Student Name: Shreyas Padhye
Database Name: db11g

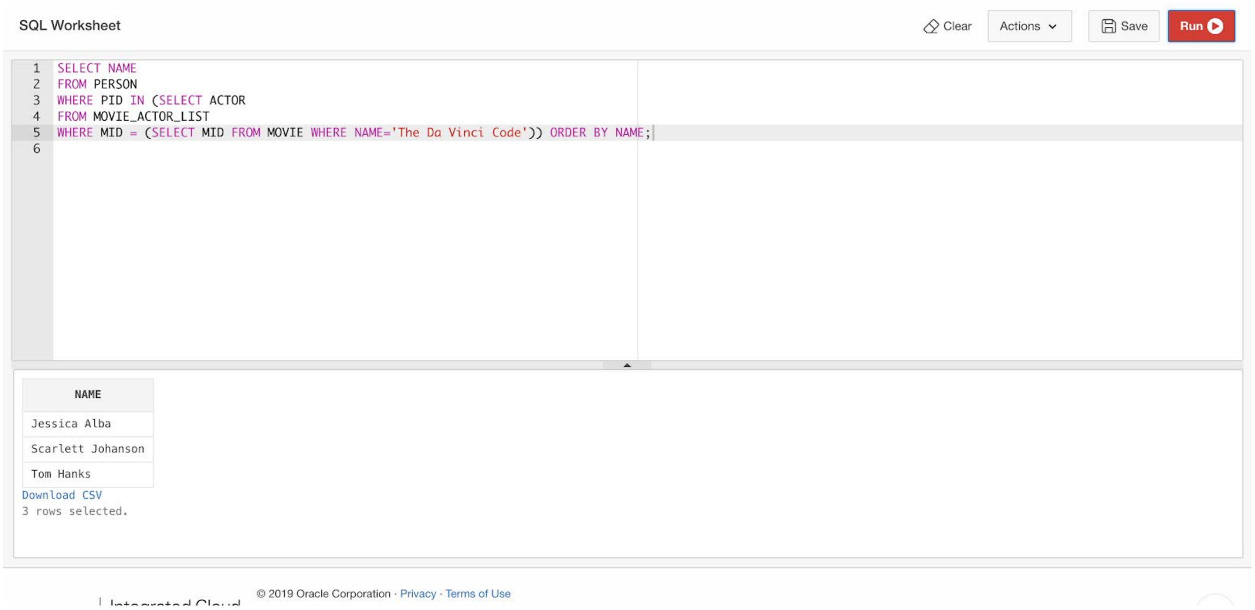
Tables:

1. MOVIE
2. MOVIE_ACTOR_LIST
3. MOVIE_GENRE_LIST
4. PERSON
5. IMDB_USER
6. REVIEWS
7. ROLES
8. FAVORITE_MOVIE_LIST
9. SHOW
10. FAVORITE_SHOW_LIST
11. TV_SHOW_TEAM
12. REVIEWED_MOVIE_LIST
13. TOWATCH_MOVIE_LIST
14. REVIEWED_SHOW_LIST
15. TOWATCH_SHOW_LIST
16. SEASONS
17. EPISODES
18. SCENES
19. AWARDS

Note:
Q2. num ratings - considered as the number of times the movie was rated
Q3. highest number of ratings - considered the movie that was rated highest number of times and still had least average rating

OUTPUTS:

Q1.

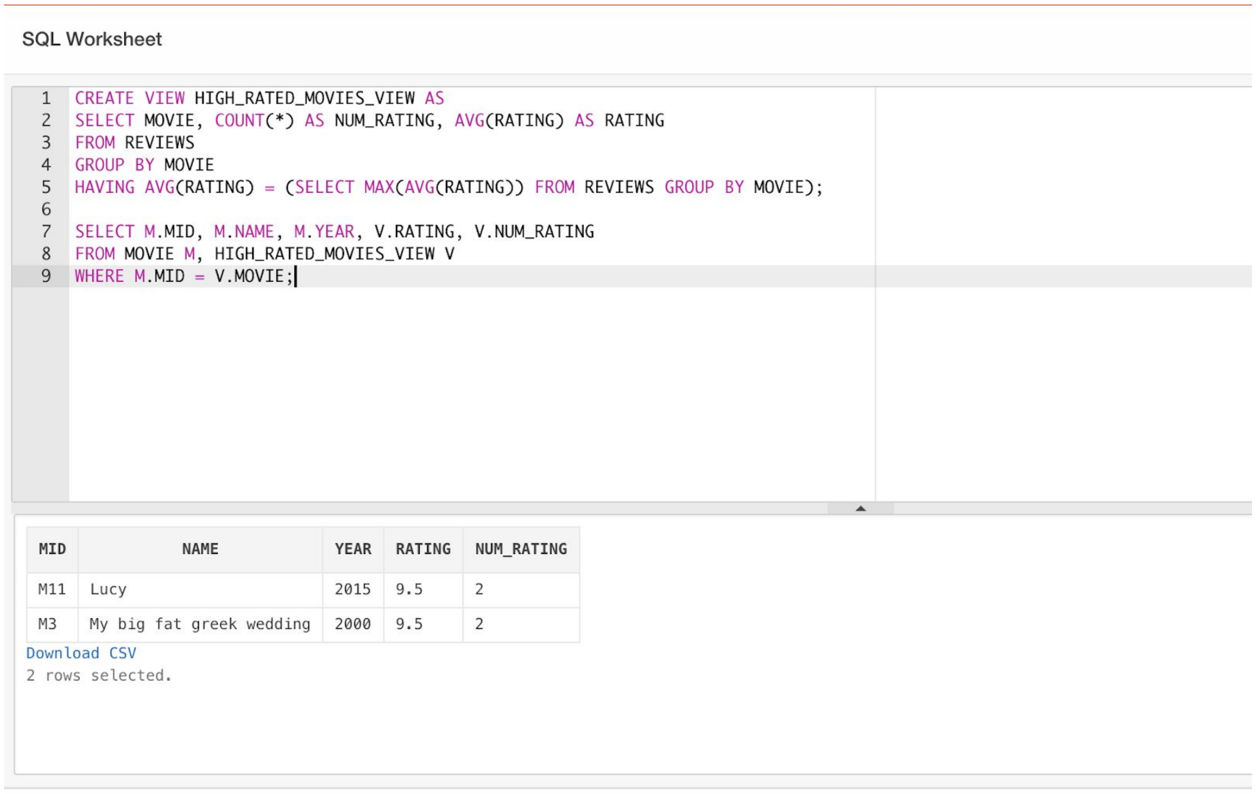


```sql
SELECT NAME
FROM PERSON
WHERE PID IN (SELECT ACTOR
FROM MOVIE_ACTOR_LIST
WHERE MID = (SELECT MID FROM MOVIE WHERE NAME='The Da Vinci Code')) ORDER BY NAME;
```

NAME
Jessica Alba
Scarlett Johanson
Tom Hanks

```
Download CSV
3 rows selected.
```

Q2.

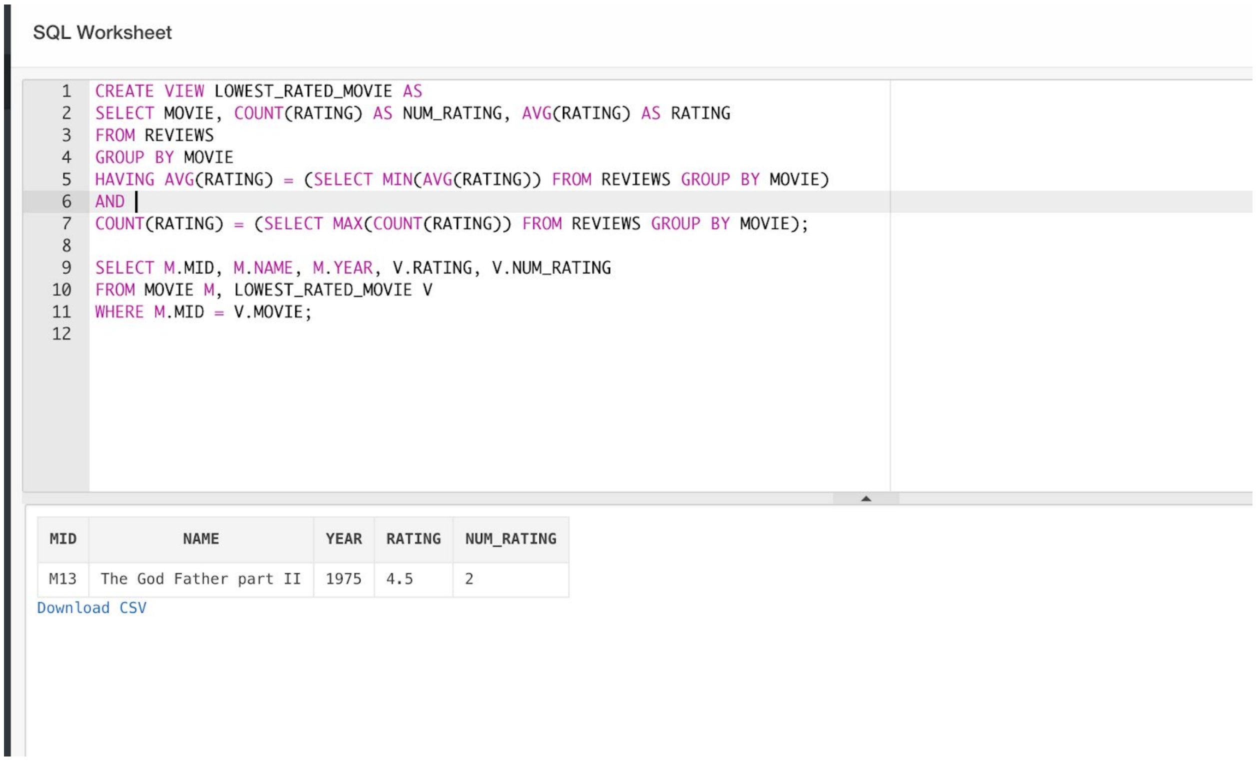


```sql
CREATE VIEW HIGH_RATED_MOVIES_VIEW AS
SELECT MOVIE, COUNT(*) AS NUM_RATING, AVG(RATING) AS RATING
FROM REVIEWS
GROUP BY MOVIE
HAVING AVG(RATING) = (SELECT MAX(AVG(RATING)) FROM REVIEWS GROUP BY MOVIE);

SELECT M.MID, M.NAME, M.YEAR, V.RATING, V.NUM_RATING
FROM MOVIE M, HIGH_RATED_MOVIES_VIEW V
WHERE M.MID = V.MOVIE;
```

MID	NAME	YEAR	RATING	NUM_RATING
M11	Lucy	2015	9.5	2
M3	My big fat greek wedding	2000	9.5	2

```
Download CSV
2 rows selected.
```

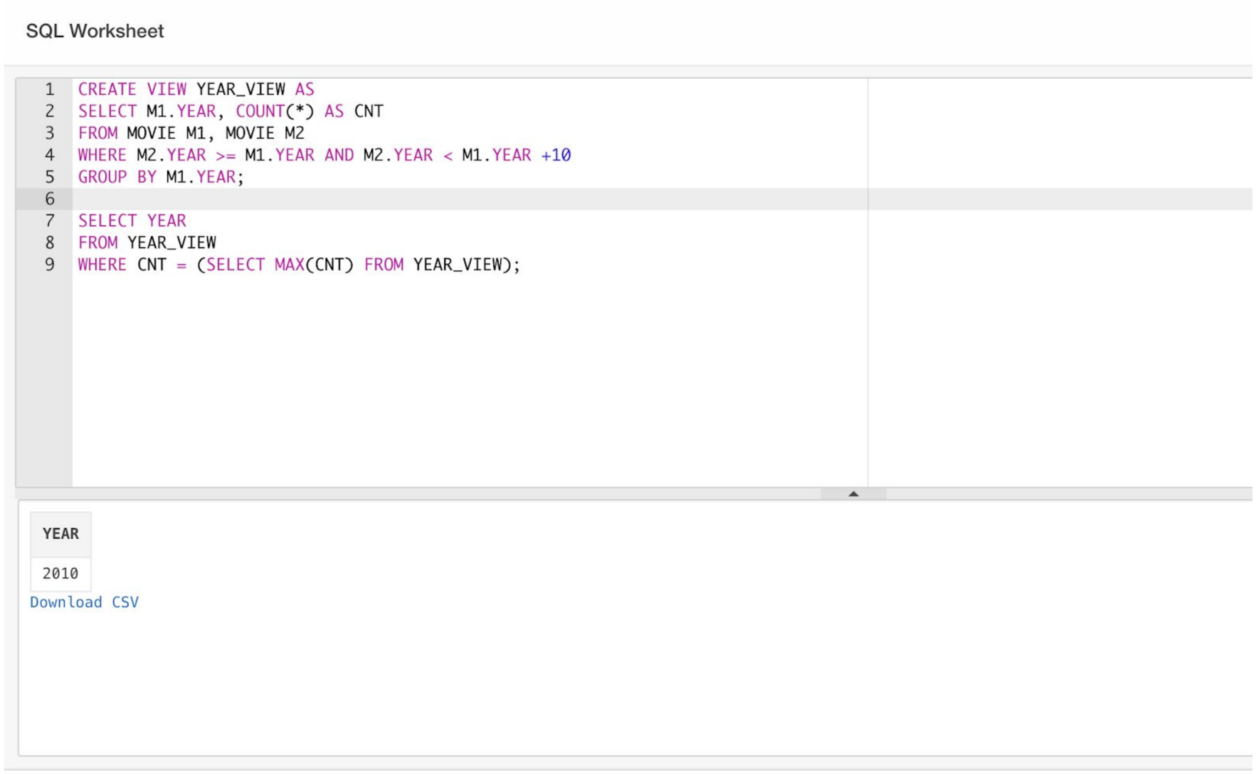
Q3.



```
CREATE VIEW LOWEST_RATED_MOVIE AS
SELECT MOVIE, COUNT(RATING) AS NUM_RATING, AVG(RATING) AS RATING
FROM REVIEWS
GROUP BY MOVIE
HAVING AVG(RATING) = (SELECT MIN(AVG(RATING)) FROM REVIEWS GROUP BY MOVIE)
AND
COUNT(RATING) = (SELECT MAX(COUNT(RATING)) FROM REVIEWS GROUP BY MOVIE);

SELECT M.MID, M.NAME, M.YEAR, V.RATING, V.NUM_RATING
FROM MOVIE M, LOWEST_RATED_MOVIE V
WHERE M.MID = V.MOVIE;
```

MID	NAME	YEAR	RATING	NUM_RATING
M13	The God Father part II	1975	4.5	2

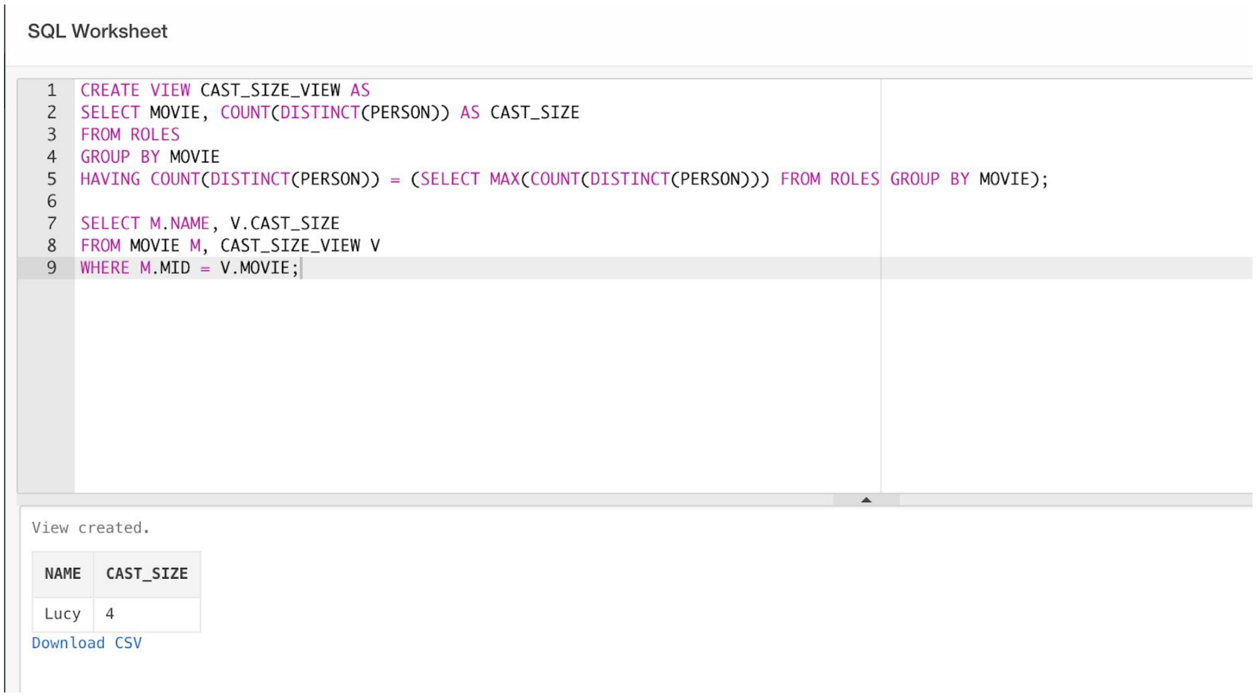
Q4.



```
CREATE VIEW YEAR_VIEW AS
SELECT M1.YEAR, COUNT(*) AS CNT
FROM MOVIE M1, MOVIE M2
WHERE M2.YEAR >= M1.YEAR AND M2.YEAR < M1.YEAR +10
GROUP BY M1.YEAR;

SELECT YEAR
FROM YEAR_VIEW
WHERE CNT = (SELECT MAX(CNT) FROM YEAR_VIEW);
```

YEAR
2010

Q5.


SQL Worksheet

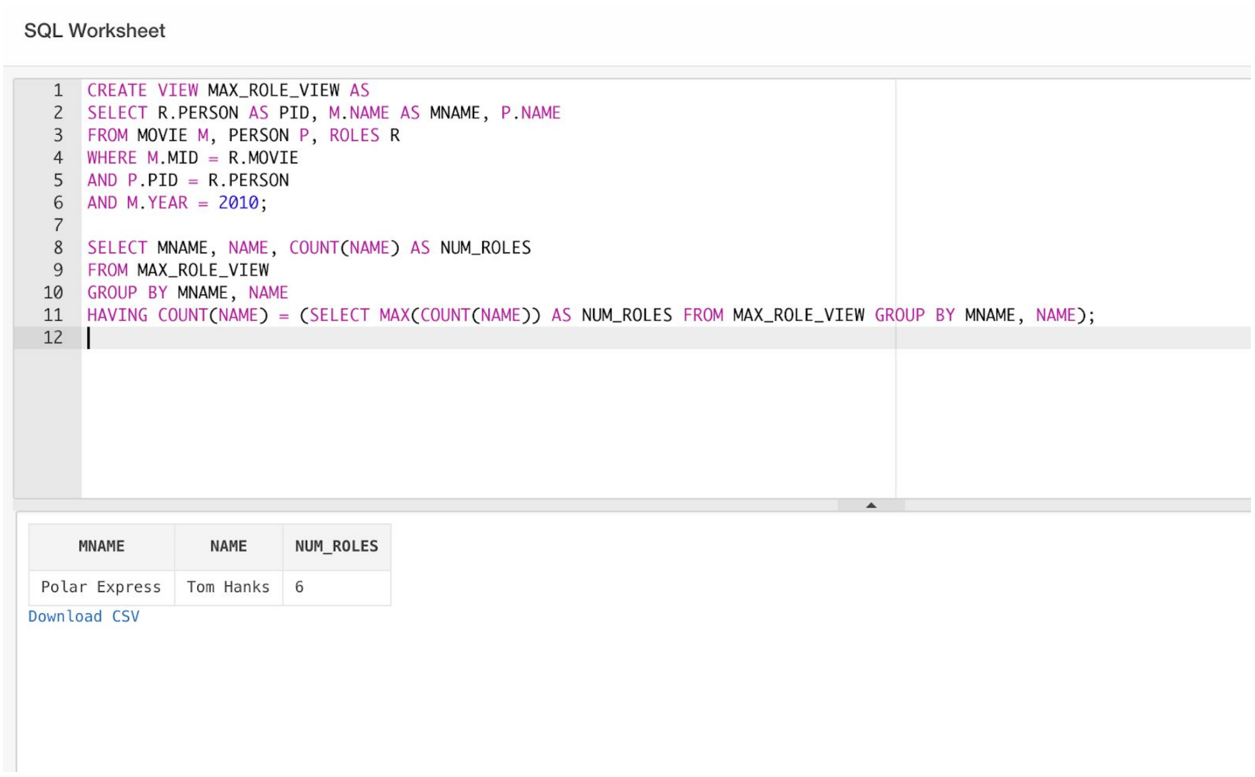
```sql
CREATE VIEW CAST_SIZE_VIEW AS
SELECT MOVIE, COUNT(DISTINCT(PERSON)) AS CAST_SIZE
FROM ROLES
GROUP BY MOVIE
HAVING COUNT(DISTINCT(PERSON)) = (SELECT MAX(COUNT(DISTINCT(PERSON))) FROM ROLES GROUP BY MOVIE);

SELECT M.NAME, V.CAST_SIZE
FROM MOVIE M, CAST_SIZE_VIEW V
WHERE M.MID = V.MOVIE;
```

```
View created.
```

NAME	CAST_SIZE
Lucy	4

Download CSV

Q6.

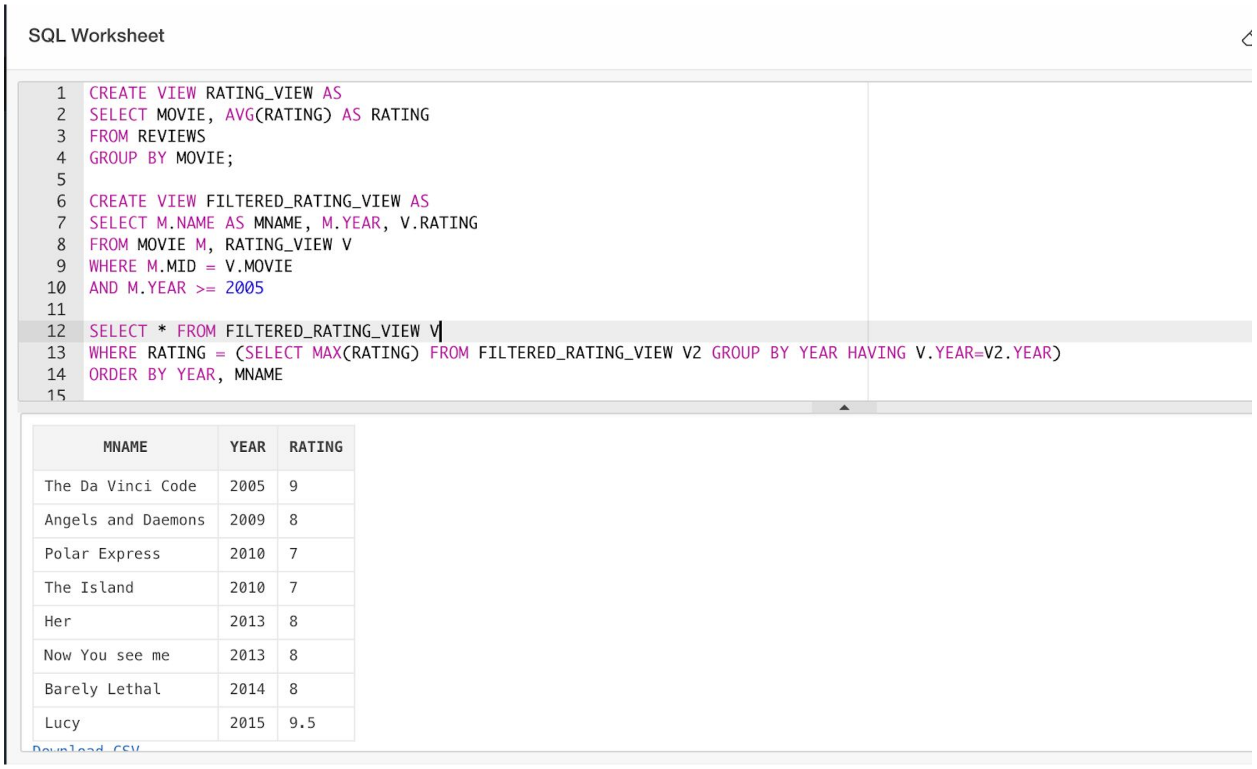

SQL Worksheet

```sql
CREATE VIEW MAX_ROLE_VIEW AS
SELECT R.PERSON AS PID, M.NAME AS MNAME, P.NAME
FROM MOVIE M, PERSON P, ROLES R
WHERE M.MID = R.MOVIE
AND P.PID = R.PERSON
AND M.YEAR = 2010;

SELECT MNAME, NAME, COUNT(NAME) AS NUM_ROLES
FROM MAX_ROLE_VIEW
GROUP BY MNAME, NAME
HAVING COUNT(NAME) = (SELECT MAX(COUNT(NAME)) AS NUM_ROLES FROM MAX_ROLE_VIEW GROUP BY MNAME, NAME);

```

MNAME	NAME	NUM_ROLES
Polar Express	Tom Hanks	6

Download CSV

Q7.



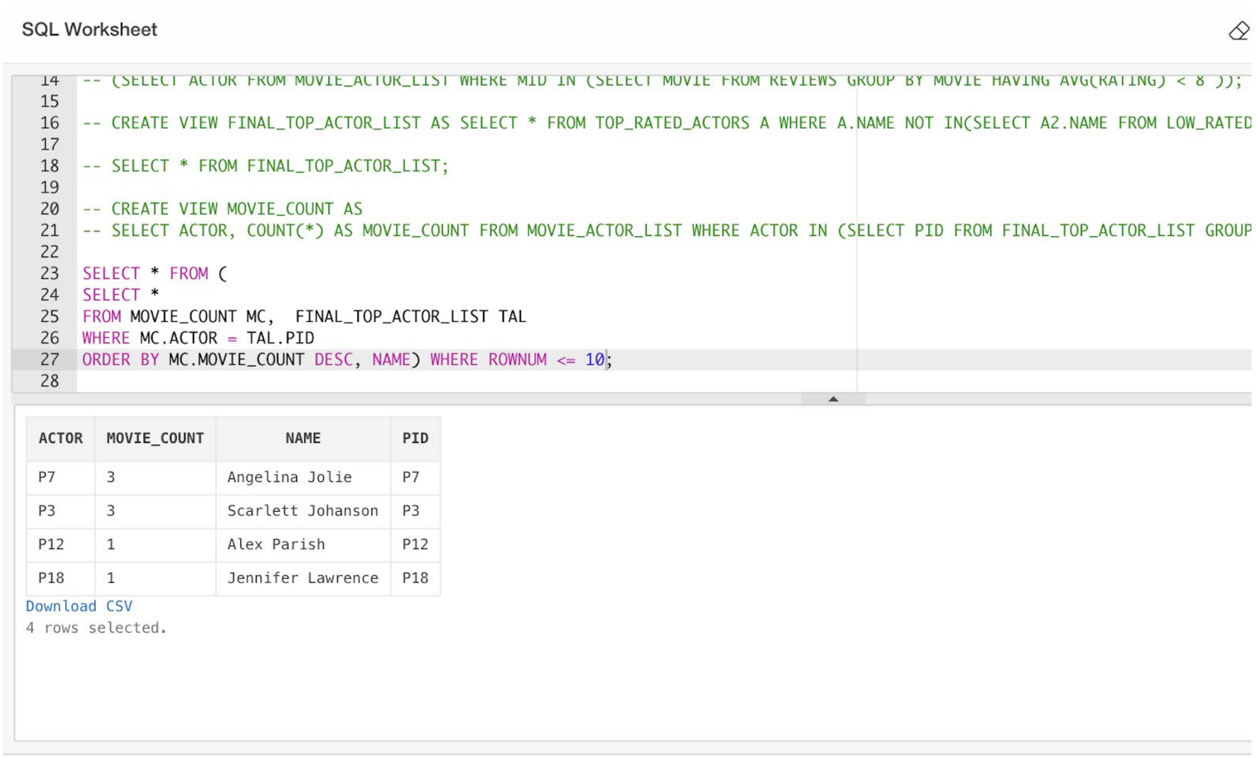
SQL Worksheet

```sql
CREATE VIEW RATING_VIEW AS
SELECT MOVIE, AVG(RATING) AS RATING
FROM REVIEWS
GROUP BY MOVIE;

CREATE VIEW FILTERED_RATING_VIEW AS
SELECT M.NAME AS MNAME, M.YEAR, V.RATING
FROM MOVIE M, RATING_VIEW V
WHERE M.MID = V.MOVIE
AND M.YEAR >= 2005

SELECT * FROM FILTERED_RATING_VIEW V
WHERE RATING = (SELECT MAX(RATING) FROM FILTERED_RATING_VIEW V2 GROUP BY YEAR HAVING V.YEAR=V2.YEAR)
ORDER BY YEAR, MNAME
```

MNAME	YEAR	RATING
The Da Vinci Code	2005	9
Angels and Daemons	2009	8
Polar Express	2010	7
The Island	2010	7
Her	2013	8
Now You see me	2013	8
Barely Lethal	2014	8
Lucy	2015	9.5

Download CSV

Q8.



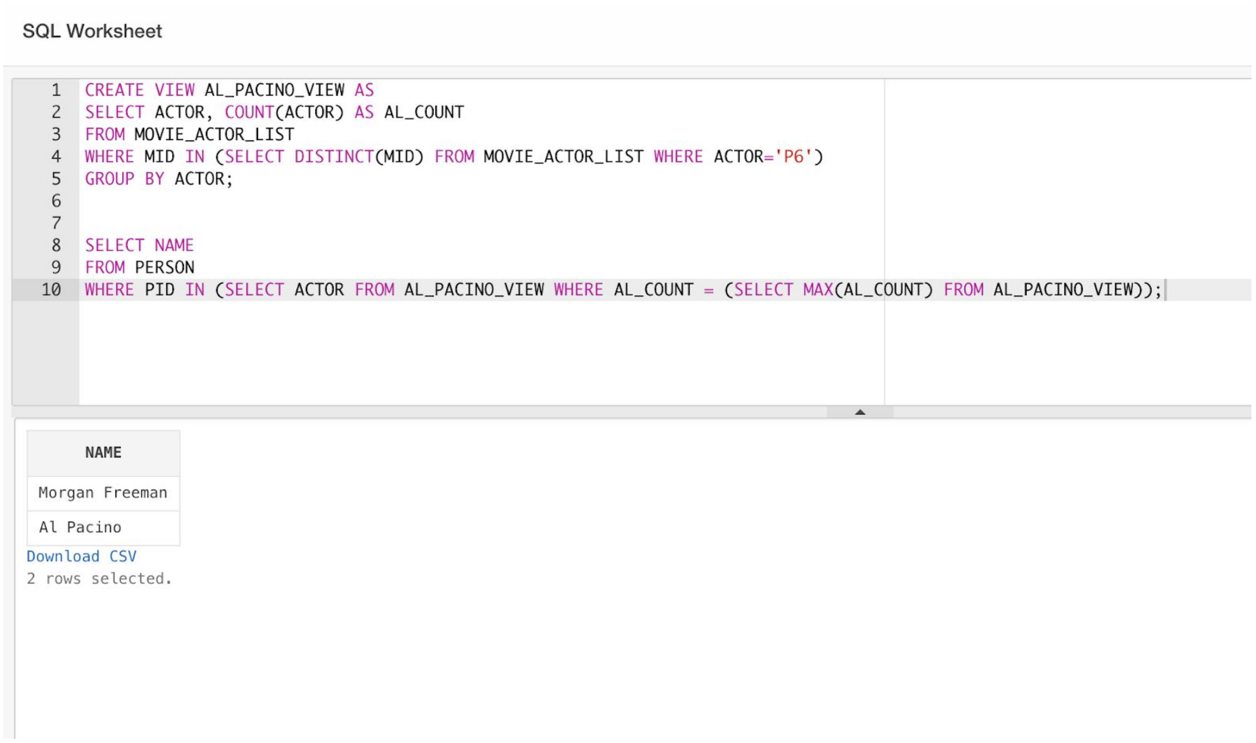
SQL Worksheet

```sql
-- (SELECT ACTOR FROM MOVIE_ACTOR_LIST WHERE MID IN (SELECT MOVIE FROM REVIEWS GROUP BY MOVIE HAVING AVG(RATING) < 8 ));

-- CREATE VIEW FINAL_TOP_ACTOR_LIST AS SELECT * FROM TOP_RATED_ACTORS A WHERE A.NAME NOT IN(SELECT A2.NAME FROM LOW_RATED

-- SELECT * FROM FINAL_TOP_ACTOR_LIST;

-- CREATE VIEW MOVIE_COUNT AS
-- SELECT ACTOR, COUNT(*) AS MOVIE_COUNT FROM MOVIE_ACTOR_LIST WHERE ACTOR IN (SELECT PID FROM FINAL_TOP_ACTOR_LIST GROUP

SELECT * FROM (
SELECT *
FROM MOVIE_COUNT MC,  FINAL_TOP_ACTOR_LIST TAL
WHERE MC.ACTOR = TAL.PID
ORDER BY MC.MOVIE_COUNT DESC, NAME) WHERE ROWNUM <= 10;
```

ACTOR	MOVIE_COUNT	NAME	PID
P7	3	Angelina Jolie	P7
P3	3	Scarlett Johanson	P3
P12	1	Alex Parish	P12
P18	1	Jennifer Lawrence	P18

Download CSV
4 rows selected.

Q9.



SQL Worksheet

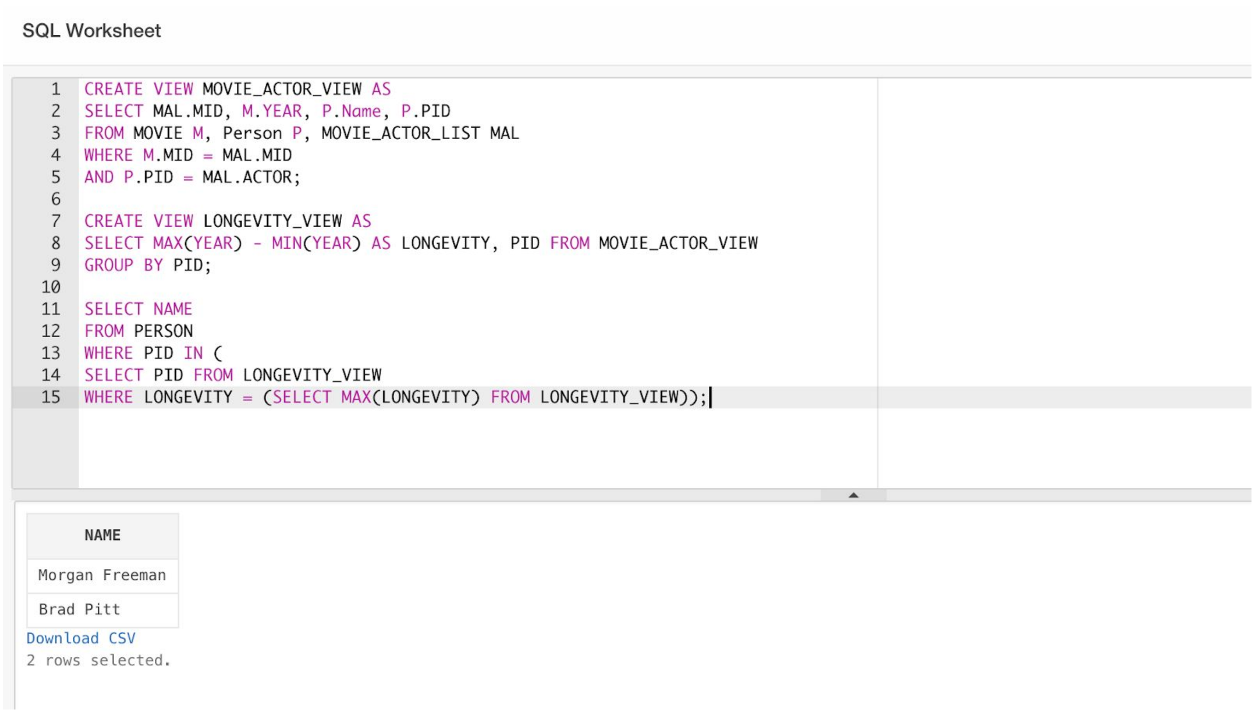
```
CREATE VIEW AL_PACINO_VIEW AS
SELECT ACTOR, COUNT(ACTOR) AS AL_COUNT
FROM MOVIE_ACTOR_LIST
WHERE MID IN (SELECT DISTINCT(MID) FROM MOVIE_ACTOR_LIST WHERE ACTOR='P6')
GROUP BY ACTOR;


SELECT NAME
FROM PERSON
WHERE PID IN (SELECT ACTOR FROM AL_PACINO_VIEW WHERE AL_COUNT = (SELECT MAX(AL_COUNT) FROM AL_PACINO_VIEW));
```

NAME
Morgan Freeman
Al Pacino

Download CSV

2 rows selected.

Q10.



SQL Worksheet

```
CREATE VIEW MOVIE_ACTOR_VIEW AS
SELECT MAL.MID, M.YEAR, P.Name, P.PID
FROM MOVIE M, Person P, MOVIE_ACTOR_LIST MAL
WHERE M.MID = MAL.MID
AND P.PID = MAL.ACTOR;

CREATE VIEW LONGEVITY_VIEW AS
SELECT MAX(YEAR) - MIN(YEAR) AS LONGEVITY, PID FROM MOVIE_ACTOR_VIEW
GROUP BY PID;

SELECT NAME
FROM PERSON
WHERE PID IN (
SELECT PID FROM LONGEVITY_VIEW
WHERE LONGEVITY = (SELECT MAX(LONGEVITY) FROM LONGEVITY_VIEW));
```

NAME
Morgan Freeman
Brad Pitt

Download CSV

2 rows selected.