|  |  |  |
| --- | --- | --- |
| **Name: Rohan C** | **SRN: PES1UG20CS345** | **Section: F** |
| **Date: 03/11/2022** | **Project SQL File** |

-- phpMyAdmin SQL Dump

-- version 5.2.0

-- https://www.phpmyadmin.net/

--

-- Host: localhost

-- Generation Time: Nov 30, 2022 at 08:35 AM

-- Server version: 10.4.21-MariaDB

-- PHP Version: 8.1.6

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

START TRANSACTION;

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

--

-- Database: `IMDb`

--

-- Delete IMDb database if necessary

DROP DATABASE IF EXISTS IMDb;

-- Create IMDb database

CREATE DATABASE IMDb;

-- Use IMDb database

USE IMDb;

-- Character set

-- want to be able to distinguish text with accents

ALTER DATABASE IMDb CHARACTER SET utf8mb4 COLLATE utf8mb4\_bin;

-- Drop old tables if they exist

-- DROP TABLE IF EXISTS Titles;

-- DROP TABLE IF EXISTS Title\_ratings;

-- DROP TABLE IF EXISTS Aliases;

-- DROP TABLE IF EXISTS Alias\_types;

-- DROP TABLE IF EXISTS Alias\_attributes;

-- DROP TABLE IF EXISTS Episode\_belongs\_to;

-- DROP TABLE IF EXISTS Title\_genres;

-- DROP TABLE IF EXISTS Names\_;

-- DROP TABLE IF EXISTS Name\_worked\_as;

-- DROP TABLE IF EXISTS Had\_role;

-- DROP TABLE IF EXISTS Known\_for;

-- DROP TABLE IF EXISTS Directors;

-- DROP TABLE IF EXISTS Writers;

-- DROP TABLE IF EXISTS Principals;

-- Create tables only

CREATE TABLE Titles (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

title\_type VARCHAR(50),

primary\_title TEXT, -- some are really long

original\_title TEXT, -- some are really long

is\_adult BOOLEAN,

start\_year INTEGER, -- add better domain here (>1800)

end\_year INTEGER, -- add better domain here (>0)

runtime\_minutes INTEGER -- add better domain here (>0)

);

CREATE TABLE Title\_ratings (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

average\_rating FLOAT,

num\_votes INTEGER

);

CREATE TABLE Aliases (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

ordering INTEGER NOT NULL, -- not null bc PK

title TEXT NOT NULL,

region CHAR(4),

language CHAR(4),

is\_original\_title BOOLEAN

);

CREATE TABLE Alias\_types (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

ordering INTEGER NOT NULL, -- not null bc PK

type VARCHAR(255) NOT NULL-- Only stored if not null

);

CREATE TABLE ALias\_attributes (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

ordering INTEGER NOT NULL, -- not null bc PK

attribute VARCHAR(255) NOT NULL -- only stored if not null

);

CREATE TABLE Episode\_belongs\_to (

episode\_title\_id VARCHAR(255) NOT NULL, -- not null bc PK

parent\_tv\_show\_title\_id VARCHAR(255) NOT NULL,

season\_number INTEGER,

episode\_number INTEGER

);

CREATE TABLE Title\_genres (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

genre VARCHAR(255) NOT NULL -- not null bc PK

);

-- Names and name is a reserved word in MySQL, so we add an underscore

CREATE TABLE Names\_ (

name\_id VARCHAR(255) NOT NULL, -- not null bc PK

name\_ VARCHAR(255) NOT NULL, -- everybody has a name

birth\_year SMALLINT, -- add a better domain here

death\_year SMALLINT -- add a better domain here

);

CREATE TABLE Name\_worked\_as (

name\_id VARCHAR(255) NOT NULL, -- not null bc PK

profession VARCHAR(255) NOT NULL -- not null bc PK

);

-- NOTE: All 3 must must be used as the primary key

-- role is a reserved word in MySQL, so we add an underscore

CREATE TABLE Had\_role (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

name\_id VARCHAR(255) NOT NULL, -- not null bc PK

role\_ TEXT NOT NULL -- not null bc PK

);

CREATE TABLE Known\_for (

name\_id VARCHAR(255) NOT NULL, -- not null bc PK

title\_id VARCHAR(255) NOT NULL -- not null bc PK

);

CREATE TABLE Directors (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

name\_id VARCHAR(255) NOT NULL -- not null bc PK

);

CREATE TABLE Writers (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

name\_id VARCHAR(255) NOT NULL -- not null bc PK

);

CREATE TABLE Principals (

title\_id VARCHAR(255) NOT NULL, -- not null bc PK

ordering TINYINT NOT NULL, -- not null bc PK

name\_id VARCHAR(255) NOT NULL,

job\_category VARCHAR(255),

job TEXT

);

-- SHOW VARIABLES LIKE "local\_infile";

SET GLOBAL local\_infile = 1;

-- Load Aliases.tsv into Aliases table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Aliases.tsv'

INTO TABLE Aliases

COLUMNS TERMINATED BY '\t';

-- Load Alias\_attributes.tsv into Alias\_attributes table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Alias\_attributes.tsv'

INTO TABLE Alias\_attributes

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Alias\_types.tsv into Alias\_types table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Alias\_types.tsv'

INTO TABLE Alias\_types

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Directors.tsv into Directors table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Directors.tsv'

INTO TABLE Directors

COLUMNS TERMINATED BY '\t';

-- Load Writers.tsv into Writers table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Writers.tsv'

INTO TABLE Writers

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Episode\_belongs\_to.tsv into Episode\_belongs\_to table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Episode\_belongs\_to.tsv'

INTO TABLE Episode\_belongs\_to

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Names\_.tsv into Names\_ table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Names\_.tsv'

INTO TABLE Names\_

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Name\_worked\_as.tsv into Name\_worked\_as table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Name\_worked\_as.tsv'

INTO TABLE Name\_worked\_as

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Known\_for.tsv into Known\_for table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Known\_for.tsv'

INTO TABLE Known\_for

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Principals.tsv into Principals table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Principals.tsv'

INTO TABLE Principals

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Had\_role.tsv into Had\_role table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Had\_role.tsv'

INTO TABLE Had\_role

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Titles.tsv into Titles table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Titles.tsv'

INTO TABLE Titles

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Title\_genres.tsv into Title\_genres table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Title\_genres.tsv'

INTO TABLE Title\_genres

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Load Title\_ratings.tsv into Title\_ratings table

LOAD DATA LOCAL INFILE '/Users/rohan/Documents/Programming/DBMS/Project/MySQL\_IMDb\_Project-master/Title\_ratings.tsv'

INTO TABLE Title\_ratings

COLUMNS TERMINATED BY '\t'

IGNORE 1 LINES;

-- Add constraints individually

ALTER TABLE Names\_

ADD CONSTRAINT Names\_pri\_key PRIMARY KEY (name\_id);

ALTER TABLE Titles

ADD CONSTRAINT Titles\_pri\_key PRIMARY KEY (title\_id);

ALTER TABLE Aliases

ADD CONSTRAINT Aliases\_pri\_key PRIMARY KEY (title\_id,ordering);

ALTER TABLE Alias\_attributes

ADD CONSTRAINT Alias\_attributes\_pri\_key PRIMARY KEY (title\_id,ordering);

ALTER TABLE Alias\_types

ADD CONSTRAINT Alias\_types\_pri\_key PRIMARY KEY (title\_id,ordering);

ALTER TABLE Directors

ADD CONSTRAINT Directors\_pri\_key PRIMARY KEY (title\_id,name\_id);

ALTER TABLE Directors

ADD CONSTRAINT Directors\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

ALTER TABLE Writers

ADD CONSTRAINT Writers\_pri\_key PRIMARY KEY (title\_id,name\_id);

ALTER TABLE Writers

ADD CONSTRAINT Writers\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

ALTER TABLE Episode\_belongs\_to

ADD CONSTRAINT Episode\_belongs\_to\_pri\_key PRIMARY KEY (episode\_title\_id);

ALTER TABLE Episode\_belongs\_to

ADD CONSTRAINT Episode\_belongs\_to\_ep\_title\_id\_fkey FOREIGN KEY (episode\_title\_id) REFERENCES Titles(title\_id);

ALTER TABLE Name\_worked\_as

ADD CONSTRAINT Name\_worked\_as\_pri\_key PRIMARY KEY (name\_id,profession);

ALTER TABLE Name\_worked\_as

ADD CONSTRAINT Name\_worked\_as\_name\_id\_fkey FOREIGN KEY (name\_id) REFERENCES Names\_(name\_id);

ALTER TABLE Known\_for

ADD CONSTRAINT Known\_for\_pri\_key PRIMARY KEY (name\_id,title\_id);

ALTER TABLE Known\_for

ADD CONSTRAINT Known\_for\_name\_id\_fkey FOREIGN KEY (name\_id) REFERENCES Names\_(name\_id);

ALTER TABLE Principals

ADD CONSTRAINT Principals\_pri\_key PRIMARY KEY (title\_id,ordering);

-- role\_ is TEXT, so we need to add indexing length (255)

ALTER TABLE Had\_role

ADD CONSTRAINT Had\_role\_pri\_key PRIMARY KEY (title\_id,name\_id,role\_(255));

ALTER TABLE Title\_genres

ADD CONSTRAINT Title\_genres\_pri\_key PRIMARY KEY (title\_id,genre);

ALTER TABLE Title\_genres

ADD CONSTRAINT Title\_genres\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

ALTER TABLE Title\_ratings

ADD CONSTRAINT Title\_ratings\_pri\_key PRIMARY KEY (title\_id);

ALTER TABLE Title\_ratings

ADD CONSTRAINT Title\_ratings\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

-- Issues with missing data in title.basics.tsv.gz, name.basics.tsv.gz, ...

-- Disable foreign key check lock

SET foreign\_key\_checks = 0;

-- Aliases has titles that do not exist in Titles, i.e., there are entries in

-- IMDb's title.akas.tsv.gz that are not present in title.basics.tsv.gz. The same

-- issue arises when setting the foreign key for the Alias\_attributes and

-- Alias\_types tables.

ALTER TABLE Aliases

ADD CONSTRAINT Aliases\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

-- SELECT \* FROM Aliases AS A WHERE A.title\_id NOT IN (SELECT title\_id FROM Titles) LIMIT 10;

-- SELECT \* FROM Titles WHERE title\_id = 'tt0021006';

-- SELECT \* FROM Aliases WHERE title\_id = 'tt0021006';

ALTER TABLE Alias\_attributes

ADD CONSTRAINT Alias\_attributes\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

ALTER TABLE Alias\_types

ADD CONSTRAINT Alias\_types\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

-- Ditto for Episode\_belongs\_to table.

ALTER TABLE Episode\_belongs\_to

ADD CONSTRAINT Episode\_belongs\_to\_show\_title\_id\_fkey FOREIGN KEY (parent\_tv\_show\_title\_id) REFERENCES Titles(title\_id);

-- SELECT DISTINCT E.parent\_tv\_show\_title\_id

-- FROM Episode\_belongs\_to AS E

-- WHERE E.parent\_tv\_show\_title\_id NOT IN (SELECT title\_id FROM Titles)

-- LIMIT 10;

-- SELECT \* FROM Titles WHERE title\_id = 'tt6403604';

-- SELECT \* FROM Episode\_belongs\_to WHERE parent\_tv\_show\_title\_id = 'tt6403604';

ALTER TABLE Known\_for

ADD CONSTRAINT Known\_for\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

-- SELECT \* FROM Known\_for AS K WHERE K.title\_id NOT IN (SELECT title\_id FROM Titles) LIMIT 10;

-- SELECT \* FROM Titles WHERE title\_id = 'tt0331007';

-- SELECT \* FROM Known\_for WHERE title\_id = 'tt0331007' LIMIT 5;

ALTER TABLE Principals

ADD CONSTRAINT Principals\_name\_id\_fkey FOREIGN KEY (name\_id) REFERENCES Names\_(name\_id);

-- SELECT \* FROM Principals AS P WHERE P.name\_id NOT IN (SELECT name\_id FROM Names\_) LIMIT 10;

-- SELECT \* FROM Names\_ WHERE name\_id = 'nm0730493';

-- SELECT \* FROM Principals WHERE name\_id = 'nm0730493';

ALTER TABLE Principals

ADD CONSTRAINT Principals\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

-- SELECT \* FROM Principals AS P WHERE P.title\_id NOT IN (SELECT title\_id FROM Titles) LIMIT 10;

-- SELECT \* FROM Titles WHERE title\_id = 'tt0047941';

-- SELECT \* FROM Principals WHERE title\_id = 'tt0047941';

ALTER TABLE Had\_role

ADD CONSTRAINT Had\_role\_title\_id\_fkey FOREIGN KEY (title\_id) REFERENCES Titles(title\_id);

-- SELECT \* FROM Had\_role AS H WHERE H.title\_id NOT IN (SELECT title\_id FROM Titles) LIMIT 10;

-- SELECT \* FROM Titles WHERE title\_id = 'tt0047941';

-- SELECT \* FROM Had\_role WHERE title\_id = 'tt0047941';

ALTER TABLE Had\_role

ADD CONSTRAINT Had\_role\_name\_id\_fkey FOREIGN KEY (name\_id) REFERENCES Names\_(name\_id);

-- SELECT \* FROM Had\_role AS H WHERE H.name\_id NOT IN (SELECT name\_id FROM Names\_) LIMIT 10;

-- SELECT \* FROM Names\_ WHERE name\_id = 'nm0241605';

-- SELECT \* FROM Had\_role WHERE name\_id = 'nm0241605';

ALTER TABLE Directors

ADD CONSTRAINT Directors\_name\_id\_fkey FOREIGN KEY (name\_id) REFERENCES Names\_(name\_id);

-- SELECT \* FROM Directors AS D WHERE D.name\_id NOT IN (SELECT name\_id FROM Names\_) LIMIT 10;

-- SELECT \* FROM Names\_ WHERE name\_id = 'nm10576972';

-- SELECT \* FROM Directors WHERE name\_id = 'nm10576972';

ALTER TABLE Writers

ADD CONSTRAINT Writers\_name\_id\_fkey FOREIGN KEY (name\_id) REFERENCES Names\_(name\_id);

-- SELECT \* FROM Writers AS W WHERE W.name\_id NOT IN (SELECT name\_id FROM Names\_) LIMIT 10;

-- SELECT \* FROM Names\_ WHERE name\_id = 'nm10032129';

-- SELECT \* FROM Writers WHERE name\_id = 'nm10032129';

-- Enable foreign key check lock

SET foreign\_key\_checks = 1;

-- Add at least one index per table

-- Alias\_attributes

CREATE INDEX Alias\_attributes\_index ON Alias\_attributes(title\_id);

-- Alias\_types

CREATE INDEX Alias\_types\_index ON Alias\_types(title\_id);

-- Aliases

CREATE INDEX Aliases\_index ON Aliases(title\_id);

-- Directors

CREATE INDEX Directors\_title\_id\_index ON Directors(title\_id);

CREATE INDEX Directors\_name\_id\_index ON Directors(name\_id);

-- Episode\_belongs\_to

CREATE INDEX Episode\_belongs\_to\_ep\_title\_id\_index ON Episode\_belongs\_to(episode\_title\_id);

CREATE INDEX Episode\_belongs\_to\_show\_title\_id\_index ON Episode\_belongs\_to(parent\_tv\_show\_title\_id);

-- Had\_role

CREATE INDEX Had\_role\_title\_id\_index ON Had\_role(title\_id);

CREATE INDEX Had\_role\_name\_id\_index ON Had\_role(name\_id);

-- Known\_for

CREATE INDEX Known\_for\_index ON Known\_for(name\_id);

-- Name\_worked\_as

CREATE INDEX Name\_worked\_as\_index ON Name\_worked\_as(profession);

-- Names\_

CREATE INDEX Names\_index ON Names\_(name\_id);

-- Principals

CREATE INDEX Principals\_index ON Principals(title\_id);

-- Title\_genres

CREATE INDEX Title\_genres\_title\_id\_index ON Title\_genres(title\_id);

CREATE INDEX Title\_genres\_genre\_index ON Title\_genres(genre);

-- Title\_ratings

CREATE INDEX Title\_ratings\_index ON Title\_ratings(title\_id);

-- Titles

CREATE INDEX Titles\_index ON Titles(title\_id);

-- Writers

CREATE INDEX Writers\_title\_id\_index ON Writers(title\_id);

CREATE INDEX Writers\_name\_id\_index ON Writers(name\_id);

-- ALL QUERIES RUN AS A PART OF THE PROJECT

-- Query 1

-- Select All titles that are of Horror Genre, Not adult, and released later than 2021

SELECT \* FROM Titles AS T

JOIN Title\_genres AS TG ON T.title\_id = TG. title\_id

WHERE genre LIKE 'Horror%' AND is\_adult = 0 AND start\_year>2021

LIMIT 25;

-- Query 2

-- Select the Name and Role of all the cast members of Sholay?

SELECT name\_, role\_

FROM Names\_ AS N

JOIN Had\_role AS H ON N.name\_id = H.name\_id

WHERE title\_id IN

(SELECT title id FROM Titles

WHERE primary\_title LIKE 'Sholay');

-- Query 3

-- Names And id's of actors who have played the role of James Bond

SELECT N.name\_id, N.name FROM Names\_ AS N

INNER JOIN Had\_role AS H WHERE N.name\_id = H.name\_id

AND H.role\_ LIKE '%James Bond%'

AND title\_id IN

(SELECT title\_id FROM Titles

WHERE title\_type LIKE 'movie')

GROUP BY name\_id;

-- Query 4

-- List All Movies of Tom Cruise and the Role Played in them in Ascending order of their release

SELECT T.title\_id, T.primary\_title, T.start\_year, H.role\_

FROM Titles AS T JOIN Had\_role AS H ON T.title\_id = H.title\_id

WHERE H.name\_id =

(SELECT name\_id FROM Names\_ AS N WHERE N.name\_ LIKE 'Tom Cruise')

AND T.title\_type LIKE 'movie' ORDER BY T.start\_year ASC;

--Aggregate Queries

-- Query 1

-- Total number of titles of type 'movie' grouped by genre in descending order?

SELECT G.genre, COUNT (G. genre) AS Count

FROM Title\_genres AS G, Titles AS T

WHERE T. title\_id = G.title\_id

AND T.title\_type = 'movie'

GROUP BY genre

ORDER BY Count DESC;

-- Query 2

-- How many times has an actor played the role of James Bond?

SELECT N.name\_id, N. name\_, COUNT (\*) AS number\_of\_films

FROM Names\_ AS N, Had\_role AS H, Titles AS T

WHERE H. role\_ LIKE 'James Bond'

AND T. title\_type LIKE 'movie'

AND T.title\_id = H.title\_id

AND N. name\_id = H. name\_

GROUP BY N. name\_id;

-- Query 3

-- Most popular episode of The tv show 'The X Files'?

CREATE OR REPLACE VIEW Q( season\_number, episode\_number,primary\_title, average\_rating)

AS SELECT E. season\_number, E.episode\_number, T2.primary\_title, R.average\_rating

FROM Titles AS T1, Titles AS T2, Episode\_belongs\_to AS E, Title\_ratings AS R

WHERE T1.primary\_title = 'The X-Files'

AND T1. title\_type = 'tvSeries'

AND T1.title\_id = E.parent\_tv\_show\_title\_id

AND T2.title\_type = 'tvEpisode'

AND T2.title\_id = E.episode\_title\_id

AND T2.title\_id = R.title\_id

ORDER BY E. season\_number, E.episode\_number;

SELECT Q. season\_number, Q.episode\_number, Q.primary\_title, Q.average\_rating

FROM O

WHERE Q.average\_rating = (SELECT MAX(Q.average\_rating) FROM Q);

-- Query 4

-- How many episodes were there in The X-Files per season? And what was the average of the episode ratings ?

CREATE OR REPLACE VIEW Q(season\_number, episode\_number,primary\_title, average\_rating)

AS SELECT E. season\_number, E.episode\_number, T2.primary\_title, R.average\_rating

FROM Titles AS T1, Titles AS T2, Episode\_belongs\_to AS E, Title\_ratings AS R

WHERE TI.primary\_title = 'The X-Files'

AND T1.title\_type = 'tvSeries'

AND TI.title\_id = E.parent\_tv\_show\_title\_id

AND T2.title\_type = 'tvEpisode'

AND T2.title\_id = E.episode\_title\_id

AND T2.title\_id = R.title\_id

ORDER BY E. season\_number, E.episode\_number;

SELECT Q. season\_number, COUNT (\*) AS Number\_of\_episodes, AVG(Q.average\_rating) AS Average\_of\_ep\_average\_ratings

FROM Q

GROUP BY Q.season\_number

ORDER BY Q.season\_number;

-- Set Queries

-- Query 1

-- Movies Directed by both Karan Johar and Sanjay Leela Bhansali

SELECT T.title id, T.primary title FROM Titles AS T JOIN Directors AS D ON T.title\_id D.title id WHERE D. name id = (SELECT N.name\_id FROM Names\_ AS N WHERE N.name\_ = 'Karan Johar')

UNION

SELECT T.title id, T.primary title FROM Titles AS T JOIN Directors AS D ON T.title\_id D.title id WHERE D. name id = (SELECT N.name\_id FROM Names\_ AS N WHERE N.name\_ = 'Sanjay Leela Bhansali');

-- Query 2

-- Movies starring both Chris Hemsworth and Chris Evans

(SELECT T.title\_id, T.primary\_title, T.start\_year FROM Titles AS T

JOIN Had\_role AS H ON T.title\_id=H.title\_id

WHERE H.name\_id = (SELECT N. name\_id FROM Names\_ AS N WHERE N. name\_ LIKE 'Chris Hemsworth'

AND T.title\_type LIKE 'movie'))

INTERSECT

(SELECT T.title\_id, T.primary\_title, T.start\_year FROM Titles AS T

JOIN Had\_role AS H ON T.title\_id=H.title\_id

WHERE H.name\_id = (SELECT N. name\_id FROM Names\_ AS N WHERE N. name\_ LIKE 'Chris Evans'))

-- Query 3

-- Find all movies starring either Ranveer Singh and Deepika Padukone

-- OR Ranbir Kapoor and Alia Bhatt

((SELECT DISTINCT T.title\_id, T.primary\_title, T.start\_year FROM Titles AS T JOIN Had\_role AS H ON T.title\_id=H.title\_id WHERE H.name\_id IN (SELECT N. name\_id FROM

Names\_ AS N WHERE N.name\_ LIKE 'Deepika Padukone') AND T.title\_type LIKE 'movie')

INTERSECT

(SELECT DISTINCT T.title\_id, T.primary\_title, T.start\_year FROM Titles AS T JOIN Had\_role AS H ON T.title\_id=H.title\_id WHERE H.name\_id IN (SELECT N. name\_id FROM

Names\_ AS N WHERE N.name\_ LIKE 'Ranveer Singh') AND T.title\_type LIKE 'movie'))

UNION

((SELECT DISTINCT T.title id, T.orimarv title, T.start vear FROM Titles AS T JOIN Had\_role AS H ON T.title\_id=H.title\_id WHERE H.name\_id IN (SELECT N. name id FROM

Names\_ AS N WHERE N.name\_ LIKE 'Alia Bhatt') AND T.title\_type LIKE 'movie')

INTERSECT

(SELECT DISTINCT T.title id, T.orimarv title, T.start vear FROM Titles AS T JOIN Had\_role AS H ON T.title\_id=H.title\_id WHERE H. name\_id IN (SELECT N. name id FROM

Names\_ AS N WHERE N.name\_ LIKE 'Ranbir Kapoor') AND T. title\_type LIKE 'movie'));

-- Query 4

-- Movies Starring Alia Bhatt BUT NOT Shah Rukh Khan

(SELECT DISTINCT T.title\_id, T.primary\_title, T.start\_year FROM Titles AS T JOIN Had\_role AS H ON T.title\_id=H.title\_id WHERE H.name\_id IN (SELECT N.name\_id FROM

Names\_ AS N WHERE N.name\_ LIKE 'Alia Bhatt') AND T.title\_type LIKE 'movie')

EXCEPT

(SELECT DISTINCT T.title\_id, T.primary\_title, T.start\_year FROM Titles AS T JOIN Had\_role AS H ON T.title\_id=H.title\_id WHERE H.name\_id IN (SELECT N. name\_id FROM

Names\_ AS N WHERE N. name\_ LIKE 'Shah Rukh Khan') AND T.title\_type LIKE 'movie');

-- Function

-- To classify recency of titles

DELIMITER $$

CREATE FUNCTION title\_recency (title\_age INT)

RETURNS VARCHAR(50)

DETERMINISTIC

BEGIN

DECLARE recency VARCHAR;

IF title\_age > 2022 THEN:

SET recency = 'Not Released';

ELSE IF (title\_age > 2021) THEN:

SET recency = Latest

ELSE IF (title\_age >2015) THEN:

SET recency = 'NEW':

ELSE IF (title\_age >2000) THEN:

SET recency = 'Old';

ELSE IF tItle\_age>1990 THEN:

SET recency = 'very old';

ELSE IF title\_age <1990 THEN:

SET recency = 'Ancient'

END IF ;

RETURN (recencv);

END: $$

DELIMITER H

SELECT Titles.title\_id, Titles.primary\_title, Titles.start\_year, title\_recency(Titles.start\_year) FROM Titles ORDER BY Titles.primary\_title DESC;

-- Procedure

-- To return the number of movies directed by a director

DELIMITER $$

CREATE PROCEDURE display\_count\_movies(IN id\_director VARCHAR(50), OUT num INT)

BEGIN

SELECT COUNT(title\_id) INTO num FROM Directors WHERE name\_id = id\_director;

END $$

DELIMITER ;

CALL display\_count\_movies((SELECT name\_id FROM Names WHERE name\_ LIKE 'Rajkumar Hirani'), @num);

SELECT @num;

-- Trigger

-- To prevent name from being deleted if it is birth\_year is greater than 2022

BEGIN

DECLARE error\_msg VARCHAR(255);

SET error\_msg = ('Birth Year Cannot be later than 2022');

IF new.birth\_year > 2022 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = error\_msg;

END IF;

END

--Cursor

DELIMITER $$

CREATE OR REPLACE PROCEDURE get\_time\_span (IN id\_title VARCHAR(255), INOUT fullList varchar(4000))

BEGIN

DECLARE finished INTEGER DEFAULT 0;

DECLARE full varchar(100) DEFAULT "";

DECLARE curName

CURSOR FOR

SELECT concat(start\_year , '-' , end\_year) FROM Titles WHERE Titles.title\_id=id\_title;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET finished = 1;

OPEN curName;

getName: LOOP

FETCH curName INTO full;

IF finished = 1 THEN LEAVE getName;

END IF;

SET fullList = CONCAT(full," ",fullList);

END LOOP getName;

CLOSE curName;

END$$

DELIMITER ;

SET @fullList = "";

CALL get\_time\_span('tt0039123', @fullList);

SELECT @fullList;

--Other Miscellaneous Queries

-- Query 1

-- How many different title\_types are there? How many of each?

CREATE OR REPLACE VIEW Q1(title\_type,Count)

AS SELECT T.title\_type, COUNT(\*)

FROM Titles AS T

GROUP BY T.title\_type

ORDER BY T.title\_type ASC;

SELECT \* FROM Q1;

-- Query 2

-- How many different professions are there? What are they?

CREATE OR REPLACE VIEW Q2(Job, Count)

AS SELECT P.job\_category, COUNT(\*)

FROM Principals AS P

GROUP BY P.job\_category

ORDER BY P.job\_category ASC;

SELECT \* FROM Q2;

-- Query 3

-- What genres are there? How many movies are there in each genre?

CREATE OR REPLACE VIEW Q3(Genre,Count)

AS SELECT G.genre, COUNT(G.genre) AS Count

FROM Title\_genres AS G, Titles AS T

WHERE T.title\_id = G.title\_id

AND T.title\_type = 'movie'

GROUP BY genre

ORDER BY Count DESC;

SELECT \* FROM Q3;

-- Query 4

-- List movies (runtime\_minutes, title\_type, primary\_title) which are

-- longer than 10 hours. Place them in descending ordering of runtime\_minutes.

CREATE OR REPLACE VIEW Q4(runtime\_minutes, title\_type, primary\_title) AS

SELECT runtime\_minutes, title\_type, primary\_title

FROM Titles WHERE runtime\_minutes > (10\*60)

ORDER BY runtime\_minutes DESC, title\_type ASC;

SELECT \* FROM Q4 LIMIT 10;

-- Query 5

-- How many actors are there in the database?

CREATE OR REPLACE VIEW Q5(Number\_of\_actors)

AS SELECT COUNT(DISTINCT N.name\_id) AS Number\_of\_actors

FROM Name\_worked\_as AS N

WHERE N.profession IN ('actor','actress');

SELECT \* FROM Q5;

-- Query 6

-- How many movies are there in the database?

CREATE OR REPLACE VIEW Q6(Number\_of\_movies)

AS SELECT COUNT(DISTINCT T.title\_id) AS Number\_of\_movies

FROM Titles AS T

WHERE T.title\_type IN ('movie','video');

SELECT \* FROM Q6;

-- Query 7

-- What time period does the database cover?

CREATE OR REPLACE VIEW Q7(Earliest,Latest)

AS SELECT LEAST(MIN(T.start\_year), MIN(T.end\_year)) AS Earliest,

GREATEST(MAX(T.start\_year), MAX(T.end\_year)) AS Latest

FROM Titles AS T;

SELECT \* FROM Q7;

-- Query 8

-- How many movies where made each year over the past 30 years? (Up to and

-- including 2019)

CREATE OR REPLACE VIEW Q8(Year, Number\_of\_movies)

AS SELECT T.start\_year, COUNT(\*) AS Number\_of\_movies

FROM Titles AS T

WHERE T.title\_type IN ('movie','video')

GROUP BY T.start\_year

HAVING T.start\_year BETWEEN 1990 AND 2019

ORDER BY T.start\_year ASC;

SELECT \* FROM Q8;

-- Query 9

-- Who are the actors who played James Bond in a movie? How many times did they

-- play the role of James Bond?

CREATE OR REPLACE VIEW Q9(name\_id,name\_,number\_of\_films)

AS SELECT N.name\_id, N.name\_, COUNT(\*) AS number\_of\_films

FROM Names\_ AS N, Had\_role AS H, Titles AS T

WHERE H.role\_ LIKE 'James Bond'

AND T.title\_type LIKE 'movie'

AND T.title\_id = H.title\_id

AND N.name\_id = H.name\_id

GROUP BY N.name\_id;

SELECT \* FROM Q9;

-- Query 10

-- How many actors played James Bond?

CREATE OR REPLACE VIEW Q10(number\_of\_JB\_actors)

AS SELECT COUNT(DISTINCT name\_id) AS number\_of\_JB\_actors

FROM Q9;

SELECT \* FROM Q10;

-- Query 11

-- I don't recognise some of these names lets look at them more closely

CREATE OR REPLACE VIEW Q11(name\_,title\_id,primary\_title,start\_year)

AS SELECT Q9.name\_, T.title\_id, T.primary\_title, T.start\_year

FROM Q9, Titles AS T, Had\_role AS H

WHERE Q9.name\_id = H.name\_id

AND H.role\_ LIKE 'James Bond'

AND T.title\_id = H.title\_id

AND T.title\_type LIKE 'movie'

ORDER BY T.start\_year DESC;

SELECT \* FROM Q11;

-- Query 12

-- Find all the movies made by Don "The Dragon" Wilson, the former light heavy

-- weight kickboxing champion. He was born in 1954 and is famous for the

-- Bloodfist series. Omit entries where he appears as himself. Output the start

-- year, the title type, title and the role he played. Order these by year.

CREATE OR REPLACE VIEW Q12(start\_year, title\_type, primary\_title, role\_)

AS SELECT DISTINCT T.start\_year, T.title\_type, T.primary\_title, H.role\_

FROM Titles AS T, Had\_role AS H

WHERE T.title\_id = H.title\_id

AND H.role\_ <> 'Himself'

AND T.title\_type IN ('movie','video')

AND H.name\_id = (

SELECT N.name\_id

FROM Names\_ AS N

WHERE N.name\_ LIKE 'Don Wilson' AND N.birth\_year = 1954)

ORDER BY T.start\_year ASC;

SELECT \* FROM Q12;

-- Query 13

-- Did he ever play a role multiple times ?

CREATE OR REPLACE VIEW Q13(role\_,Count)

AS SELECT Q12.role\_, COUNT(\*) AS Count

FROM Q12

GROUP BY Q12.role\_

HAVING Count >=2;

SELECT \* FROM Q13;

-- Query 14

-- What movies were these ?

CREATE OR REPLACE VIEW Q14(primary\_title,role\_)

AS SELECT Q12.primary\_title, Q12.role\_

FROM Q12, Q13

WHERE Q12.role\_ = Q13.role\_;

SELECT \* FROM Q14;

-- Query 15

-- What are the top 250 movies as determined by the average rating with the over

-- 100,000 votes?

CREATE OR REPLACE VIEW Q15(title\_id,primary\_title,average\_rating)

AS SELECT T.title\_id, T.primary\_title, R.average\_rating

FROM Titles AS T, Title\_ratings AS R

WHERE T.title\_id = R.title\_id

AND T.title\_type = 'movie'

AND R.num\_votes > 100000

ORDER BY R.average\_rating DESC

LIMIT 250;

SELECT \* FROM Q15 LIMIT 15;

-- Query 16

-- Who are the top 10 actors who have made the most movies listed in the top

-- 250 movies (determined as in Q15) and how many?

CREATE OR REPLACE VIEW Q16(name\_id,name\_,Count)

AS SELECT H.name\_id, N.name\_, COUNT(\*) AS Count

FROM Q15, Titles AS T, Names\_ AS N, Had\_role AS H

WHERE Q15.title\_id = T.title\_id

AND T.title\_id = H.title\_id

AND N.name\_id = H.name\_id

GROUP BY H.name\_id

ORDER BY Count DESC

LIMIT 10;

SELECT \* FROM Q16;

-- Query 17

-- List all actor names and their roles who starred in the movie Back to the

-- future

CREATE OR REPLACE VIEW Q17(name\_,role\_)

AS SELECT N.name\_, H.role\_

FROM Titles AS T, Had\_role AS H, Names\_ AS N

WHERE T.primary\_title LIKE 'Back to the Future'

AND T.title\_type LIKE 'movie'

AND T.title\_id = H.title\_id

AND H.name\_id = N.name\_id;

SELECT \* FROM Q17;

-- Query 18

-- What are the average ratings of the entire back to the future series?

CREATE OR REPLACE VIEW Q18(primary\_title,average\_rating)

AS SELECT T.primary\_title, R.average\_rating

FROM Titles AS T, Title\_ratings AS R

WHERE T.primary\_title REGEXP '^Back to the Future.\*'

AND T.title\_id = R.title\_id

AND T.title\_type = 'movie';

SELECT \* FROM Q18;

-- Query 19

-- What are the average ratings of the entire Trancers series?

CREATE OR REPLACE VIEW Q19(primary\_title,average\_rating)

AS SELECT T.primary\_title, R.average\_rating

FROM Titles AS T, Title\_ratings AS R

WHERE T.primary\_title REGEXP '^Trancers.\*'

AND T.title\_id = R.title\_id

AND T.title\_type IN ('movie','video');

SELECT \* FROM Q19;

-- Query 20

-- How many horror movies are made in leap years>? (~start\_year divisible by 4)

CREATE OR REPLACE VIEW Q20(start\_year,Number\_of\_horror\_movies)

AS SELECT T.start\_year, COUNT(DISTINCT T.title\_id) AS Number\_of\_horror\_movies

FROM Titles AS T, Title\_genres AS G

WHERE T.title\_id = G.title\_id

AND G.genre = 'Horror'

AND T.title\_type IN ('movie','video')

AND (T.start\_year % 4) = 0

GROUP BY T.start\_year

ORDER BY T.start\_year DESC;

SELECT \* FROM Q20;

-- Query 21

-- What were the episodes of Fawlty Towers?

CREATE OR REPLACE VIEW Q21(season\_number,episode\_number,primary\_title)

AS SELECT E.season\_number, E.episode\_number, T2.primary\_title

FROM Titles AS T1, Titles AS T2, Episode\_belongs\_to AS E

WHERE T1.primary\_title = 'Fawlty Towers'

AND T1.title\_type = 'tvSeries'

AND T1.title\_id = E.parent\_tv\_show\_title\_id

AND T2.title\_type = 'tvEpisode'

AND T2.title\_id = E.episode\_title\_id

ORDER BY E.season\_number, E.episode\_number;

SELECT \* FROM Q21;

-- Query 22

-- What were the names and average ratings of each episode of The X-Files?

CREATE OR REPLACE VIEW Q22(season\_number,episode\_number,primary\_title,average\_rating)

AS SELECT E.season\_number, E.episode\_number, T2.primary\_title, R.average\_rating

FROM Titles AS T1, Titles AS T2, Episode\_belongs\_to AS E, Title\_ratings AS R

WHERE T1.primary\_title = 'The X-Files'

AND T1.title\_type = 'tvSeries'

AND T1.title\_id = E.parent\_tv\_show\_title\_id

AND T2.title\_type = 'tvEpisode'

AND T2.title\_id = E.episode\_title\_id

AND T2.title\_id = R.title\_id

ORDER BY E.season\_number, E.episode\_number;

SELECT \* FROM Q22;

-- Query 23

-- What was the most popular episode of The X-Files?

CREATE OR REPLACE VIEW Q23(season\_number,episode\_number,primary\_title,average\_rating)

AS SELECT Q22.season\_number, Q22.episode\_number, Q22.primary\_title, Q22.average\_rating

FROM Q22

WHERE Q22.average\_rating = (SELECT MAX(Q22.average\_rating) FROM Q22);

SELECT \* FROM Q23;

-- Query Q24

-- How many episodes were there in The X-Files per season? And what was the average of the average episode ratings ?

CREATE OR REPLACE VIEW Q24(season\_number,Number\_of\_episodes,Average\_of\_ep\_average\_ratings)

AS SELECT Q22.season\_number, COUNT(\*) AS Number\_of\_episodes, AVG(Q22.average\_rating) AS Average\_of\_ep\_average\_ratings

FROM Q22

GROUP BY Q22.season\_number

ORDER BY Q22.season\_number;

SELECT \* FROM Q24;