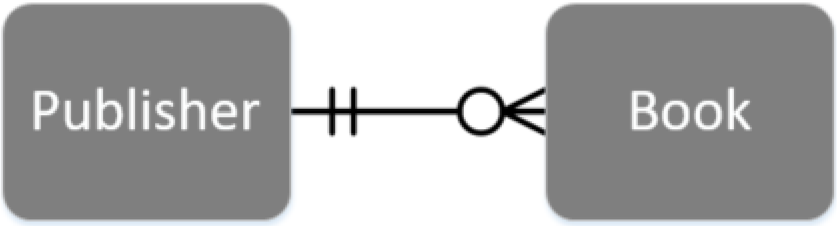
# JOIN in SQL

## Welcome to this lab activity

In this lab activity, you will explore how to add more than one table to your database. Furthermore, you will see how to set up the tables with foreign keys and insert dummy data. Finally, you will execute some basic and advanced SQL statements including JOIN to retrieve details about the schema and the data it contains.

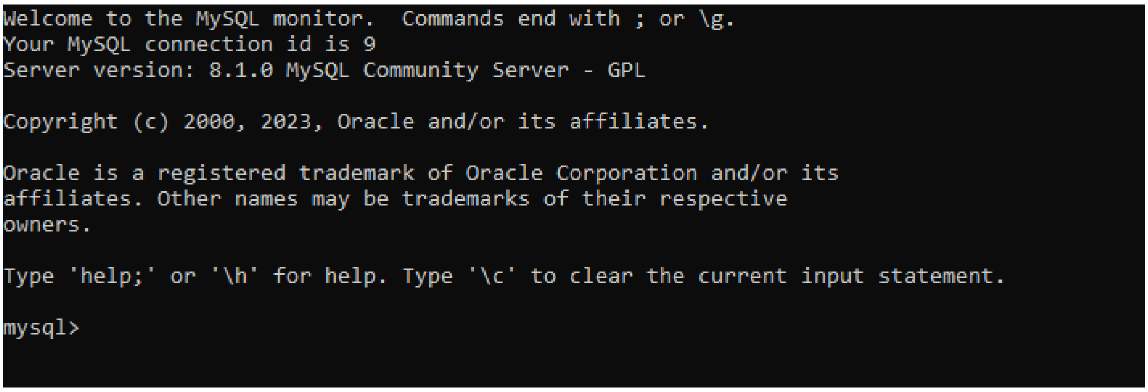
You will create a new database called `myOtherBookshop` that implements the database schema shown in this diagram:



## 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. To create the new database, use:

```
CREATE DATABASE myOtherBookshop;
```

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

3. To check if you successfully created your new database, use:

```
SHOW DATABASES;
```

# Task 3: Create tables

4. Switch to the new database that you created:

```
USE myOtherBookshop;
```

5. Create a table named `Publisher` based on the database design in the ER diagram above. What kind of field types are the best for this table? Here is my suggestion. Do you agree?

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

6. Can you create another table named ‘`Book`’ based on the database design in the diagram above?

What sort of association would you use to connect the `Publisher` and `Book` tables?

7. Note that the `Book` table includes a foreign key to the `Publisher` table. Do you remember how to create a table including foreign keys? Here is my suggestion. Do you agree?

```
CREATE TABLE Book (
  id           INT AUTO_INCREMENT,
  publisher_id INT,
  name         VARCHAR(50),
  price        DECIMAL(5, 2),
  category     VARCHAR(20),
  PRIMARY KEY (id),
  FOREIGN KEY(publisher_id) REFERENCES Publisher(id));
```

**8.** See the tables created:

```
SHOW TABLES;
```

**9.** 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 the `INSERT INTO` statement.

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

```
INSERT INTO Publisher (name, address)
VALUES ('Avon Books', 'Somewhere1');
```

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

```
INSERT INTO Book (publisher_id, name, price, category)
VALUES ((SELECT id FROM Publisher WHERE name = 'Avon Books'), 'Database
Systems', 40.25, 'Computing');
```

Notice that you need to insert the `publisher_id` for Avon Books into the `Book` table. The `SELECT` statement `SELECT id FROM Publisher WHERE name = 'Avon Books'` embedded in the `INSERT` statement retrieves this id form the `Publisher` table so that it can be inserted into the `Book` table.

Here you are inserting a new record for a book named 'Database Systems', with a price of £40.25 published by 'Avon Books'

**12.** Add two more publishers with the following data to your database:

**name:** 'Thames Books'

**address:** 'somewhere2'

**name:** 'Cam Books'

**address:** 'somewhere3'

**13.** Add six more books to your book table with the following data to your database:

**publisher:** ‘Cam Books’

**name:** ‘Node.js for Beginners’

**price:** 25

**category:** Computing

**publisher:** ‘Thames Books’

**name:** ‘Web Architecture’

**price:** 31.99

**category:** Computing

**publisher:** ‘Thames Books’

**name:** ‘Biochemistry’

**price:** 50.40

**category:** Science

**publisher:** ‘Thames Books’

**name:** ‘Physics’

**price:** 12.99

**category:** Science

**publisher:** ‘Cam Books’

**name:** ‘Chemical Engineering’

**price:** 28.50

**category:** Science

**publisher**: ‘Avon Books’

**name:** ‘Zoology’

**price:** 19.99

**category:** Science

## 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;
```

**14.** See what data the `Book` and `Publisher` tables contain.

**15.** List all the books published by ‘Avon Books’ using the following SELECT statement:

```
SELECT Book.name, Book.price, Book.category
FROM Book
JOIN Publisher ON Book.publisher_id = Publisher.id
WHERE Publisher.name = 'Avon Books';
```

Notice how we join the `Book` and `Publisher` table on the publisher `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.

Write a query to find all books and their publishers. Select all the columns from both tables.

Add another publisher, but don’t add a book to it. Run the query that you wrote above again. What do you see? Is it what you expected?

## End of lab

Congratulations on completing this lab.

In the next lab activity, you will explore more database operations.