You've started a new movie-rating website, and you've been collecting data on reviewers' ratings of various movies. There's not much data yet, but you can still try out some interesting queries. Here's the schema:

Movie (mID, title, year, director)
English: There is a movie with ID number *mID*, a *title*, a release *year*, and a *director*.

Reviewer (rID, name)
English: The reviewer with ID number *rID* has a certain *name*.

Rating (rID, mID, stars, ratingDate)
English: The reviewer *rID* gave the movie *mID* a number of *stars* rating (1-5) on a certain *ratingDate*.

Your queries will run over a small data set conforming to the schema. View the database. (You can also download the schema and data.)

Instructions: Each problem asks you to write a query in SQL. To run your query against our back-end sample database using SQLite, click the "Submit" button. You will see a display of your query result and the expected result. If the results match, your query will be marked "correct". You may run as many queries as you like for each question.

Important Notes:

- Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.
- Unless a specific result ordering is asked for, you can return the result rows in any order.
- *You are to translate the English into a SQL query that computes the desired result over all possible databases.* All we actually check is that your query gets the right answer on the small sample database. Thus, even if your solution is marked as correct, it is possible that your query does not correctly reflect the problem at hand. (For example, if we ask for a complex condition that requires accessing all of the tables, but over our small data set in the end the condition is satisfied only by Star Wars, then the query "select title from Movie where title = 'Star Wars'" will be marked correct even though it doesn't reflect the actual question.) Circumventing the system in this fashion will get you a high score on the exercises, but it won't help you learn SQL. On the other hand, an incorrect attempt at a general solution is unlikely to produce the right answer, so you shouldn't be led astray by our checking system.

You may perform these exercises as many times as you like, so we strongly encourage you to keep working with them until you complete the exercises with full credit.

Q1

1.0/1.0 point (ungraded)
Find the names of all reviewers who rated Gone with the Wind.

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT DISTINCT name
FROM Movie NATURAL JOIN Reviewer NATURAL JOIN Rating
WHERE title = "Gone with the Wind"
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Mike Anderson
Sarah Martinez

Expected Query Result:

Mike Anderson
Sarah Martinez

Submit

Q2

1.0/1.0 point (ungraded)

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.

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
1 SELECT name, title, stars
2 FROM Movie NATURAL JOIN Reviewer NATURAL JOIN Rating
3 WHERE director = name
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

James Cameron	Avatar	5

Expected Query Result:

James Cameron	Avatar	5

Submit

Q3

1.0/1.0 point (ungraded)

Return all reviewer names and movie names together in a single list, alphabetized. (Sorting by the first name of the reviewer and first word in the title is fine; no need for special processing on last names or removing "The".)

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT name
FROM Reviewer
UNION
SELECT title
FROM Movie
ORDER BY name
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Ashley White
Avatar
Brittany Harris
Chris Jackson
Daniel Lewis
E.T.
Elizabeth Thomas
Gone with the Wind
James Cameron
Mike Anderson
Raiders of the Lost Ark
Sarah Martinez
Snow White
Star Wars
The Sound of Music
Titanic

Expected Query Result:

Ashley White
Avatar
Brittany Harris
Chris Jackson
Daniel Lewis
E.T.
Elizabeth Thomas
Gone with the Wind

James Cameron
Mike Anderson
Raiders of the Lost Ark
Sarah Martinez
Snow White
Star Wars
The Sound of Music
Titanic

(Order matters)

Submit

Q4

1.0/1.0 point (ungraded)

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

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT title
FROM Movie
WHERE mID NOT IN (SELECT mID
                  FROM Reviewer NATURAL JOIN Rating
                  WHERE name = "Chris Jackson")
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Avatar
Gone with the Wind
Snow White
Star Wars
Titanic

Expected Query Result:

Avatar
Gone with the Wind
Snow White
Star Wars
Titanic

Submit

Q5

1.0/1.0 point (ungraded)

For all pairs of reviewers such that both reviewers gave a rating to the same movie, return the names of both reviewers. Eliminate duplicates, don't pair reviewers with themselves, and include each pair only once. For each pair, return the names in the pair in alphabetical order.

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT DISTINCT R1.name, R2.name
FROM (SELECT rID, mID, name FROM Rating NATURAL JOIN Reviewer) AS R1,
    (SELECT rID, mID, name FROM Rating NATURAL JOIN Reviewer) AS R2
WHERE R1.mID = R2.mID AND R1.name < R2.name
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Ashley White	Chris Jackson
Brittany Harris	Chris Jackson
Daniel Lewis	Elizabeth Thomas
Elizabeth Thomas	James Cameron
Mike Anderson	Sarah Martinez

Expected Query Result:

Ashley White	Chris Jackson
Brittany Harris	Chris Jackson
Daniel Lewis	Elizabeth Thomas
Elizabeth Thomas	James Cameron
Mike Anderson	Sarah Martinez

Submit

Q6

1.0/1.0 point (ungraded)

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

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT name, title, stars
FROM (Movie JOIN Reviewer) JOIN (SELECT * FROM Rating
                                 WHERE stars = (SELECT MIN(stars) AS minStars FROM Rating))
                           USING (mID, rID)
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Brittany Harris	Raiders of the Lost Ark	2
Brittany Harris	The Sound of Music	2
Chris Jackson	E.T.	2
Sarah Martinez	Gone with the Wind	2

Expected Query Result:

Brittany Harris	Raiders of the Lost Ark	2
Brittany Harris	The Sound of Music	2
Chris Jackson	E.T.	2
Sarah Martinez	Gone with the Wind	2

Submit

Q7

1.0/1.0 point (ungraded)

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.

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT title, AVG(stars) AS avgStars
FROM Movie JOIN Rating USING (mID)
GROUP BY mID
ORDER BY avgStars DESC, title
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Snow White	4.5
Avatar	4.0
Raiders of the Lost Ark	3.33333333333
Gone with the Wind	3.0
E.T.	2.5
The Sound of Music	2.5

Expected Query Result:

Snow White	4.5
Avatar	4.0
Raiders of the Lost Ark	3.33333333333
Gone with the Wind	3.0
E.T.	2.5
The Sound of Music	2.5

(Order matters)

Submit

Q8

1.0/1.0 point (ungraded)

Find the names of all reviewers who have contributed three or more ratings. (As an extra challenge, try writing the query without HAVING or without COUNT.)

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
1  -- ASSUMPTION: combination (rID, mID, stars) is unique (i.e., can be used as key);
2
3  SELECT DISTINCT name
4  FROM Reviewer JOIN Rating USING (rID)
5  WHERE rID IN
6      (SELECT R1.rID
7       FROM Rating AS R1, Rating AS R2, Rating AS R3
8       WHERE R1.rID = R2.rID AND R2.rID = R3.rID
9             AND R1.mID || R1.stars < R2.mID || R2.stars
10            AND R2.mID || R2.stars < R3.mID || R3.stars)
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Brittany Harris
Chris Jackson

Expected Query Result:

Brittany Harris
Chris Jackson

Submit

Q9

1.0/1.0 point (ungraded)

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. (As an extra challenge, try writing the query both with and without COUNT.)

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
/*
-- using COUNT:
SELECT title, director
FROM Movie JOIN (SELECT director, COUNT (*) AS Movies
                 FROM Movie GROUP BY director HAVING Movies > 1)
USING (director)
ORDER BY director, title
*/

SELECT title, director
FROM Movie JOIN (SELECT M1.director
                 FROM Movie AS M1, Movie AS M2
                 WHERE M1.director = M2.director AND M1.mID < M2.mID)
USING (director)
ORDER BY director, title
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Avatar	James Cameron
Titanic	James Cameron
E.T.	Steven Spielberg
Raiders of the Lost Ark	Steven Spielberg

Expected Query Result:

Avatar	James Cameron
Titanic	James Cameron
E.T.	Steven Spielberg
Raiders of the Lost Ark	Steven Spielberg

(Order matters)

Submit

Q10

1.0/1.0 point (ungraded)

Find the movie(s) with the highest average rating. Return the movie title(s) and average rating. (Hint: This query is more difficult to write in SQLite than other systems; you might think of it as finding the highest average rating and then choosing the movie(s) with that average rating.)

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT title, AVG(stars)
FROM Movie JOIN Rating USING (mID)
WHERE mID IN (
SELECT mID
```

```
FROM (SELECT mID, AVG(stars) AS avgStars FROM Rating GROUP BY mID)
WHERE avgStars = (SELECT MAX(avgStars)
                  FROM (SELECT mID, AVG(stars) AS avgStars
                        FROM Rating
                        GROUP BY mID)
                  JOIN Movie USING (mID))
)
GROUP BY mID
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

Snow White	4.5

Expected Query Result:

Snow White	4.5

Submit

Q11

1.0/1.0 point (ungraded)

Find the movie(s) with the lowest average rating. Return the movie title(s) and average rating. (Hint: This query may be more difficult to write in SQLite than other systems; you might think of it as finding the lowest average rating and then choosing the movie(s) with that average rating.)

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT title, AVG(stars)
FROM Movie JOIN Rating USING (mID)
WHERE mID IN (
SELECT mID
FROM (SELECT mID, AVG(stars) AS avgStars FROM Rating GROUP BY mID)
WHERE avgStars = (SELECT MIN(avgStars)
                  FROM (SELECT mID, AVG(stars) AS avgStars
                        FROM Rating
                        GROUP BY mID)
                  JOIN Movie USING (mID))
)
GROUP BY mID
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

E.T.	2.5
The Sound of Music	2.5

Expected Query Result:

E.T.	2.5
The Sound of Music	2.5

Submit

Q12

1.0/1.0 point (ungraded)

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.

Note: Your queries are executed using SQLite, so you must conform to the SQL constructs supported by SQLite.

```
SELECT director, title, MAX(maxStars)
FROM (SELECT mID, MAX(stars) AS maxStars
        FROM Rating
        GROUP BY mID) JOIN Movie USING (mID)
GROUP BY director
HAVING director NOT NULL
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

James Cameron	Avatar	5
Robert Wise	The Sound of Music	3
Steven Spielberg	Raiders of the Lost Ark	4
Victor Fleming	Gone with the Wind	4

Expected Query Result:

James Cameron	Avatar	5
Robert Wise	The Sound of Music	3
Steven Spielberg	Raiders of the Lost Ark	4
Victor Fleming	Gone with the Wind	4

Submit