SQL Query examples:
Following are the required tables for the query.

Table Movie (Movie_ID, title, year, director)

Table Reviewer (Rating_ID, name)

Table Rating (Rating_ID, Movie_ID, stars, ratingDate)

Query:

***Find the titles of all movies directed by Steven Spielberg.**

```
select title
from movie
where director ='Steven Spielberg’;
```

***Find all years that have a movie that received a rating of 4 or 5, and sort them in increasing order.**

```
select distinct(Movie.year)
from Movie
where Movie.Movie_ID = Rating.Movie_ID
and stars =4 or stars =5
order by year asc;
```

***Find the titles of all movies that have no ratings**

```
select title
from movie
where Movie_ID Not in (select Movie_ID from Rating)
```

***Some reviewers didn't provide a date with their rating. Find the names of all reviewers who have ratings with a NULL value for the date.**

```
select name

from Reviewer,Rating

where Reviewer.Rating_ID = Rating.Rating_ID

and ratingDate is NULL;
```

***query to return the ratings data in a more readable format: reviewer name, movie title, stars, and ratingDate. Also, sort the data, first by reviewer name, then by movie title, and lastly by number of stars.**

```
select Reviewer.name,Movie.title,Rating.stars,Rating.ratingDate

from Reviewer,Movie,Rating

where Reviewer.Rating_ID = Rating.Rating_ID

and Movie.Movie_ID = Rating.Movie_ID

order by Reviewer.name,Movie.Title, Rating.Stars
```

***For all cases where the same reviewer rated the same movie twice and gave it a higher rating the second time, return the reviewer's name and the title of the movie.**

```
SELECT name, title

FROM Movie

INNER JOIN Rating R1 USING(Movie_ID)

INNER JOIN Rating R2 USING(Rating_ID)

INNER JOIN Reviewer USING(Rating_ID)

WHERE R1.Movie_ID = R2.Movie_ID AND R1.ratingDate < R2.ratingDate
AND R1.stars < R2.stars;
```

*** For each movie that has at least one rating, find the highest number of stars that movie received. Return the movie title and number of stars. Sort by movie title.**

```
select title, MAX(stars)

from Movie

inner join Rating using(Movie_ID)

group By Movie_ID

order by title;
```

***For each movie, return the title and the 'rating spread', that is, the difference between highest and lowest ratings given to that movie. Sort by rating spread from highest to lowest, then by movie title.**

```
select title, (MAX(stars)-MIN(stars)) as rating_spread

from Movie

inner join  Rating on Movie.Movie_ID = Rating.Movie_ID

group by title

order by rating_spread DESC, title;
```

***Find the difference between the average rating of movies released before 1980 and the average rating of movies released after 1980. (Make sure to calculate the average rating for each movie, then the average of those averages for movies before 1980 and movies after. Don't just calculate the overall average rating before and after 1980.)**

```
SELECT AVG(Before1980.avg) - AVG(After1980.avg)

FROM (

  SELECT AVG(stars) AS avg
  FROM Movie
  INNER JOIN Rating USING(Movie_ID)
  WHERE year < 1980
  GROUP BY Movie_ID
) AS Before1980, (
  SELECT AVG(stars) AS avg
  FROM Movie
  INNER JOIN Rating USING(Movie_ID)
  WHERE year > 1980
  GROUP BY Movie_ID
) AS After1980;
```

***Find the names of all reviewers who rated Gone with the Wind.**

```
select Distinct(name)
from Movie
inner join Rating Using(Movie_ID)
inner join Reviewer Using(Rating_ID)
where title = 'Gone with the Wind’;
```

***For any rating where the reviewer is the same as the director of the movie, return the reviewer name, movie title, and number of stars.**

```
select Reviewer.name, Movie.title,(rating.stars)
from Movie
inner join rating on rating.Movie_ID = Movie.Movie_ID
inner join Reviewer on Reviewer.Rating_ID = rating.Rating_ID
where Reviewer.name = Movie.director;
```

*Return all reviewer names and movie names together in a single list, alphabetized.

```
select title from movie
union
select name from reviewer
order by name, title;
```

* Find the titles of all movies not reviewed by Chris Jackson

```
select title
from movie
where Movie_ID not in
(
select Movie_ID
from Rating
inner join Reviewer on Reviewer.Rating_ID = rating.Rating_ID
where name ='Chris Jackson'
)
```

*For each rating that is the lowest (fewest stars) currently in the database, return the reviewer name, movie title, and number of stars.

```
select reviewer.name, movie.title,rating.stars
from movie
inner join rating on rating.Movie_ID = Movie.Movie_ID
inner join Reviewer on Reviewer.Rating_ID = rating.Rating_ID
where stars in
(select min(stars) from Rating);
```

*List movie titles and average ratings, from highest-rated to lowest-rated. If two or more movies have the same average rating, list them in alphabetical order.

```
select Movie.title, avg(stars) as average

from movie

inner join rating on rating.Movie_ID = Movie.Movie_ID

group by Movie.Movie_ID

order by average DESC, title;
```

*Find the names of all reviewers who have contributed three or more ratings.

```
select name

from reviewer

where

(select count (*)

from Rating

where reviewer.Rating_ID = rating.Rating_ID)>=3;
```

*Some directors directed more than one movie. For all such directors, return the titles of all movies directed by them, along with the director name. Sort by director name, then movie title.

```
select m1.title, director

from movie m1

inner join movie m2 using(director)

group by m1.Movie_ID

having count(*) >1

order by director, m1.title
```

***Find the movie(s) with the highest average rating. Return the movie title(s) and average ratin**g

```
select Movie.title, avg(stars) as average
from movie
inner join rating on rating.Movie_ID = Movie.Movie_ID
group by Movie.Movie_ID
order by average DESC, title
limit 1;
```

***Find the movie(s) with the lowest average rating. Return the movie title(s) and average rating**

```
select Movie.title, avg(stars) as average
from movie
inner join rating on rating.Movie_ID = Movie.Movie_ID
group by Movie.Movie_ID
order by average ASC, title
limit 2;
```

***For each director, return the director's name together with the title(s) of the movie(s) they directed that received the highest rating among all of their movies, and the value of that rating. Ignore movies whose director is NULL.**

```
select Director,title, MAX(stars)
from movie
inner join rating on rating.Movie_ID = Movie.Movie_ID
where director is not null
group by Director
```