Movies Database SQL Q/A

```
create database movies;
```

Create Table:

```
create table movies ( id int,
        title varchar(200),
        overview varchar(1000),
        release_date date,
        popularity decimal(10, 2),
        vote_average decimal(10, 2),
        vote_count int);
```

Load File Into Database:

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/movies.csv'
INTO TABLE moviess
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

Count Duplicate:

```
select title, release_date, count(*) as duplicate_count
from moviess
group by title, release_date
having count(*) > 1;
SELECT * FROM movies.movies;
```

1-- Find the top 10 highest rated movie

```
SELECT
   title, release_date
FROM
   movies
ORDER BY release_date DESC
LIMIT 10;
```

2-- List all movies released in 2023.

```
 SELECT
   title, release_date
FROM
   movies
WHERE
   EXTRACT(YEAR FROM release_date) = '2023';
```

3-- Find movies with more than 1000 votes and average rating above 8.

```
SELECT
   title, vote_average, vote_count
FROM
   movies
WHERE
   vote_count > 1000 AND vote_average > 8;
```

4-- Count how many movies were released each year.

```
SELECT
   EXTRACT(YEAR FROM release_date) AS years,
   COUNT(*) AS movie_count
FROM
   movies
GROUP BY years
ORDER BY years;
```

5-- Find the most popular movie in each year.

```
select * from (
select title, popularity, extract(year from release_date) as year,
rank() over (partition by extract(year from release_date)
order by popularity desc) as rnk
from movies) sub
where rnk = 1;
```

6-- Calculate average vote for each year.

```
SELECT
   EXTRACT(YEAR FROM release_date) AS years,
   ROUND(AVG(vote_average), 2) AS avg_vote
FROM movies GROUP BY years ORDER BY years;
```

7-- Find movies whose title or overview contains the word "love".

```
SELECT
   title, overview
FROM
   movies
WHERE
   LOWER(title) LIKE '%love%'
      AND LOWER(overview) LIKE '%love%';
```

8-- Show top 5 movies with highest popularity in 2020.

```
SELECT
   title, popularity
FROM
   movies
WHERE
   EXTRACT(YEAR FROM release_date) = 2020
ORDER BY popularity DESC
LIMIT 5;
```

9-- Compare average rating of movies before and after 2010.

```
SELECT
   CASE
      WHEN EXTRACT(YEAR FROM release_date) < 2010 THEN 'Before
2010'
      ELSE '2010 and after'
   END AS era,
   AVG(vote_average) AS avg_rating
FROM
   movies
GROUP BY era;
```

10-- Find the year with the highest average popularity.

```
select years from (
select extract(year from release_date) as years, avg(popularity) as
avg_popularity
from movies
group by years ) sub
order by avg_popularity desc
limit 1;
```