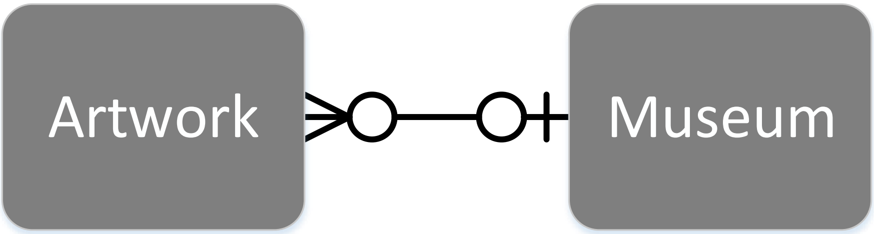
# LEFT and RIGHT JOIN in SQL

## Welcome to this lab activity

In this lab activity, you will explore how to use the RIGHT and LEFT JOIN functions. You will create a new database containing data about museums and art works.

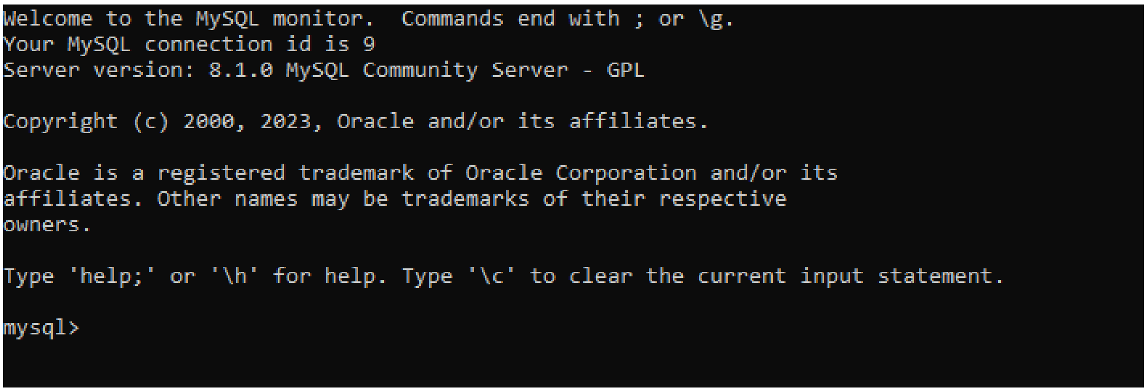
You will create a new database called `myArt` that implements the database schema shown in this diagram:



## Task 1: Start the MySQL interactive shell

Start the MySQL shell, logging in with the root user and password.

When you start the MySQL shell, you should see the MySQL prompt:



```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.1.0 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

## Task 2: Create a new database

1. First, see what databases you already have:

```
SHOW DATABASES;
```

You will already have a couple, including the `myBookshop` database that you created previously.

2. Create a new database called `myArt`:

```
CREATE DATABASE myArt;
```

In the above, ‘`myArt`’ is the name of the new database.

To check if your new database was successfully created, use:

```
SHOW DATABASES;
```

## Task 3: Create tables

Switch to the new database that you created:

```
USE myArt
```

3. Create a table named ‘`artwork`’ based on the database design that you see below. What kind of field types are the best to use for this table?

I'll give you the answer for the first table:

```
CREATE TABLE Museum (
  id        INT           AUTO_INCREMENT,
  name      VARCHAR(50),
  address   VARCHAR(100),
  website   VARCHAR(100),
  PRIMARY KEY(id)
);
```

4. Can you create another table named ‘`Artwork`’ based on the database design that you can see above?

What sort of association would you use to connect `museum` and `artwork` tables?

5. Note that the ‘`artwork`’ table includes a foreign key to the `museum` table. Do you remember how to create a table including foreign keys? Here is my suggestion. Do you agree?

```
CREATE TABLE Artwork (
  id         INT          AUTO_INCREMENT,
  museum_id  INT,
  name       VARCHAR(50),
  artist     VARCHAR(50),
  PRIMARY KEY (id),
  FOREIGN KEY(museum_id) REFERENCES
  Museum(id)
);
```

**6.** See the tables created:

```
SHOW TABLES;
```

**7.** See the fields defined within a given table:

```
DESCRIBE TableName;
```

Replace TableName with the name of your tables.

## Task 4: Input the dummy data

To insert some dummy data into the database, you need to use INSERT INTO statement.

**8.** Insert dummy data into the `museum` table, including the rows that you can see in the database design above.

```
INSERT INTO Museum(name, address, website)
VALUES ('National Gallery','London','www.nationalgallery.org.uk'),
('The Louvre','Paris','www.louvre.fr'),
('The Met','New York',' www.metmuseum.org ');
```

**9.** Insert dummy data into `Artwork`. Here's an example of how you can insert data in a table (Artwork) with a foreign key:

```
INSERT INTO Artwork (museum_id, name, artist)
VALUES (NULL, 'Broken Table', 'Vincent van Gogh');

INSERT INTO Artwork (museum_id, name, artist)
VALUES ((SELECT id FROM Museum WHERE Museum.name='National
Gallery'), 'The Hay Wain', 'John Constable');

INSERT INTO Artwork (museum_id, name, artist)
VALUES ((SELECT id FROM Museum WHERE Museum.name='The Met'), 'The
Dance Class', ' Edgar Degas');
```

## Task 5: Query the data in MySQL shell

Now that you have inserted the data, you can perform SQL queries on it. You may use the wildcard (*) to return all the fields in a table:

```
SELECT * FROM TableName;
```

10. See what data is contained in the ‘`Artwork`’ and ‘`Museum`’ tables.

11. See a list of all the artworks and the museum that they are in. You can achieve this by using a JOIN operation:

```
SELECT Artwork.name, Artwork.artist, Museum.name
FROM Artwork
JOIN Museum
ON Artwork.museum_id = Museum.id;
```

12. Performing a LEFT JOIN operation:

```
SELECT Artwork.name, Artwork.artist, Museum.name
FROM Artwork
LEFT JOIN Museum
ON Artwork.museum_id = Museum.id;
```

13. Performing a RIGHT JOIN operation:

```
SELECT Artwork.name, Artwork.artist, Museum.name
FROM Artwork
RIGHT JOIN Museum
ON Artwork.museum_id = Museum.id;
```

## Task 6: Exit MySQL shell

In your Terminal panel, type the following command:

```
exit
```

## Task 7: Explore further

When tackling these lab activities, it’s always good to stretch yourself by doing some research and attempting some changes on your own.

Add some more artworks then add them to the National Gallery museum. Confirm that the JOIN, LEFT JOIN and RIGHT JOIN operations work as you would expect them to on the new data.

## End of lab

Congratulations on completing this lab.

In the next lab activity, you will practise more database operations and explore how you can use nested select to query your databases.