# Correlated Queries

This lesson discusses the correlated queries.

## Correlated Queries

Correlated queries are a type of nested queries. The distinguishing feature about them is that the inner query references a table or a column from the outer query.

> 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/34lesson.sh** and wait for the MySQL prompt to start-up.

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

-- Query 1
SELECT FirstName
FROM Actors
INNER JOIN DigitalAssets
ON Id = ActorId
WHERE URL LIKE CONCAT("%",FirstName,"%")
AND AssetType="Twitter";

-- Query 2
SELECT FirstName
FROM Actors
WHERE EXISTS (SELECT URL
              FROM DigitalAssets
              WHERE URL LIKE CONCAT("%",FirstName,"%")
              AND AssetType="Twitter");
```

Terminal

1. Let's say we want to know which actors have their first name as part of their Twitter handle. We can use an inner join query as follows:

```
SELECT FirstName
FROM Actors
INNER JOIN DigitalAssets
ON Id = ActorId
WHERE URL LIKE CONCAT("%",FirstName,"%")
AND AssetType="Twitter";
```

```
mysql> SELECT FirstName
    -> FROM Actors
    -> INNER JOIN DigitalAssets
    -> ON Id = ActorId
    -> WHERE URL LIKE CONCAT("%",FirstName,"%")
    -> AND AssetType="Twitter";
+-----------+
| FirstName |
+-----------+
| Jennifer  |
| Kim       |
| Tom       |
+-----------+
3 rows in set (0.00 sec)
```

An alternative is to write a correlated query to glean the same information from the database. Let's see what the query looks like:

```
SELECT FirstName

FROM Actors

WHERE EXISTS (SELECT URL
              FROM DigitalAssets
              WHERE URL LIKE CONCAT("%",FirstName,"%")
              AND AssetType="Twitter");
```

```
mysql> SELECT FirstName
    ->
    -> FROM Actors
    ->
    -> WHERE EXISTS (SELECT URL
    ->                FROM DigitalAssets
    ->                WHERE URL LIKE CONCAT("%",FirstName,"%")
    ->                AND AssetType="Twitter");
+-----------+
| FirstName |
+-----------+
| Jennifer  |
| Tom       |
| Kim       |
+-----------+
3 rows in set (0.00 sec)
```

The inner query references the column FirstName in its WHERE clause even though FirstName is part of the **Actors** table which is referenced only in the outer query. It is legal to access a table or any of its columns referenced in the outer query inside a sub-query. The value of FirstName for each row in the **Actors** table is provided as a scalar value to the inner query.

Pay attention to how we have used the **EXISTS** operator. We want to check for a condition that is true about each actor. We aren't interested in what the inner query returns for each actor, rather only if it returns anything or not. If no result is returned for an actor, then the outer query for that actor evaluates to false and the name isn't printed on the console.