INTO
TABLE episode_alias IGNORE 1 ROWS;
```

----------MOVIE_ALIAS----------

```
CREATE TABLE movie_alias
(
      movieID VARCHAR(10),
      ordering INT,
      title VARCHAR(50),
      region VARCHAR(10),
      isOriginalTitle VARCHAR(50),
      PRIMARY KEY (movieID, ordering),
      FOREIGN KEY (movieID) REFERENCES movie(movieID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/movie_alias.tsv" INTO
TABLE movie_alias IGNORE 1 ROWS;
```

----------SERIES_ALIAS----------

```
CREATE TABLE series_alias
(
      seriesID VARCHAR(10),
      ordering INT,
      title VARCHAR(50),
      region VARCHAR(10),
      isOriginalTitle VARCHAR(50)
      PRIMARY KEY (seriesID, ordering),
      FOREIGN KEY (seriesID) REFERENCES series(seriesID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/series_alias.tsv" INTO
TABLE series_alias IGNORE 1 ROWS;
```

----------ACTOR TABLE----------

```
CREATE TABLE actor
(
      actorID VARCHAR(10),
      primaryName VARCHAR(50),
      birthYear INT(4),
      deathYear INT(4),
      gender VARCHAR(10)
      PRIMARY KEY (actorID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/actor.tsv" INTO TABLE
actor IGNORE 1 ROWS;
```

----------DIRECTOR----------

```
CREATE TABLE director
(
      directorID VARCHAR(10),
      primaryName VARCHAR(50),
      birthYear INT(4),
      deathYear INT(4),
      PRIMARY KEY (directorID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/director.tsv" INTO TABLE
director IGNORE 1 ROWS;
```

----------WRITER----------

```
CREATE TABLE writer
(
      writerID VARCHAR(10),
      primaryName VARCHAR(50),
      birthYear INT(4),
      deathYear INT(4),
      PRIMARY KEY (writerID)
);
```

```
LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/writer.tsv" INTO TABLE
writer IGNORE 1 ROWS;
```

----------ACTOR_KNOWN_FOR_TITLES----------

```
CREATE TABLE actor_known_for_titles
(
     actorID VARCHAR(10),
     knownForTitles VARCHAR(10)
     PRIMARY KEY (actorID, knownForTitles),
     FOREIGN KEY (actorID) REFERENCES actor(actorID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/actorknownfortitles.tsv"
                INTO TABLE actor_known_for_titles IGNORE 1 ROWS;
```

----------WRITER_KNOWN_FOR_TITLES----------

```
CREATE TABLE writer_known_for_titles
(
     writerID VARCHAR(10),
     knownForTitles VARCHAR(10)
     PRIMARY KEY (writerID, knownForTitles),
     FOREIGN KEY (writerID) REFERENCES writer(writerID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/writerknownfortitles.tsv"
     INTO TABLE writer_known_for_titles IGNORE 1 ROWS;
```

----------DIRECTOR_KNOWN_FOR_TITLES----------

```
CREATE TABLE director_known_for_titles
(
     directorID VARCHAR(10),
     knownForTitles VARCHAR(10)
     PRIMARY KEY (directorID, knownForTitles),
     FOREIGN KEY (directorID) REFERENCES director(directorID)
```

```
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/directorknownfortitles.tsv
"
          INTO TABLE director_known_for_titles IGNORE 1 ROWS;
```

----------MOVIE_DIRECTOR_RELATION----------

```
CREATE TABLE movie_director_relation
(
     movieID VARCHAR(10),
     ordering INT,
     directorID VARCHAR(10),
     PRIMARY KEY (movieID, ordering, directorID),
     FOREIGN KEY (movieID) REFERENCES movie(movieID),
     FOREIGN KEY (directorID) REFERENCES director(directorID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/movie_director_relation.ts
v"
          INTO TABLE movie_director_relation IGNORE 1 ROWS;
```

----------MOVIE_WRITER_RELATION----------

```
CREATE TABLE movie_writer_relation
(
     movieID VARCHAR(10),
     ordering INT,
     writerID VARCHAR(10),
     PRIMARY KEY (movieID, ordering, writerID),
     FOREIGN KEY (movieID) REFERENCES movie(movieID),
     FOREIGN KEY (writerID) REFERENCES writer(writerID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/movie_writer_relation.tsv"
          INTO TABLE movie_writer_relation IGNORE 1 ROWS;
```

----------MOVIE_ACTOR_RELATION----------

```
CREATE TABLE movie_actor_relation
(
     movieID VARCHAR(10),
     ordering INT,
     actorID VARCHAR(10),
     Characters VARCHAR(10),
     PRIMARY KEY (movieID, ordering, actorID),
     FOREIGN KEY (movieID) REFERENCES movie(movieID),
     FOREIGN KEY (actorID) REFERENCES actor(actorID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/movie_actor_relation.tsv"
          INTO TABLE movie_actor_relation IGNORE 1 ROWS;
```

----------SERIES_DIRECTOR_RELATION----------

```
CREATE TABLE series_director_relation
(
     seriesID VARCHAR(10),
     ordering INT,
     directorID VARCHAR(10),
     PRIMARY KEY (seriesID, ordering, directorID),
     FOREIGN KEY (seriesID) REFERENCES series(seriesID),
     FOREIGN KEY (directorID) REFERENCES director(directorID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/series_director_relation.t
sv"
          INTO TABLE series_director_relation IGNORE 1 ROWS;
```

----------SERIES_WRITER_RELATION----------

```
CREATE TABLE series_writer_relation
(
     seriesID VARCHAR(10),
     ordering INT,
```

```
     writerID VARCHAR(10),
     PRIMARY KEY (seriesID, ordering, writerID),
     FOREIGN KEY (seriesID) REFERENCES series(seriesID),
     FOREIGN KEY (writerID) REFERENCES writer(writerID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/series_writer_relation.tsv
"
          INTO TABLE series_writer_relation IGNORE 1 ROWS;
```

----------SERIES_ACTOR_RELATION----------

```
CREATE TABLE series_actor_relation
(
     seriesID VARCHAR(10),
     ordering INT,
     actorID VARCHAR(10),
     characters VARCHAR(10),
     PRIMARY KEY (seriesID, ordering, actorID),
     FOREIGN KEY (seriesID) REFERENCES series(seriesID),
     FOREIGN KEY (actorID ) REFERENCES actor(actorID)
);

LOAD DATA LOCAL INFILE
"/Users/viraj1604/Documents/Final_Datasets/series_actor_relation.tsv"
          INTO TABLE series_actor_relation IGNORE 1 ROWS;
```

---------- QUERIES ----------

1) All details for the movie for the particular actor which is also known for other titles:

```
SELECT * FROM movie
WHERE movieID IN (
SELECT knownForTitles FROM actor_known_for_titles WHERE actorID = (
SELECT actorID FROM actor WHERE primaryName = 'John Belushi'));
```

```
mysql> SELECT * FROM movie  WHERE movieID IN  ( SELECT knownForTitles FROM actor_known_for_titles WHERE actorID =  ( SELECT
actorID FROM actor WHERE primaryName = 'John Belushi' ));
+-----------+--------------------+--------------------+---------+-------------+----------------+---------------+----------+
| movieID   | primaryTitle       | originalTitle      | isAdult | releaseYear | runtimeMinutes | averageRating | numVotes |
+-----------+--------------------+--------------------+---------+-------------+----------------+---------------+----------+
| tt0077975 | Animal House       | Animal House       |       0 |        1978 |            109 |           7.6 |   104112 |
| tt0078723 | 1941               | 1941               |       0 |        1979 |            118 |           5.8 |    29008 |
| tt0080455 | The Blues Brothers | The Blues Brothers |       0 |        1980 |            133 |           7.9 |   165665 |
+-----------+--------------------+--------------------+---------+-------------+----------------+---------------+----------+
3 rows in set (0.75 sec)
```

2) No. of movies directed by the director and in which genre alphabetically by name and highest count as per genre?

```
SELECT distinct d.primaryName, mg.genres, count(m.movieID)
FROM  movie m, director d, movie_director_relation md, movie_genre mg
WHERE md.movieID = m.movieID
AND mg.movieID = m.movieID
AND md.directorID = d.directorID
GROUP BY d.primaryName,mg.genres
ORDER BY d.primaryName, count(m.movieID) DESC
LIMIT 200;
```

```
mysql> SELECT distinct d.primaryName, mg.genres, count(m.movieID) FROM  movie m, director d, movie_director_relation md, mov
ie_genre mg WHERE md.movieID = m.movieID AND mg.movieID = m.movieID AND md.directorID = d.directorID GROUP BY d.primaryName,
mg.genres ORDER BY d.primaryName, count(m.movieID) DESC LIMIT 200;
+----------------------------------------+------------+------------------+
| primaryName                            | genres     | count(m.movieID) |
+----------------------------------------+------------+------------------+
| 'Atlas' Ramachandran                   | Drama      |                1 |
| 'Bob' Insert                           | Adult      |                1 |
| 'Chico' Hernandez                      | Family     |                3 |
| 'Fish' Mark Pickart                    | Comedy     |                1 |
| 'Fish' Mark Pickart                    | Short      |                1 |
| 'Hollywood' Steve Huey                 | Adventure  |                1 |
| 'Hollywood' Steve Huey                 | Action     |                1 |
```

3) Top Ten rated movies:

```
SELECT * FROM(
SELECT m.primaryTitle, m.averageRating FROM movie AS m
UNION
SELECT s.primaryTitle, s.averageRating FROM series AS s ) AS tt
ORDER BY tt.averageRating DESC LIMIT 10;
```

```
mysql> SELECT * FROM ( SELECT m.primaryTitle, m.averageRating FROM movie AS m  UNION SELECT s.primaryTitle, s.averageRating
FROM series AS s ) AS tt  ORDER BY tt.averageRating DESC LIMIT 10;
+----------------------------+---------------+
| primaryTitle               | averageRating |
+----------------------------+---------------+
| The Sharing Circle         |          8400 |
| In Town Tonight            |          5220 |
| Great Chefs of the World   |          4800 |
| Desire                     |          3900 |
| White Deer Plain           |          3825 |
| Kara melek                 |          3600 |
| Salmeboka minutt for minutt |          3600 |
| Alim Dayi                  |          3000 |
| Kôya no surônin            |          2925 |
| Pepsi Top of the Pops      |          2565 |
+----------------------------+---------------+
```

4) Top ten rated movies and their actor, writer and director.

```
Select m.primaryTitle, m.averageRating, a.primaryName ,
ma.characters, d.primaryName
from movie m, actor a, movie_actor_relation ma,
movie_director_relation md, director d
where m.movieID = ma.movieID
and a.actorID = ma.actorID
and d.directorID = md.directorID
and md.movieID = m.movieID
order by averageRating DESC limit 10;
```

```
mysql> Select m.primaryTitle, m.averageRating, a.primaryName , ma.characters, d.primaryName from movie m, actor a, movie_act
or_relation ma, movie_director_relation md, director d where m.movieID = ma.movieID  and a.actorID = ma.actorID  and d.direc
torID = md.directorID  and md.movieID = m.movieID order by averageRating DESC limit 10;
+----------------+---------------+-----------------------+------------+--------------------+
| primaryTitle   | averageRating | primaryName           | characters | primaryName        |
+----------------+---------------+-----------------------+------------+--------------------+
| Sworn          |            10 | David C. Hernandez    | "[""Hero"" | Wesley Quinn       |
| Sworn          |            10 | Ramsey Krull          | "[""Thorn" | Wesley Quinn       |
| Sworn          |            10 | Chase McKendry        | "[""Hawk F | Wesley Quinn       |
| Hand Rolled    |            10 | Peter Weller          | "[""Narrat | Steve Gherebean    |
| Pink Taxi      |            10 | Argiris Tafralidis    | "[""Manage | Gabriel Athanasiou |
| Pink Taxi      |            10 | Christina Tafralidou  | "[""Assist | Gabriel Athanasiou |
| Pink Taxi      |            10 | Apostolos Tsamis      | "[""Pink T | Gabriel Athanasiou |
| Pink Taxi      |            10 | Giorgos Vezirgiannidis | "[""Vezos" | Gabriel Athanasiou |
| Crooked Strings |            10 | Stephen Brown         | "[""Office | Tone Melo          |
| Crooked Strings |            10 | K'Sean Collard        | "[""Kenny  | Tone Melo          |
+----------------+---------------+-----------------------+------------+--------------------+
10 rows in set (0.79 sec)
```

5) Top Movie Actors and their character name.

```
Select m.primaryTitle, m.averageRating, a.primaryName , ma.Characters
from movie m
INNER JOIN movie_actor_relation ma ON  m.movieID = ma.movieID
INNER JOIN actor a ON a.actorID = ma.actorID
ORDER BY m.averageRating
LIMIT 10;
```

```
mysql> Select m.primaryTitle, m.averageRating, a.primaryName , ma.Characters from movie m  INNER JOIN movie_actor_relation m
a ON  m.movieID = ma.movieID  INNER JOIN actor a ON a.actorID = ma.actorID  ORDER BY m.averageRating LIMIT 10;
+---------------------------------+---------------+---------------------+------------+
| primaryTitle                    | averageRating | primaryName         | Characters |
+---------------------------------+---------------+---------------------+------------+
| Irezumi dorei fujin             |             0 | Naomi Hagio         | NULL       |
| Irezumi dorei fujin             |             0 | Tôru Ibuki          | NULL       |
| Park Life                       |             0 | Molly Glover        | ["""Amy""] |
| Park Life                       |             0 | Alex McDonald-Smith | ["""Olly"" |
| Park Life                       |             0 | Joe Warriner        | ["""Charli |
| Cup of Kindness for the Holidays |             0 | Patrick Stoffer     | ["""Danny" |
| Cup of Kindness for the Holidays |             0 | Rebecca Noble       | ["""Sam""] |
| Golden Pussy                    |             0 | Alexander Hilson    | ["""Golden |
| Golden Pussy                    |             0 | Cole Chauvin        | ["""Profes |
| Golden Pussy                    |             0 | Matt Curtis         | ["""Chief  |
+---------------------------------+---------------+---------------------+------------+
10 rows in set (0.79 sec)
```

6) Top 10 movies on basis of region

```
Select m.primaryTitle, ma.region, ma.title,m.averageRating
FROM movie m
INNER JOIN movie_alias ma ON  m.movieID = ma.movieID
AND ma.region != 'NULL'
ORDER BY m.averageRating DESC
LIMIT 10;
```

```
mysql> Select m.primaryTitle, ma.region, ma.title,m.averageRating FROM movie m  INNER JOIN movie_alias ma ON  m.movieID = ma
.movieID  AND ma.region != 'NULL' ORDER BY m.averageRating DESC LIMIT 10;
+---------------------------------------+--------+---------------------------------------+---------------+
| primaryTitle                          | region | title                                 | averageRating |
+---------------------------------------+--------+---------------------------------------+---------------+
| Pink Taxi                             | GR     | Ροζ Ταξί                              |            10 |
| Crooked Strings                       | US     | Crooked Strings                       |            10 |
| The Steampunk Adventures of Salem Tusk | US     | The Steampunk Adventures of Salem Tusk |            10 |
| Second Unit: A Mockumentary           | US     | Second Unit: A Mockumentary           |            10 |
| Second Unit: A Mockumentary           | US     | Second Unit                           |            10 |
| Life                                  | RS     | Život                                 |            10 |
| Life                                  | XWW    | Life                                  |            10 |
| Life                                  | PT     | Vida                                  |            10 |
| The Cassie Confession                 | US     | The Cassie Confession                 |            10 |
| Movie Night                           | IL     | Erev MeHasratim                       |            10 |
+---------------------------------------+--------+---------------------------------------+---------------+
10 rows in set (0.78 sec)
```

7) First 50 entries WriterName according to their series and their genres.

```
SELECT distinct w.primaryName as WriterName, s.primaryTitle,
sg.genres
FROM writer w, series_writer_relation sw, series s, series_genre sg
WHERE w.writerID = sw.writerID
AND s.seriesID = sw.seriesID
AND s.seriesID = sg.seriesID
AND w.writerID NOT IN
(
SELECT w1.writerID
FROM writer w1, movie_writer_relation mw
```

```
WHERE w1.writerID = mw.writerID)
LIMIT 10;
```

```
mysql> SELECT distinct w.primaryName as WriterName, s.primaryTitle, sg.genres FROM writer w, series_writer_relation sw, seri
es s, series_genre sg WHERE w.writerID = sw.writerID AND s.seriesID = sw.seriesID AND s.seriesID = sg.seriesID AND w.writerI
D NOT IN  ( SELECT w1.writerID FROM writer w1, movie_writer_relation mw WHERE w1.writerID = mw.writerID) LIMIT 10;
+------------------+------------------+------------+
| WriterName       | primaryTitle     | genres     |
+------------------+------------------+------------+
| Jim Carrey       | I'm Dying Up Here | Comedy     |
| Jim Carrey       | I'm Dying Up Here | Drama      |
| Leonardo DiCaprio | Greensburg       | Reality-TV |
| Whoopi Goldberg  | Strong Medicine  | Drama      |
| Whoopi Goldberg  | Just for Kicks   | Comedy     |
| Whoopi Goldberg  | Just for Kicks   | Drama      |
| Whoopi Goldberg  | Just for Kicks   | Family     |
| Fran Drescher    | La nany          | Comedy     |
| Fran Drescher    | Niania           | Comedy     |
| Fran Drescher    | La niñera        | Comedy     |
+------------------+------------------+------------+
10 rows in set (0.01 sec)
```

8) Count of movies in each genre, according to the highest first HAVING movies greater than 20000.

```
SELECT distinct genres, count(movieID)
FROM movie_genre
GROUP BY genres
HAVING count(movieID) > 20000;
```

```
mysql> SELECT distinct genres, count(movieID)  FROM movie_genre  GROUP BY genres  HAVING count(movieID) > 20000;
+-------------+----------------+
| genres      | count(movieID) |
+-------------+----------------+
| Documentary |         232885 |
| Short       |         631447 |
| Animation   |          53648 |
| Comedy      |         256722 |
| Romance     |          67011 |
| Drama       |         411227 |
| Family      |          43651 |
| Fantasy     |          36413 |
| Horror      |          60857 |
| Biography   |          28161 |
| Music       |          36964 |
| Crime       |          49718 |
| Adventure   |          37542 |
| History     |          21935 |
| Action      |          66067 |
| Mystery     |          31903 |
| Sci-Fi      |          30667 |
| Thriller    |          58272 |
+-------------+----------------+
18 rows in set (1.16 sec)
```

9) Series name and the actor name where its actor also acted in a movie:

```
Select a.primaryName as ActorName, s.primaryTitle as SeriesName
from Series s, actor a, series_actor_relation sa
where s.seriesID = sa.seriesID
```

```
and a.actorID = sa.actorID
and a.actorID in (Select distinct a.actorID
from movie m1, actor a1, movie_actor_relation ma
where m1.movieID = ma.movieID
and a1.actorID = ma.actorID) LIMIT 10;
```

```
mysql> Select a.primaryName as ActorName, s.primaryTitle as SeriesName from Series s, actor a, series_actor_relation sa wher
e s.seriesID = sa.seriesID and a.actorID = sa.actorID and a.actorID in (Select distinct a.actorID from movie m1, actor a1, m
ovie_actor_relation ma where m1.movieID = ma.movieID and a1.actorID = ma.actorID) LIMIT 10;
+---------------+---------------------------------------+
| ActorName     | SeriesName                            |
+---------------+---------------------------------------+
| Fred Astaire  | It Takes a Thief                      |
| Fred Astaire  | The Wasp of the Anime                 |
| Lauren Bacall | Mr. Broadway                          |
| Lauren Bacall | The General Motors Playwrights Theater |
| Lauren Bacall | CBS: On the Air                       |
| Lauren Bacall | Private View                          |
| Lauren Bacall | The Living Edens                      |
| Lauren Bacall | Ron Russell's Set the Record Straight |
| Marlon Brando | The Godfather Saga                    |
| Richard Burton | The James Mason Show                  |
+---------------+---------------------------------------+
10 rows in set (0.01 sec)
```