==

I certify that:

- **This paper/project/exam is entirely my own work.**

- **I have not quoted the words of any other person from a printed source or a website without indicating what has been quoted and providing an appropriate citation.**

- **I have not submitted this paper / project to satisfy the requirements of any other course.**

Signature____Jack Stillwell________

Date_______9 June 2018___________

_____

==

Notes for the Professor:

You mentioned you might like to run our code in SQL but expressly disallowed the submission of multiple files. Therefore, I created and saved SQL scripts which I have tested on Drexel's database at the following link so you can download and test them at your convenience rather than laboriously copy-pasting every statement. I hope this makes grading everything a little easier on you!

CreateAndInsertAllTables : https://drive.google.com/file/d/1RedSdGSGtxYiSfuPNJvRTjopFHqI-v4A/view?usp=sharing

AllQueries : https://drive.google.com/file/d/1Fh5t2HrDpwURuoW9Wrwb9-sxKOPPwHYG/view?usp=sharing

DeleteAllTables : https://drive.google.com/file/d/1nZE0dma4zsKJeoyQUvwj_TzRFPiV2HJH/view?usp=sharing

Additionally, I was unable to create arrows (they kept changing location upon save and open) so I named my relational schema fields PARENT_columnname. I hope this is clear and doesn't cause any readability issues.

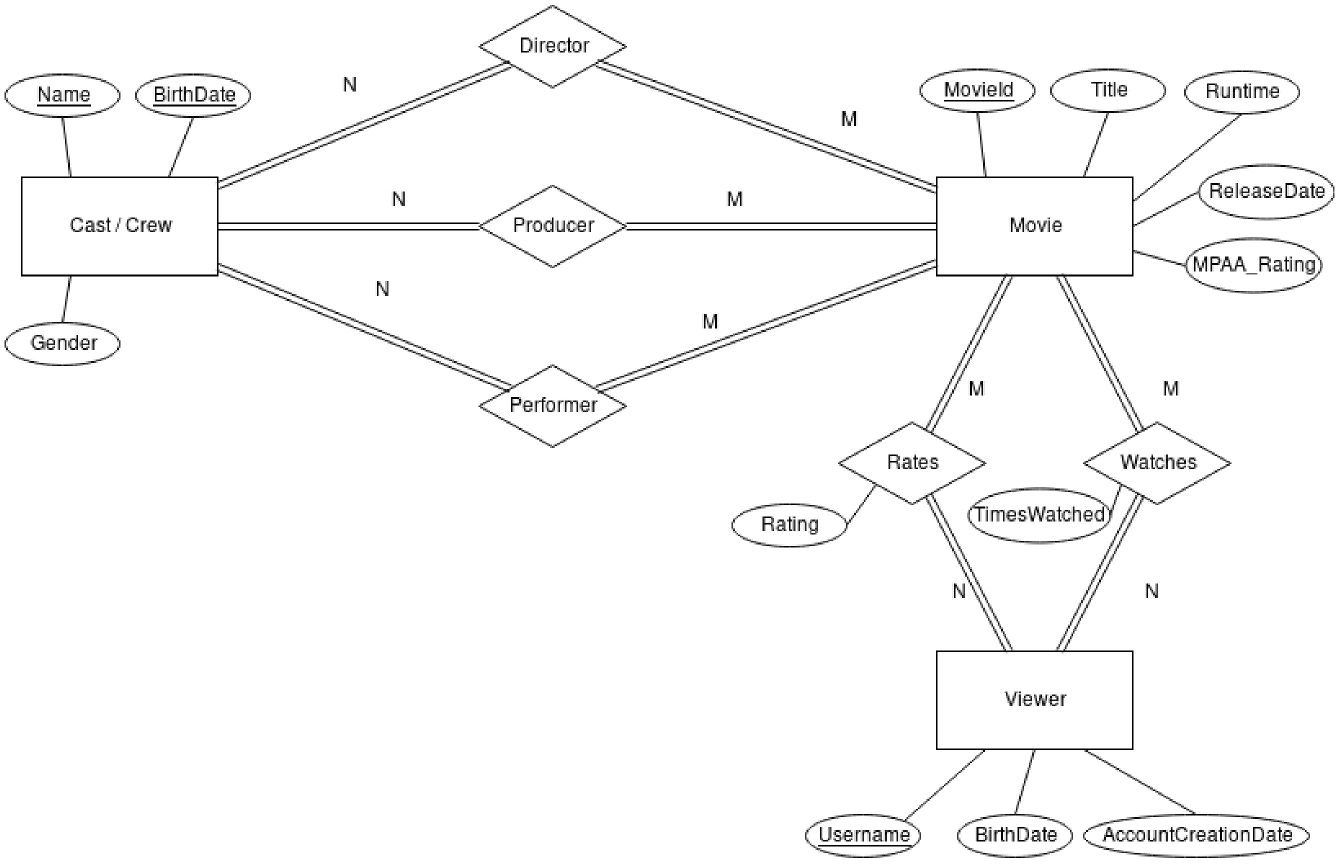
Table of Contents:

Requirements

My database will contain data associated with a private movie collection stored on a personal media server. The user of my database will be cataloging their collection of movies as well as data on the viewing habits of their viewers. All movies must have a title, running time, release date, and MPAA rating. Additionally, each movie will have a unique numeric identifier. The release date will be the movie's theatrical release date, if there was no theatrical release, the digital or film release date may be used. Any movies not rated by the MPAA or an equivalent agency according to MPAA standards will receive an "UNRATED" tag. Each movie will also have information about the cast and crew recorded, including the director(s), producer(s), and prominent performers. Cast and crew will have age and gender recorded. Viewers will be able to rate movies on a scale from 1 (very bad) to 5 (very good), and their watch histories will be tracked, including the amount of times they watch each movie.

The user of the database needs to be able to add movies and all related data (such as directors, producers, and performers), as well as create viewer accounts. The user also needs to search for data related to viewers, such as the most popular movie by views, all movies available to a viewer based on viewer age and movie rating, a viewers favorite performer (as determined by participation in movies with ratings 3 or higher), overall worst movie in the collection (as determined by least views with an average rating of 2 or below), and the favorite movie for an age group such as 10-20 year-olds (as determined by most views and an average rating of 3 or higher).

ER Diagram


Director
N
M
Name
BirthDate
MovieId
Title
Runtime
ReleaseDate
Cast / Crew
N
Producer
M
Movie
MPAA_Rating
N
M
Gender
Performer
M
M
Rates
Watches
Rating
TimesWatched
N
N
Viewer
Username
BirthDate
AccountCreationDate

Relational Schema

Movie(MovieId, Title, Release Date, Running Time, MPAARating)

CastAndCrew(Name, BirthDate, Gender)

Viewer(Username, AccountCreationDate, BirthDate)

Producer(CastAndCrew_Name, CastAndCrew_BirthDate, Movie_MovieId)

Director(CastAndCrew_Name, CastAndCrew_BirthDate, Movie_MovieId)

Performer(CastAndCrew_Name, CastAndCrew_BirthDate, Movie_MovieId)

Ratings(Viewer_Username, Movie_MovieId, Rating)

WatchHistory(Viewer_Username, Movie_MovieId, Times_Watched)

Translation Method Employed : Null-Sensitive

NOTE : The lack of 1 to 1 or 1 to N relationships made the difference between translation methods negligible.

Data Dictionaries

Movie: Contains information about a Motion Picture in a private collection

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
MovieId	A unique number identifying a unique movie in the collection	NUMBER(9)	All	N	Y	N
Title	The title of the movie	VARCHAR(50)	All	N	N	N
ReleaseDate	The theatrical release, if none then digital / film release.	DATE	1/1/1900 < x < now	N	N	N
RunningTime	The time in minutes of the movie.	NUMBER(3)	All	N	N	N
MPAARating	The rating assigned by the MPAA, “UNRATED” if none.	VARCHAR(7)	“G” , “PG”, “PG-13” , “R” , “NC-17” , “UNRATED”	N	N	N

CastAndCrew: Contains information about the cast and crew of movies in a collection

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
Name	The name of the cast / crew member.	VARCHAR(50)	All	N	Y	N
BirthDate	The birthday of the cast / crew member	DATE	All	N	Y	N
Gender	The gender of the cast / crew member	CHAR(1)	"M" , "F" , "X"	N	N	N

Viewers: Contains information about the viewers of the movie collection

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
Username	The unique username of the viewer.	VARCHAR(20)	All	N	Y	N
AccountCreationDate	The date the viewer's account was created.	DATE	All	N	N	N
BirthDate	The reported birthday of the viewer.	DATE	> 1 / 1/ 1900	N	N	N

Producer: Contains information about the Producer(s) for movies in the collection.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
Name	The name of the cast / crew member.	VARCHAR(50)	All	N	Y	Y
BirthDate	The birthday of the cast / crew member	DATE	All	N	Y	Y
MovieId	A unique number identifying a unique movie in the collection	NUMBER(9)	Al	N	Y	Y

Director: Contains information about the Producer(s) for movies in the collection.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
Name	The name of the cast / crew member.	VARCHAR(50)	All	N	Y	Y
BirthDate	The birthday of the cast / crew member	DATE	All	N	Y	Y
MovieId	A unique number identifying a unique movie in the collection	NUMBER(9)	0-999999999	N	Y	Y

Performer: Contains information about the Producer(s) for movies in the collection.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
Name	The name of the cast / crew member.	VARCHAR(50)	All	N	Y	Y
BirthDate	The birthday of the cast / crew member	DATE	All	N	Y	Y
MovieId	A unique number identifying a unique movie in the collection	NUMBER(9)	All	N	Y	Y

Ratings: Contains information about a viewer's ratings for movies in the collection.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
Username	The name of the cast / crew member.	VARCHAR20)	All	N	Y	Y
MovieId	A unique number identifying a unique movie in the collection	NUMBER(9)	All	N	Y	Y
Rating	A number from 1 to 5 indicating the viewer's opinion of a movie, 1 being horrible	NUMBER(1)	1-5	N	N	N

	and 5 being excellent.					

WatchHistory: Contains information about a viewer's viewings of movies in the collection.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
Username	The name of the cast / crew member.	VARCHAR(20)	All	N	Y	Y
MovieId	A unique number identifying a unique move in the collection	NUMBER(9)	0-999999999	N	Y	Y
TimesWatched	The number of times a viewer has watched a particular movie.	NUMBER(3)	0-999	N	N	N

DDL

```
CREATE TABLE Movie
(
   MovieId NUMBER(9)
      CONSTRAINT movie_pk PRIMARY KEY,
   Title VARCHAR(50)
      CONSTRAINT movie_nn_title NOT NULL,
   ReleaseDate DATE
      CONSTRAINT movie_nn_releasedate NOT NULL
      CONSTRAINT movie_valid_releasedate CHECK (ReleaseDate > TO_DATE('1900/01/01',
'yyyy/mm/dd','NLS_DATE_LANGUAGE = American')),
   RunningTime NUMBER(3)      CONSTRAINT movie_nn_runningtime NOT NULL,
   MPAARating VARCHAR(7)
      CONSTRAINT movie_nn_movierating NOT NULL
      CONSTRAINT movie_valid_movierating CHECK (MPAARating IN ('G','PG','PG-13','R','NC-
17','UNRATED'))
);

CREATE TABLE CastAndCrew
(
   Name VARCHAR(50),
   BirthDate DATE,
   Gender CHAR(1)
      CONSTRAINT castandcrew_nn_gender NOT NULL
      CONSTRAINT castandcrew_valid_gender CHECK ( Gender IN ( 'M' , 'F' , 'X' ) ),
   CONSTRAINT castandcrew_pk PRIMARY KEY ( Name , BirthDate )
);

CREATE TABLE Viewers
(
   Username VARCHAR(20)
      CONSTRAINT viewers_pk PRIMARY KEY,
   BirthDate DATE
```

```
        CONSTRAINT viewers_nn_birthdate NOT NULL
        CONSTRAINT viewers_valid_birthdate CHECK (BirthDate > TO_DATE('1900/01/01',
'yyyy/mm/dd','NLS_DATE_LANGUAGE = American')),
     AccountCreationDate DATE DEFAULT SYSDATE
);

CREATE TABLE Producer
(
    Name VARCHAR(50),
    BirthDate DATE,
    MovieId NUMBER(9)
        CONSTRAINT producer_fk_movieid REFERENCES Movie(MovieId) ON DELETE
CASCADE,
    CONSTRAINT producer_fk_castandcrew FOREIGN KEY (Name , BirthDate) REFERENCES
CastAndCrew( Name , BirthDate ) ON DELETE CASCADE,
    CONSTRAINT producer_pk PRIMARY KEY (Name, BirthDate, MovieId)
);

CREATE TABLE Director
(
    Name VARCHAR(50),
    BirthDate DATE,
    MovieId NUMBER(9)
        CONSTRAINT director_fk_movieid REFERENCES Movie(MovieId) ON DELETE CASCADE,
    CONSTRAINT director_fk_castandcrew FOREIGN KEY (Name , BirthDate) REFERENCES
CastAndCrew( Name , BirthDate ) ON DELETE CASCADE,
    CONSTRAINT director_pk PRIMARY KEY (Name, BirthDate, MovieId)
);

CREATE TABLE Performer
(
    Name VARCHAR(50),
    BirthDate DATE,
    MovieId NUMBER(9)
```

```
        CONSTRAINT performer_fk_movieid REFERENCES Movie(MovieId) ON DELETE
CASCADE,
    CONSTRAINT performer_fk_castandcrew FOREIGN KEY (Name , BirthDate) REFERENCES
CastAndCrew( Name , BirthDate ) ON DELETE CASCADE,
    CONSTRAINT performer_pk PRIMARY KEY (Name, BirthDate, MovieId)
);

CREATE TABLE Ratings
(
    Username VARCHAR(20)
        CONSTRAINT ratings_fk_username REFERENCES Viewers(Username) ON DELETE
CASCADE,
    MovieId NUMBER(9)
        CONSTRAINT ratings_fk_movieid REFERENCES Movie(MovieId) ON DELETE CASCADE,
    Rating NUMBER(1)
        CONSTRAINT ratings_nn_rating NOT NULL
        CONSTRAINT ratings_valid_rating CHECK ( Rating BETWEEN 1 AND 5 ),
    CONSTRAINT ratings_pk PRIMARY KEY (Username , MovieId)
);

CREATE TABLE WatchHistory
(
    Username VARCHAR(20)
        CONSTRAINT watchhistory_fk_username REFERENCES Viewers(Username) ON DELETE
CASCADE,
    MovieId NUMBER(9)
        CONSTRAINT watchhistory_fk_movieid REFERENCES Movie(MovieId) ON DELETE
CASCADE,
    TimesWatched NUMBER(3)
        CONSTRAINT watchhistory_nn_timeswatched NOT NULL
        CONSTRAINT watchhistory_valid_timeswatch CHECK ( TimesWatched BETWEEN 0 AND
999 ),
    CONSTRAINT watchhistory_pk PRIMARY KEY (Username , MovieId)
);
```

DML

Movie Inserts

```
INSERT INTO Movie (MovieId, Title, ReleaseDate, RunningTime, MPAARating) VALUES
(000000001, 'About Time', TO_DATE('2013/11/08', 'yyyy/mm/dd','NLS_DATE_LANGUAGE =
American'), 123, 'R');

INSERT INTO Movie (MovieId, Title, ReleaseDate, RunningTime, MPAARating) VALUES
(000000002, 'Spirit: Stallion of the Cimmaron', TO_DATE('2002/05/24',
'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 83, 'G');

INSERT INTO Movie (MovieId, Title, ReleaseDate, RunningTime, MPAARating) VALUES
(000000003, 'Arrival', TO_DATE('2016/11/11', 'yyyy/mm/dd','NLS_DATE_LANGUAGE =
American'), 116, 'PG-13');
```

CastAndCrew Inserts

```
INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('Richard Curtis',
TO_DATE('1956/11/08', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'M');

INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('Domhnall Gleeson',
TO_DATE('1983/05/12', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'M');

INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('Tim Bevan',
TO_DATE('1957/12/20', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'M');

INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('Kelly Asbury',
TO_DATE('1960/01/15', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'M');

INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('Matt Damon',
TO_DATE('1970/10/08', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'M');

INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('Jeffrey Katzenberg',
TO_DATE('1950/12/21', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'M');
```

INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('Denis Villeneuve', TO_DATE('1967/10/03', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'M');

INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('Amy Adams', TO_DATE('1974/08/20', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'F');

INSERT INTO CastAndCrew (Name, BirthDate, Gender) VALUES ('David Linde', TO_DATE('1960/02/08', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 'M');

Viewers Inserts

INSERT INTO Viewers (Username, BirthDate) VALUES ('Jack', TO_DATE('1997/06/20', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'));

INSERT INTO Viewers (Username, BirthDate) VALUES ('Parent', TO_DATE('1953/10/15', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'));

INSERT INTO Viewers (Username, BirthDate) VALUES ('YoungSibling', TO_DATE('2001/06/18', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'));

Producer Inserts

INSERT INTO Producer (Name, BirthDate, MovieId) VALUES ('Tim Bevan', TO_DATE('1957/12/20', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000001);

INSERT INTO Producer (Name, BirthDate, MovieId) VALUES ('Jeffrey Katzenberg', TO_DATE('1950/12/21', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000002);

INSERT INTO Producer (Name, BirthDate, MovieId) VALUES ('David Linde', TO_DATE('1960/02/08', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000003);

Director Inserts

INSERT INTO Director (Name, BirthDate, MovieId) VALUES ('Richard Curtis', TO_DATE('1956/11/08', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000001);

INSERT INTO Director (Name, BirthDate, MovieId) VALUES ('Kelly Asbury', TO_DATE('1960/01/15', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000002);

INSERT INTO Director (Name, BirthDate, MovieId) VALUES ('Denis Villeneuve', TO_DATE('1967/10/03', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000003);

Performer Inserts

INSERT INTO Performer (Name, BirthDate, MovieId) VALUES ('Domhnall Gleeson', TO_DATE('1983/05/12', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000001);

INSERT INTO Performer (Name, BirthDate, MovieId) VALUES ('Matt Damon', TO_DATE('1970/10/08', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000002);

INSERT INTO Performer (Name, BirthDate, MovieId) VALUES ('Amy Adams', TO_DATE('1974/08/20', 'yyyy/mm/dd','NLS_DATE_LANGUAGE = American'), 000000003);

Ratings Inserts

INSERT INTO Ratings (Username, MovieId, Rating) VALUES ('Jack', 000000001, 4);

INSERT INTO Ratings (Username, MovieId, Rating) VALUES ('Parent', 000000001, 5);

INSERT INTO Ratings (Username, MovieId, Rating) VALUES ('YoungSibling', 000000001, 2);

INSERT INTO Ratings (Username, Movieid, Rating) VALUES ('YoungSibling', 2, 3);

INSERT INTO Ratings (Username, Movieid, Rating) VALUES ('Jack', 2, 3);

INSERT INTO Ratings (Username, MovieId, Rating) VALUES ('Jack', 3, 1);

INSERT INTO Ratings (Username, MovieId, Rating) VALUES ('Parent', 3, 1);

INSERT INTO Ratings (Username, MovieId, Rating) VALUES ('YoungSibling', 3, 1);

WatchHistory Inserts

INSERT INTO WatchHistory (Username, MovieId, TimesWatched) VALUES ('Jack', 000000001, 2);

INSERT INTO WatchHistory (Username, MovieId, TimesWatched) VALUES ('Parent', 000000001, 4);

INSERT INTO WatchHistory (Username, MovieId, TimesWatched) VALUES ('YoungSibling', 000000001, 1);
INSERT INTO WatchHistory (Username, MovieId, TimesWatched) VALUES ('Jack', 3, 1);

INSERT INTO WatchHistory (Username, MovieId, TimesWatched) VALUES ('Parent', 3, 1);

INSERT INTO WatchHistory (Username, MovieId, TimesWatched) VALUES ('YoungSibling', 3, 1);

Update Rating of Movie for Viewer

UPDATE Ratings SET Rating=3 WHERE Username='Parent' AND MovieId=1;

Update WatchHistory of Viewer for Movie

UPDATE WatchHistory SET TimesWatched=TimesWatched+1 WHERE Username='Jack' and MovieId=1;

Delete a Movie

DELETE FROM Movie WHERE Title = 'About Time'

Delete a Viewer

DELETE FROM Viewers WHERE Username='Parent'

Delete a CastAndCrew member

DELETE FROM CastAndCrew WHERE Name='Amy Adams'

Queries

Most Viewed Movie

```
SELECT movie.title , t.totalviews
FROM

(SELECT
movieid, SUM(timeswatched) as TotalViews
FROM watchhistory
GROUP BY movieid
HAVING SUM(timeswatched)= (SELECT MAX(SUM(timeswatched)) FROM watchhistory GROUP
BY movieid)) t

JOIN movie ON t.movieid=movie.movieid;
```

Movies Available based on Viewer Age and MPAA Rating

```
SELECT * FROM (SELECT v.username AS Username, m.title AS MovieTitle
FROM Viewers v, Movie m
WHERE (months_between(SYSDATE, v.birthdate)/12 < 18
and m.mpaarating = 'G' or m.mpaarating = 'PG' or m.mpaarating = 'PG-13') or
(months_between(SYSDATE, v.birthdate)/12 >= 18))
GROUP BY Username, MovieTitle
```

Viewer’s Favorite Performer

```
SELECT viewname, performer, MAX(popularity) FROM (SELECT Username AS viewname,
    pName AS performer,
    COUNT(pName) AS popularity
    FROM (SELECT * FROM (SELECT p.Name AS pName, t.Username AS Username
   FROM performer p
```

```
    JOIN (SELECT * FROM (SELECT movieid AS MovieId, username AS Username, rating AS
Rating FROM Ratings WHERE rating >= 3) GROUP BY Username, MovieId, Rating) t
    ON p.movieid = t.movieid) GROUP BY Username, pName)
    GROUP BY username, pname
)
GROUP BY viewname, performer
```

BREAKDOWN:

```
/* Get all Movies a User has rated 3 or higher */
(Select * from (Select movieid as MovieId, username as Username, rating as Rating
from Ratings
where rating >= 3) group by Username, MovieId, Rating)

/* Associate Users with Performers in Movies rated 3 or higher */
(select * from (Select p.Name as pName,
        t.Username as Username
    from performer p
    join (Select * from (Select movieid as MovieId, username as Username, rating as Rating
from Ratings
where rating >= 3) group by Username, MovieId, Rating) t
    on p.movieid = t.movieid) group by Username, pName)



/* count the number of times performers appear */
Select Username as viewname,
     pName as performer,
     count(pName) as popularity
     from (select * from (Select p.Name as pName,
        t.Username as Username
    from performer p
    join (Select * from (Select movieid as MovieId, username as Username, rating as Rating
from Ratings
where rating >= 3) group by Username, MovieId, Rating) t
```

```
    on p.movieid = t.movieid) group by Username, pName)
    group by username, pname

/* take the max value of the times performers appear for each user */

select viewname, performer, MAX(popularity) from (

Select Username as viewname,
     pName as performer,
     count(pName) as popularity
     from (select * from (Select p.Name as pName,
         t.Username as Username
    from performer p
    join (Select * from (Select movieid as MovieId, username as Username, rating as Rating
from Ratings
where rating >= 3) group by Username, MovieId, Rating) t
    on p.movieid = t.movieid) group by Username, pName)
    group by username, pname

     )
group by viewname, performer
```

Overall Worst Movie

```
SELECT m.title AS Title FROM movie m,

(SELECT movieid, MIN(viewcount)
FROM
(
   SELECT m.movieid AS movieid, t.viewcount AS viewcount FROM
      (
      /* all movies avg rating < 3 */
```

```
        SELECT * FROM (SELECT movieid, AVG(rating) as avgrate FROM ratings GROUP BY
movieid) WHERE avgrate < 3
        ) m
        JOIN
        (
        /* total movie views */
        SELECT movieid, COUNT(movieid) AS viewcount FROM watchhistory GROUP BY movieid
        ) t
        ON m.movieid = t.movieid
)
GROUP BY movieid) t WHERE m.movieId=t.movieid;
```

Favorite Movie for Age Group

```
SELECT m.title AS Title FROM movie m,

(SELECT movieid, MAX(viewcount) FROM

(SELECT movieId, COUNT(movieid) AS viewcount FROM

(SELECT * FROM (SELECT r.movieid AS MovieId, r.username AS Username, r.rating AS Rating
FROM Ratings r, (SELECT username FROM viewers WHERE (months_between(SYSDATE,
birthdate)/12 < 20) AND (months_between(SYSDATE, birthdate)/12 > 10)) t
WHERE r.username = t.username AND rating >= 3) GROUP BY Username, MovieId, Rating)

GROUP BY movieId)

GROUP BY movieid) t WHERE m.movieId=t.movieid
```

BREAKDOWN:

```
/* get all users in age group */
```

```
Select username from viewers where (months_between(SYSDATE, birthdate)/12 < 20) and
(months_between(SYSDATE, birthdate)/12 > 10)

/* get all ratings for viewers in age group */
(Select * from (Select r.movieid as MovieId, r.username as Username, r.rating as Rating
from Ratings r, (Select username from viewers where (months_between(SYSDATE, birthdate)/12 < 20)
and (months_between(SYSDATE, birthdate)/12 > 10)) t
where r.username = t.username and rating >= 3) group by Username, MovieId, Rating)

/* count the number of times a movie appears */
Select movieId, count(movieid) as viewcount from

(Select * from (Select r.movieid as MovieId, r.username as Username, r.rating as Rating
from Ratings r, (Select username from viewers where (months_between(SYSDATE, birthdate)/12 < 20)
and (months_between(SYSDATE, birthdate)/12 > 10)) t
where r.username = t.username and rating >= 3) group by Username, MovieId, Rating)

group by movieId

/* take the max count and return the movie */
select movieid, max(viewcount) from

(Select movieId, count(movieid) as viewcount from

(Select * from (Select r.movieid as MovieId, r.username as Username, r.rating as Rating
from Ratings r, (Select username from viewers where (months_between(SYSDATE, birthdate)/12 < 20)
and (months_between(SYSDATE, birthdate)/12 > 10)) t
where r.username = t.username and rating >= 3) group by Username, MovieId, Rating)

group by movieId)

group by movieid

/*return the movie info */
```

```
select m.title as Title from movie m,

(select movieid, max(viewcount) from

(Select movieId, count(movieid) as viewcount from

(Select * from (Select r.movieid as MovieId, r.username as Username, r.rating as Rating
from Ratings r, (Select username from viewers where (months_between(SYSDATE, birthdate)/12 < 20)
and (months_between(SYSDATE, birthdate)/12 > 10)) t
where r.username = t.username and rating >= 3) group by Username, MovieId, Rating)

group by movieId)

group by movieid) t where m.movieId=t.movieid
```