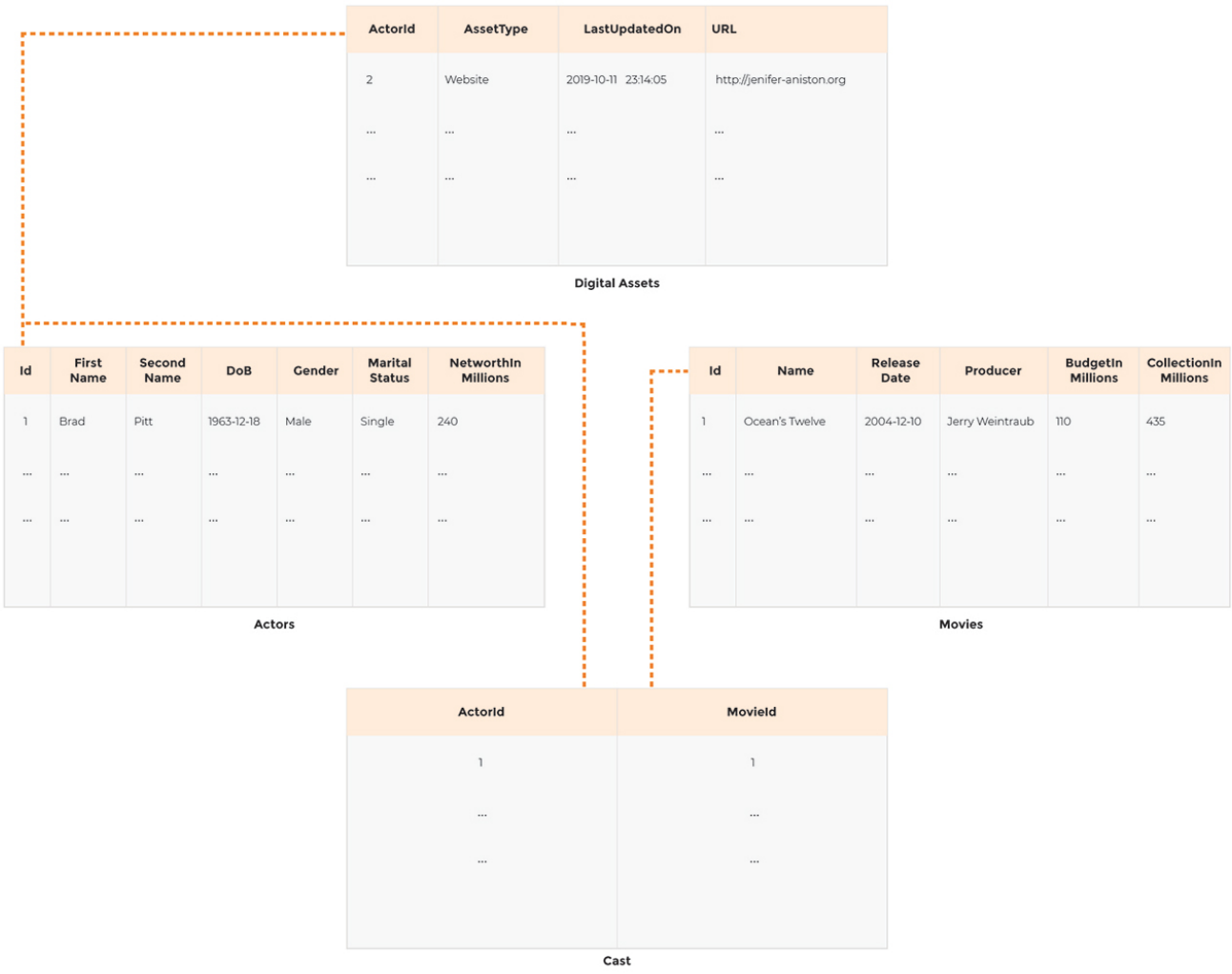
Solution Practice Set 2

Solution Practice Set 2

The database relationship model is reprinted below for reference.



Connect to the terminal below by clicking in the widget. Once connected, the command line prompt will show up. Enter or copy and paste the command **./DataJek/Lessons/quiz.sh** and wait for the MySQL prompt to start-up.

```
-- The lesson queries are reproduced below for convenient copy/paste into the terminal.

-- Question # 1, Query 1
SELECT Id, FirstName, SecondName, MovieId
FROM Actors
INNER JOIN Cast
ON Id = ActorId;

-- Question # 1, Query 2
SELECT Id, COUNT(*)
FROM Actors
INNER JOIN Cast
ON Id = ActorId
GROUP BY Id;

-- Question # 1, Query 3
SELECT Id,
COUNT(*) AS MovieCount
FROM Actors
INNER JOIN Cast
ON Id = ActorId
GROUP BY Id
HAVING MovieCount > 1;

-- Question # 1, Query 4
SELECT CONCAT (FirstName, " ", SecondName)
AS Actor_Names,
Movie_Count
FROM Actors a
INNER JOIN (SELECT Id,
            COUNT(*) AS Movie_Count
            FROM Actors
            INNER JOIN Cast
            ON Id = ActorId
            GROUP BY Id
            HAVING Movie_Count > 1) AS tbl
ON tbl.Id = a.Id;

-- Question # 2, Query 1
SELECT Id
FROM Movies
WHERE Name="Mr & Mrs. Smith";

-- Question # 2, Query 2
SELECT ActorId
FROM Cast
WHERE MovieId IN (SELECT Id
                  FROM Movies
                  WHERE Name="Mr & Mrs. Smith");

-- Question # 2, Query 3
SELECT CONCAT(FirstName, " ", SecondName)
AS "Cast Of Mr. & Mrs. Smith"
FROM Actors
WHERE Id IN ( SELECT ActorId
              FROM Cast
              WHERE MovieId IN (SELECT Id
                                FROM Movies
                                WHERE Name="Mr & Mrs. Smith"));

-- Question # 2, Query 4
SELECT CONCAT(FirstName, " ", SecondName)
```

```
AS "Cast Of Mr. & Mrs. Smith"
FROM Actors
INNER JOIN (SELECT ActorId
            FROM Cast
            INNER JOIN Movies
            ON MovieId = Id
            WHERE Name="Mr & Mrs. Smith") AS tbl
ON tbl.ActorId = Id;

-- Question # 3, Query 1
SELECT Name, ActorId
FROM Movies
INNER JOIN Cast
On Id = MovieId;

-- Question # 3, Query 2
SELECT tbl.Name AS Movie_Name,
CONCAT(FirstName, " ", SecondName) AS Actor_Name
FROM Actors
INNER JOIN (SELECT Name, ActorId
            FROM Movies
            INNER JOIN Cast
            On Id = MovieId) AS tbl
ON tbl.ActorId = Id
ORDER BY tbl.Name ASC;

-- Question # 4, Query 1
SELECT tbl.Name, COUNT(*)
FROM Actors
INNER JOIN (SELECT Name, ActorId, MovieId
            FROM Movies
            INNER JOIN Cast
            On Id = MovieId) AS tbl
ON tbl.ActorId = Id
GROUP BY tbl.MovieId;

-- Question # 4, Query 2
SELECT MovieId, COUNT(*)
FROM Cast
GROUP BY MovieId;

-- Question # 4, Query 3
SELECT Name AS Movie_Name,
Actor_Count
FROM Movies
INNER JOIN (SELECT MovieId, COUNT(*) AS Actor_Count
FROM Cast
GROUP BY MovieId) AS tbl
ON tbl.MovieID = Id;

-- Question # 5, Query 1
SELECT Id
FROM Actors
WHERE FirstName = "Tom"
AND SecondName = "Cruise";

-- Question # 5, Query 2
SELECT MovieId
FROM Cast
WHERE ActorId = (SELECT Id
                 FROM Actors
                 WHERE FirstName = "Tom"
```

```
                AND SecondName = "Cruise");

-- Question # 5, Query 3
SELECT DISTINCT Producer
FROM Movies
WHERE Id IN (SELECT MovieId
             FROM Cast
             WHERE ActorId = (SELECT Id
                              FROM Actors
                              WHERE FirstName = "Tom"
                              AND SecondName = "Cruise"));

-- Question # 5, Query 4
SELECT DISTINCT Producer
FROM Movies
WHERE Id NOT IN (SELECT MovieId
             FROM Cast
             WHERE ActorId = (SELECT Id
                              FROM Actors
                              WHERE FirstName = "Tom"
                              AND SecondName = "Cruise"));

-- Question # 5, Query 5
SELECT DISTINCT Producer
FROM Movies
WHERE Producer
NOT IN (SELECT Producer
        FROM Movies
        WHERE Id IN (SELECT MovieId
                     FROM Cast
                     WHERE ActorId = (SELECT Id
                                      FROM Actors
                                      WHERE FirstName = "Tom"
                                AND SecondName = "Cruise")));
```

Terminal

Question # 1

Write a query to display all those actors who have acted in 2 or more movies.

In this question we are required to print the names of the actors who have acted in two or more movies. The names of the actors are in the **Actors** table and the number of movies an actor has appeared in is in the **Cast** table. If we join the two tables, we can get the name of actor and the ID of the movie that the actor has starred in. Let's see how that looks like:

```
SELECT Id, FirstName, SecondName, MovieId
FROM Actors

INNER JOIN Cast
ON Id = ActorId;
```

```
mysql> SELECT Id, FirstName, SecondName, MovieId
    -> FROM Actors
    -> INNER JOIN Cast
    -> ON Id = ActorId;
+----+-----------+------------+---------+
| Id | FirstName | SecondName | MovieId |
+----+-----------+------------+---------+
|  1 | Brad      | Pitt       |       1 |
|  1 | Brad      | Pitt       |       2 |
|  1 | Brad      | Pitt       |       5 |
|  3 | Angelina  | Jolie      |       2 |
|  4 | Johnny    | Depp       |       4 |
|  5 | Natalie   | Portman    |      10 |
|  6 | Tom       | Cruise     |       3 |
|  8 | Kim       | Kardashian |       9 |
| 10 | Shahrukh  | Khan       |       8 |
| 12 | Khloe     | Kardashian |       9 |
| 13 | Kourtney  | Kardashian |       9 |
| 14 | Abhishek  | Bachchan   |       8 |
| 15 | Frank     | Sinatra    |       6 |
+----+-----------+------------+---------+
13 rows in set (0.00 sec)
```

Each row contains a movie ID in which an actor has starred. We can **GROUP BY** the result of the above query by ID of each actor so that all the movies that an actor has acted in, fall into the same group. Next, we simply count the rows in each group. So far, we have the following:

```
SELECT Id, COUNT(*)
FROM Actors
INNER JOIN Cast
ON Id = ActorId
GROUP BY Id;
```

```
mysql> SELECT Id, COUNT(*)
    -> FROM Actors
    -> INNER JOIN Cast
    -> ON Id = ActorId
    -> GROUP BY Id;
+----+----------+
| Id | COUNT(*) |
+----+----------+
|  1 |        3 |
|  3 |        1 |
|  4 |        1 |
|  5 |        1 |
|  6 |        1 |
|  8 |        1 |
| 10 |        1 |
| 12 |        1 |
| 13 |        1 |
| 14 |        1 |
| 15 |        1 |
+----+----------+
11 rows in set (0.00 sec)
```

Note we have removed the columns in the **SELECT** clause since they don't participate in the aggregation criteria. Now we'll apply the restriction to only list those groups which have more than one row to fulfill the requirement to print names of only those actors who have acted in at least two movies.

```
SELECT Id,
COUNT(*) AS MovieCount
FROM Actors
INNER JOIN Cast
ON Id = ActorId
GROUP BY Id
HAVING MovieCount > 1;
```

```
HAVING MovieCount > 1;
```

```
mysql> SELECT Id,
    -> COUNT(*) AS MovieCount
    -> FROM Actors
    -> INNER JOIN Cast
    -> ON Id = ActorId
    -> GROUP BY Id
    -> HAVING MovieCount > 1;
+----+------------+
| Id | MovieCount |
+----+------------+
|  1 |          3 |
+----+------------+
1 row in set (0.00 sec)
```

The last piece is to print the actor's name. The above query is printing actor Id and the count of movies the actor has been part of. We can join the result of the above query with the **Actors** table based on the common actor ID column the two tables hold. From the joined result we can extract the actor name and the movie count columns. The complete query appears below:

```
SELECT CONCAT (FirstName, " ", SecondName)

AS Actor_Names,

Movie_Count

FROM Actors a

INNER JOIN (SELECT Id,
            COUNT(*) AS Movie_Count
            FROM Actors
            INNER JOIN Cast
            ON Id = ActorId
            GROUP BY Id
            HAVING Movie_Count > 1) AS tbl

ON tbl.Id = a.Id;
```

```
mysql> SELECT CONCAT (FirstName, " ", SecondName)
    ->
    -> AS Actor_Names,
    ->
    -> Movie_Count
    ->
    -> FROM Actors a
    ->
    -> INNER JOIN (SELECT Id,
    ->              COUNT(*) AS Movie_Count
    ->              FROM Actors
    ->              INNER JOIN Cast
    ->              ON Id = ActorId
    ->              GROUP BY Id
    ->              HAVING Movie_Count > 1) AS tbl
    ->
    -> ON tbl.Id = a.Id;
+-------------+-------------+
| Actor_Names | Movie_Count |
+-------------+-------------+
| Brad Pitt   |           3 |
+-------------+-------------+
1 row in set (0.00 sec)
```

Question # 2

Find the cast of movie Mr and Mrs. Smith without using joins.

We are given the name of the movie but not its ID. We'll first need a query to extract the ID of the movie. If we were asked to find the cast of the movie Ocean's 11, which has two entries in the **Movies** table since there have been

two movies by the same name, we would need further information to narrow down to exactly one movie. In the case of, Mr. & Mrs. Smith we have only one entry and we can get the ID of the movie as follows:

```
SELECT Id
FROM Movies
WHERE Name="Mr & Mrs. Smith";
```

```
mysql> SELECT Id
    -> FROM Movies
    -> WHERE Name="Mr & Mrs. Smith";
+----+
| Id |
+----+
|  2 |
+----+
1 row in set (0.00 sec)
```

We can identify the cast of the movie by looking for rows in the **Cast** table that have the same value for the MovieID column as the ID of the movie Mr. & Mrs. Smith. We'll use nested query to list the IDs of the cast as follows:

```
SELECT ActorId
FROM Cast
WHERE MovieId IN (SELECT Id
                  FROM Movies
                  WHERE Name="Mr & Mrs. Smith");
```

```
mysql> SELECT ActorId
    -> FROM Cast
    -> WHERE MovieId IN (SELECT Id
    ->                   FROM Movies
    ->                   WHERE Name="Mr & Mrs. Smith");
+---------+
| ActorId |
+---------+
|       1 |
|       3 |
+---------+
2 rows in set (0.00 sec)
```

Once we have the actor IDs, we can find all the names of the actors from the **Actors** table by matching on the ID. The complete query appears below:

```
SELECT CONCAT(FirstName, " ", SecondName)
AS "Cast Of Mr. & Mrs. Smith"
FROM Actors
WHERE Id IN ( SELECT ActorId
              FROM Cast
              WHERE MovieId IN (SELECT Id
                                FROM Movies
                                WHERE Name="Mr & Mrs. Smith"));
```

```
mysql> SELECT CONCAT(FirstName, " ", SecondName)
    -> AS "Cast Of Mr. & Mrs. Smith"
    -> FROM Actors
    -> WHERE Id IN ( SELECT ActorId
    ->               FROM Cast
    ->               WHERE MovieId IN (SELECT Id
    ->                                 FROM Movies
    ->                                 WHERE Name="Mr & Mrs. Smith"));
+-------------------------+
| Cast Of Mr. & Mrs. Smith |
+-------------------------+
| Brad Pitt               |
| Angelina Jolie          |
+-------------------------+
2 rows in set (0.00 sec)
```

The solution we provided used nested queries, we could also have arrived at the same result using joins. The complete query with joins is as follows:

```
SELECT CONCAT(FirstName, " ", SecondName)

AS "Cast Of Mr. & Mrs. Smith"

FROM Actors

INNER JOIN (SELECT ActorId
            FROM Cast
            INNER JOIN Movies
            ON MovieId = Id
            WHERE Name="Mr & Mrs. Smith") AS tbl

ON tbl.ActorId = Id;
```

```
mysql> SELECT CONCAT(FirstName, " ", SecondName)
    ->
    -> AS "Cast Of Mr. & Mrs. Smith"
    ->
    -> FROM Actors
    ->
    -> INNER JOIN (SELECT ActorId
    ->             FROM Cast
    ->             INNER JOIN Movies
    ->             ON MovieId = Id
    ->             WHERE Name="Mr & Mrs. Smith") AS tbl
    ->
    -> ON tbl.ActorId = Id;
+--------------------------+
| Cast Of Mr. & Mrs. Smith |
+--------------------------+
| Brad Pitt                |
| Angelina Jolie           |
+--------------------------+
2 rows in set (0.00 sec)
```

Question # 3

Print a list of movie and the actor(s) who participated in the movie ordered by movie name.

In this question, we are asked to list the movie and actor name pairs. The **Cast** table exactly does that; however, it lists integer IDs for actors and movies instead of actual names. We have to expand the IDs into names for both actors and movies. Let's start by joining the **Cast** table with the **Actors** table on movie ID. The resultant derived table will have the movie names. The query is shown below:

```
SELECT Name, ActorId
FROM Movies
INNER JOIN Cast
On Id = MovieId;
```

```
mysql> SELECT Name, ActorId
    -> FROM Movies
    -> INNER JOIN Cast
    -> On Id = MovieId;
+----------------------------------+---------+
| Name                             | ActorId |
+----------------------------------+---------+
| Ocean's Twelve                   |       1 |
| Mr & Mrs. Smith                  |       1 |
| Ocean's 11                       |       1 |
| Mr & Mrs. Smith                  |       3 |
| London Fields                    |       4 |
| Avengers: Endgame                |       5 |
| Mission: Impossible - Fallout    |       6 |
| Keeping Up with the Kardashians  |       8 |
| Mohabbatein                      |      10 |
| Keeping Up with the Kardashians  |      12 |
| Keeping Up with the Kardashians  |      13 |
| Mohabbatein                      |      14 |
| Ocean's 11                       |      15 |
+----------------------------------+---------+
13 rows in set (0.00 sec)
```

Next, we can join the derived table from the above query with the Actors table based on actor ID and thus extract the actor name. Lastly, don't forget to order the result by movie name. The complete query is shown below:

```
SELECT tbl.Name AS Movie_Name,

CONCAT(FirstName, " ", SecondName) AS Actor_Name

FROM Actors

INNER JOIN (SELECT Name, ActorId
            FROM Movies
```

```
            INNER JOIN Cast
            On Id = MovieId) AS tbl


ON tbl.ActorId = Id

ORDER BY tbl.Name ASC;
```

```
mysql> SELECT tbl.Name AS Movie_Name,
    ->
    -> CONCAT(FirstName, " ", SecondName) AS Actor_Name
    ->
    -> FROM Actors
    ->
    -> INNER JOIN (SELECT Name, ActorId
    ->              FROM Movies
    ->              INNER JOIN Cast
    ->              On Id = MovieId) AS tbl
    ->
    -> ON tbl.ActorId = Id
    ->
    -> ORDER BY tbl.Name ASC;
+---------------------------------+---------------------+
| Movie_Name                      | Actor_Name          |
+---------------------------------+---------------------+
| Avengers: Endgame               | Natalie Portman     |
| Keeping Up with the Kardashians | Kourtney Kardashian |
| Keeping Up with the Kardashians | Kim Kardashian      |
| Keeping Up with the Kardashians | Khloe Kardashian    |
| London Fields                   | Johnny Depp         |
| Mission: Impossible - Fallout   | Tom Cruise          |
| Mohabbatein                     | Abhishek Bachchan   |
| Mohabbatein                     | Shahrukh Khan       |
| Mr & Mrs. Smith                 | Brad Pitt           |
| Mr & Mrs. Smith                 | Angelina Jolie      |
| Ocean's 11                      | Frank Sinatra       |
| Ocean's 11                      | Brad Pitt           |
| Ocean's Twelve                  | Brad Pitt           |
+---------------------------------+---------------------+
13 rows in set (0.00 sec)
```

Question # 4

Print the count of actors in each movie.

The requirement to use count hints towards aggregation. In the previous query, we were able to print the pair of actor and movie names. The astute reader would immediately realize that if we **GROUP BY** the results by movie name from the previous query, all the actors who acted in that movie will fall into that bucket. We can then count the number of actors in each bucket/group and report that as the result. However, there's one catch! If we group by movie name, then actors from two movies with the same name will all fall in one bucket and distort the counts. Therefore, we must **GROUP BY** movie ID.

Modifying previous query, we get:

```
SELECT tbl.Name, COUNT(*)

FROM Actors

INNER JOIN (SELECT Name, ActorId, MovieId
            FROM Movies
            INNER JOIN Cast
            On Id = MovieId) AS tbl

ON tbl.ActorId = Id

GROUP BY tbl.MovieId;
```

```
mysql> SELECT tbl.Name, COUNT(*)
    ->
    -> FROM Actors
    ->
    -> INNER JOIN (SELECT Name, ActorId, MovieId
    ->              FROM Movies
    ->              INNER JOIN Cast
    ->              On Id = MovieId) AS tbl
    ->
    -> ON tbl.ActorId = Id
    ->
    -> GROUP BY tbl.MovieId;
+---------------------------------+----------+
| Name                            | COUNT(*) |
+---------------------------------+----------+
| Ocean's Twelve                  |        1 |
| Mr & Mrs. Smith                 |        2 |
| Mission: Impossible - Fallout   |        1 |
| London Fields                   |        1 |
| Ocean's 11                      |        1 |
| Ocean's 11                      |        1 |
| Mohabbatein                     |        2 |
| Keeping Up with the Kardashians |        3 |
| Avengers: Endgame               |        1 |
+---------------------------------+----------+
9 rows in set (0.00 sec)
```

The above query gets us what we want but is convoluted and unnecessarily complex. Note that the information we require is available in the **Cast** and the **Movies** table. We don't require the actor names in our final result, so we don't need to join with the **Actors** table. We can **GROUP BY** the contents of the **Movies** table by movie ID and find the counts of actors for each movie as follows:

```
SELECT MovieId, COUNT(*)
```

```
FROM Cast
GROUP BY MovieId;
```

```
mysql> SELECT MovieId, COUNT(*)
    -> FROM Cast
    -> GROUP BY MovieId;
+---------+----------+
| MovieId | COUNT(*) |
+---------+----------+
|       1 |        1 |
|       2 |        2 |
|       3 |        1 |
|       4 |        1 |
|       5 |        1 |
|       6 |        1 |
|       8 |        2 |
|       9 |        3 |
|      10 |        1 |
+---------+----------+
9 rows in set (0.00 sec)
```

Now we can join the derived table from the above query with the **Movies** table on the movie to infer the movie name. The query is shown as follows:

```
SELECT Name AS Movie_Name,

Actor_Count

FROM Movies

INNER JOIN (SELECT MovieId, COUNT(*) AS Actor_Count
FROM Cast
GROUP BY MovieId) AS tbl

ON tbl.MovieID = Id;
```

```
mysql> SELECT Name AS Movie_Name,
    ->
    -> Actor_Count
    ->
    -> FROM Movies
    ->
    -> INNER JOIN (SELECT MovieId, COUNT(*) AS Actor_Count
    -> FROM Cast
    -> GROUP BY MovieId) AS tbl
    ->
    -> ON tbl.MovieID = Id;
+--------------------------------+-------------+
| Movie_Name                     | Actor_Count |
+--------------------------------+-------------+
| Avengers: Endgame              |           1 |
| Keeping Up with the Kardashians |           3 |
| London Fields                  |           1 |
| Mission: Impossible - Fallout  |           1 |
| Mohabbatein                    |           2 |
| Mr & Mrs. Smith                |           2 |
| Ocean's 11                     |           1 |
| Ocean's 11                     |           1 |
| Ocean's Twelve                 |           1 |
+--------------------------------+-------------+
9 rows in set (0.01 sec)
```

Question # 5

List the names of Producers who never produced a movie for Tom Cruise.

Information about movie producers resides in the **Movies** table. We don't know the ID of Tom Cruise on top of our head, so we'll need to query the **Actors** table too. Finally, the **Cast** table will let us connect the various queries to find all producers who didn't include Tom Cruise in the cast. Let's start by

first finding the ID of Tom Cruise.

```
SELECT Id
FROM Actors
WHERE FirstName = "Tom"
AND SecondName = "Cruise";
```

```
mysql> SELECT Id
    -> FROM Actors
    -> WHERE FirstName = "Tom"
    -> AND SecondName = "Cruise";
+----+
| Id |
+----+
|  6 |
+----+
1 row in set (0.00 sec)
```

Next, we'll find all the movies in which Tom Cruise has acted. The producers of these movies shouldn't be included in our final result as they did cast Tom Cruise in their movies.

```
SELECT MovieId
FROM Cast
WHERE ActorId = (SELECT Id
                 FROM Actors
                 WHERE FirstName = "Tom"
                 AND SecondName = "Cruise");
```

```
mysql> SELECT MovieId
    -> FROM Cast
    -> WHERE ActorId = (SELECT Id
    ->                  FROM Actors
    ->                  WHERE FirstName = "Tom"
    ->                  AND SecondName = "Cruise");
+---------+
| MovieId |
+---------+
|       3 |
+---------+
1 row in set (0.00 sec)
```

So far, we have been successful in collecting all the movie IDs in which Tom was an actor. Using these IDs we can join on the **Movies** table and find all those producers who **had** Tom in their movies.

```
SELECT DISTINCT Producer
FROM Movies
WHERE Id IN (SELECT MovieId
             FROM Cast
             WHERE ActorId = (SELECT Id
                              FROM Actors
                              WHERE FirstName = "Tom"
                              AND SecondName = "Cruise"));
```

```
mysql> SELECT DISTINCT Producer
    -> FROM Movies
    -> WHERE Id IN (SELECT MovieId
    ->              FROM Cast
    ->              WHERE ActorId = (SELECT Id
    ->                               FROM Actors
    ->                               WHERE FirstName = "Tom"
    ->                               AND SecondName = "Cruise"));
+--------------+
| Producer     |
+--------------+
| J. J. Abrams |
+--------------+
1 row in set (0.01 sec)
```

We may be tempted to add in the **NOT IN** in the outermost select clause to get all the producers who didn't have Tom act in their movies and declare it as the final result, however, that is incorrect. The resulting query will be:

```
SELECT DISTINCT Producer
FROM Movies
WHERE Id NOT IN (SELECT MovieId
             FROM Cast
             WHERE ActorId = (SELECT Id
                              FROM Actors
                              WHERE FirstName = "Tom"
                              AND SecondName = "Cruise"));
```

```
mysql> SELECT DISTINCT Producer
    -> FROM Movies
    -> WHERE Id NOT IN (SELECT MovieId
    ->                  FROM Cast
    ->                  WHERE ActorId = (SELECT Id
    ->                                   FROM Actors
    ->                                   WHERE FirstName = "Tom"
    ->                                   AND SecondName = "Cruise"));
+-----------------+
| Producer        |
+-----------------+
| Kevin Feige     |
| Ryan Seacrest   |
| Chris Hanley    |
| Yash Chopra     |
| Arnon Milchan   |
| Lewis Milestone |
| Jerry Weintraub |
| J. J. Abrams    |
+-----------------+
8 rows in set (0.01 sec)
```

You can observe from the result that J. J. Abrams is also returned in the result set even though he did a movie with Tom. A producer with two movies and Tom as an actor in only one of them, is also returned in the result whereas we want only those producers who never worked with Tom. The fix is to find all those producers which have at least one movie with Tom, which we have already done, and then subtract these producers from the set of all the

producers. The complete query appears below:

```
SELECT DISTINCT Producer
FROM Movies
WHERE Producer
NOT IN (SELECT Producer
        FROM Movies
        WHERE Id IN (SELECT MovieId
                     FROM Cast
                     WHERE ActorId = (SELECT Id
                                      FROM Actors
                                      WHERE FirstName = "Tom"
                                AND SecondName = "Cruise")));
```

```
mysql> SELECT DISTINCT Producer
    -> FROM Movies
    -> WHERE Producer
    -> NOT IN (SELECT Producer
    ->         FROM Movies
    ->         WHERE Id IN (SELECT MovieId
    ->                      FROM Cast
    ->                      WHERE ActorId = (SELECT Id
    ->                                       FROM Actors
    ->                                       WHERE FirstName = "Tom"
    ->                                 AND SecondName = "Cruise")));
+-----------------+
| Producer        |
+-----------------+
| Kevin Feige     |
| Ryan Seacrest   |
| Chris Hanley    |
| Yash Chopra     |
| Arnon Milchan   |
| Lewis Milestone |
| Jerry Weintraub |
+-----------------+
7 rows in set (0.00 sec)
```

We have chosen to use nested queries to get the desired result, however, we could also have used joins.