Introduction to SQL

LESSON

Generate a quiz about this unit

Summarize this unit

Have any doubt about this content? Ask!

Learning Objectives

By the end of this lesson, you will be able to:

- Create a database in two different ways
- Populate a database in two different ways

SQL (Structured Query Languaje)

SQL stands for Structured Query Language. It's a domain-specific language designed for managing and manipulating relational databases. In essence, SQL allows you to interact with databases to store, retrieve, modify, and delete data, among other tasks.

MySQL Workbench

MySQL Workbench is an integrated development environment (IDE) for MySQL databases. It provides a wide array of tools for database developers, administrators, and architects. With a graphical user interface, MySQL Workbench simplifies the process of designing, managing, and maintaining MySQL databases.

Key Features:

1. **SQL Development:** Provides advanced SQL scripting capabilities with syntax highlighting, code completion, and error checking. Users can execute SQL queries, view results, and manage SQL scripts.
2. **Data Modeling:** Offers a visual design tool that allows users to create, modify, and manage database schemas. Users can visually design, model, generate, and manage databases using the Entity-Relationship (ER) diagrams.
3. **Server Administration:** Includes tools for server configuration, user management, backup, and performance monitoring. It allows users to monitor server status, configure server settings, and manage user accounts.

And more...

Using SQL

Using SQL from the terminal

To start the MySQL command-line client, you can use the following command:

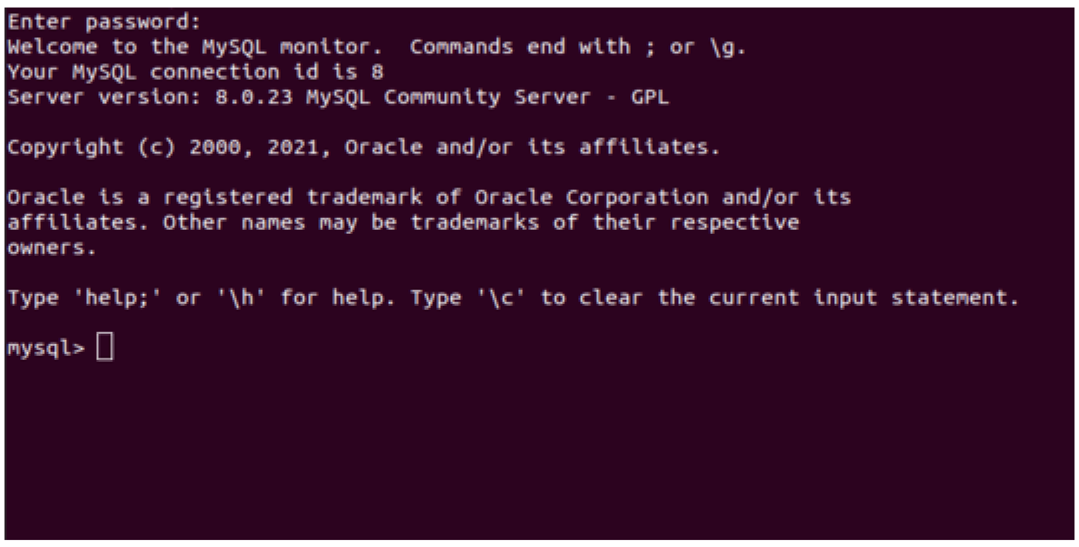
```
mysql -u [username] -p
```

Explain this code

Please replace the placeholder [username] with the appropriate values for your setup.

You'll be prompted to enter the password for the given username.

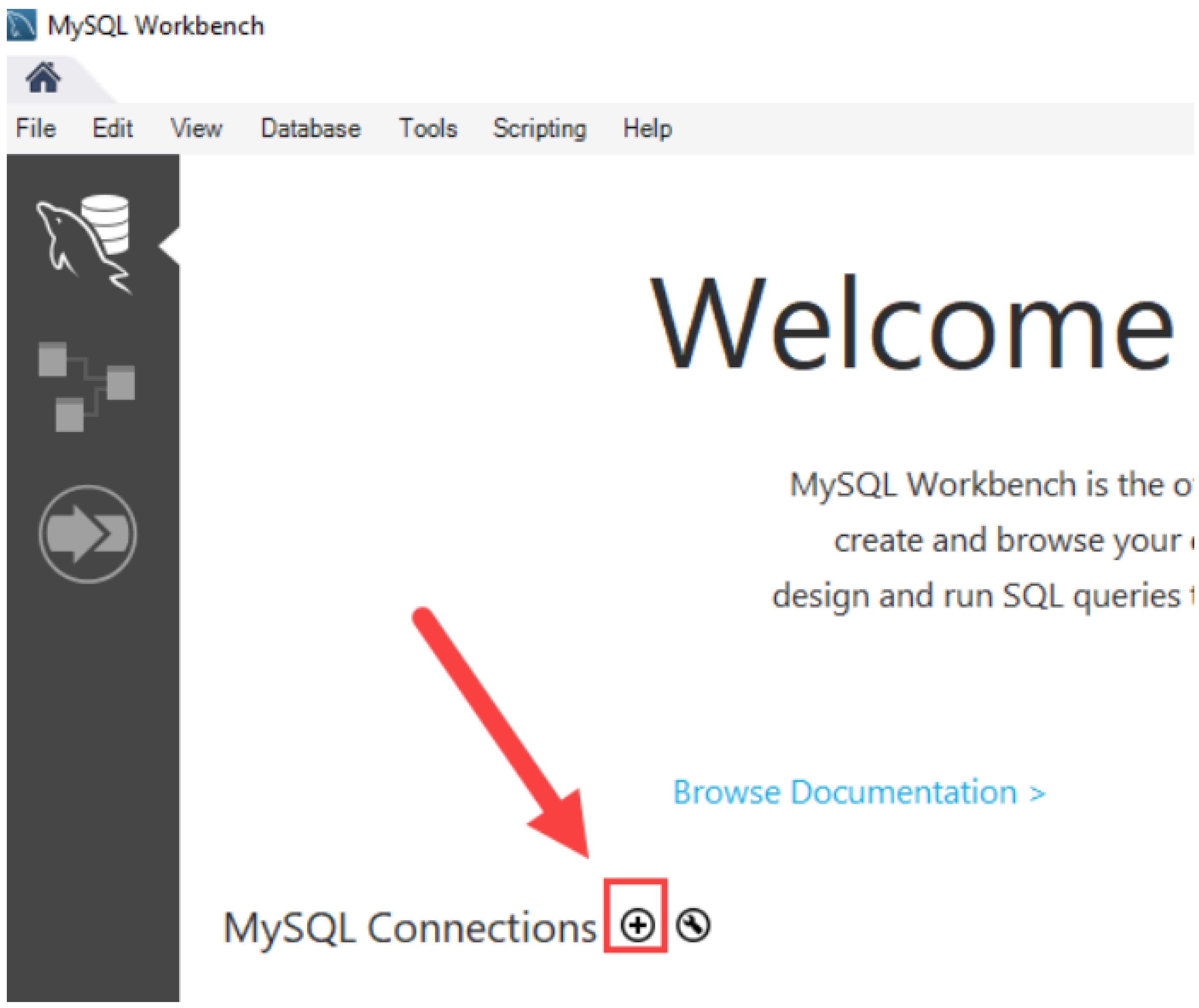
If everything goes well, you'll see the following:



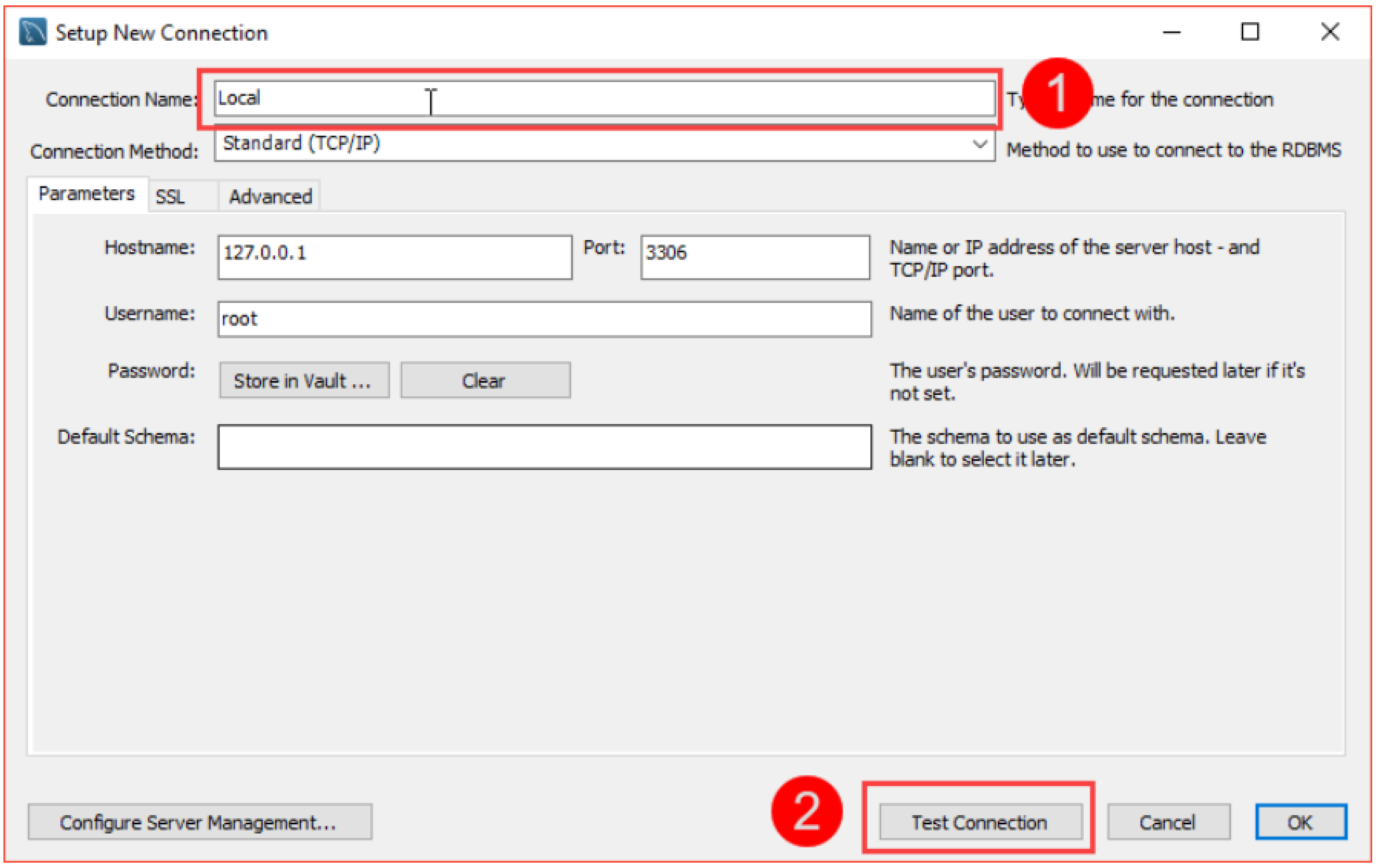
```
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.23 MySQL Community Server - GPL

Copyright (c) 2000, 2021, 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> 
```

Using SQL from MySQL Workbench

First, launch the MySQL Workbench and click the `setup new connection` button as shown in the following screenshot:



Next, type the name for the connection and click the `Test Connection` button.


Setup New Connection

Connection Name: Local — Type a name for the connection

Connection Method: Standard (TCP/IP) — Method to use to connect to the RDBMS

Parameters | SSL | Advanced

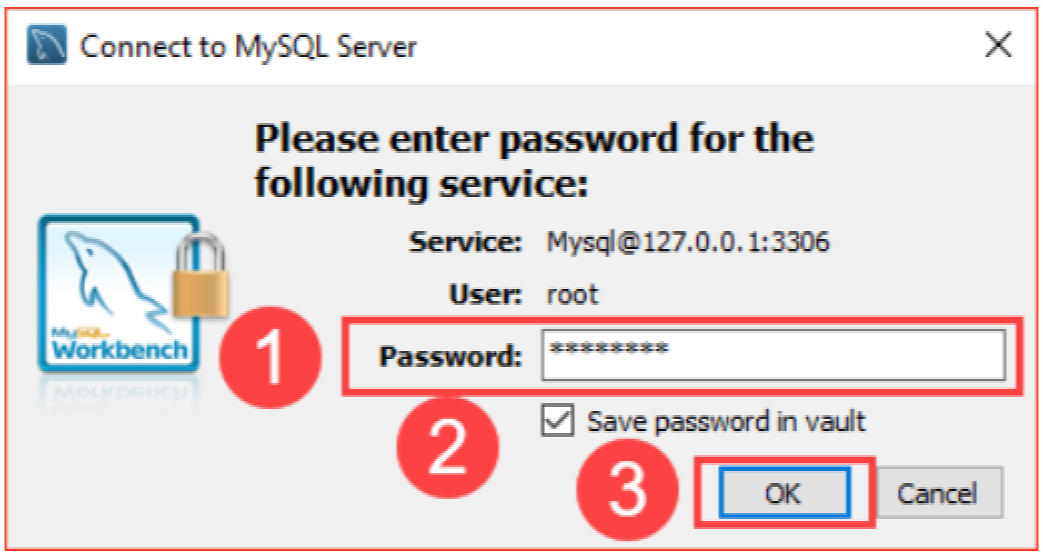
Hostname: 127.0.0.1 Port: 3306 — Name or IP address of the server host - and TCP/IP port.

Username: root — Name of the user to connect with.

Password: Store in Vault ... Clear — The user's password. Will be requested later if it's not set.

Default Schema: — The schema to use as default schema. Leave blank to select it later.

MySQL Workbench displays a dialog asking for the password of the root user:


Connect to MySQL Server

Please enter password for the following service:

Service: Mysql@127.0.0.1:3306

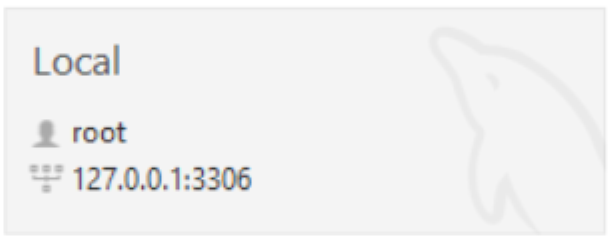
User: root

Password: ********

☑ Save password in vault

You need to (1) type the password for the root user, (2) check the Save password in vault, and (3) click OK button.

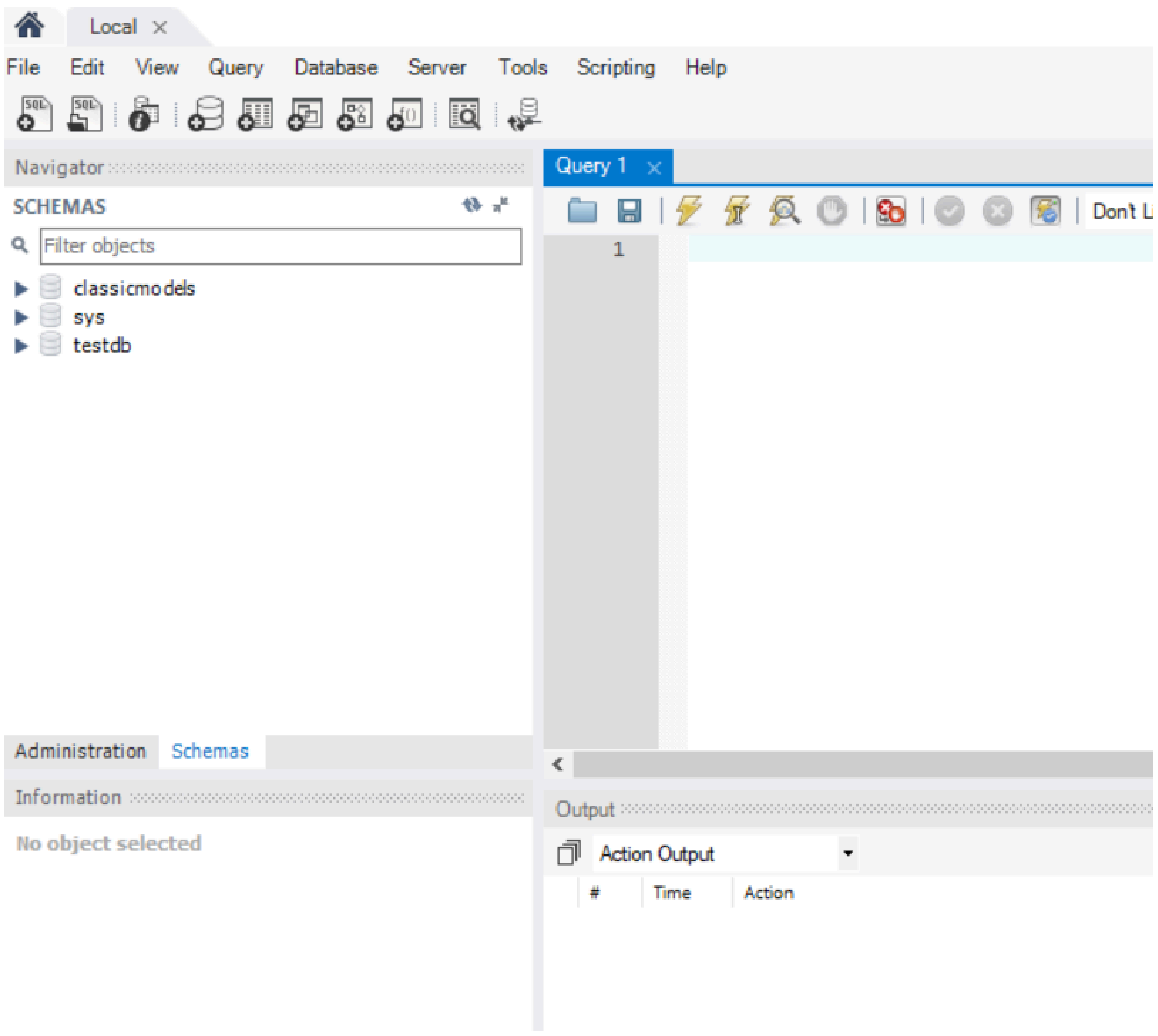
Third, double-click the connection name Local to connect to the MySQL Server.


Local

root

127.0.0.1:3306

MySQL Workbench opens the following window which consists of four parts: Navigator, Query, Information, and Output.



Write the Query

To write your SQL Query, in the SQL tab you'll see a large white area where you can type your SQL queries. Write or paste your SQL query into this space.

Execute the Query

- **To Execute the Entire Query**: Click on the lightning bolt icon located in the toolbar above the query editor. This will execute all the SQL statements written in the tab.
- **To Execute a Specific Part of the Query**: Highlight the specific SQL statement or part of the query you want to execute, then click on the lightning bolt icon.

View the Results

- After executing the query, the results will be displayed in the panel below the query editor.
- For `SELECT` statements, the resulting data will be shown in a grid format.
- For other types of statements (like `INSERT`, `UPDATE`, or `DELETE`), you'll see a message indicating how many rows were affected.

Inspect Errors (if any)

- If there's an error in your SQL statement, MySQL Workbench will highlight the problematic portion of your query and display an error message.
- The error message will give you information about what went wrong, which can help in debugging the SQL statement.

SQL Sublanguages

SQL is categorized into several sublanguages based on the type of operations they perform. These sublanguages allow for a wide range of tasks, from defining structures to modifying data. The primary sublanguages are DDL, DML, DCL, and TCL.

1. DDL (Data Definition Language)

DDL deals with the structure or schema of the database. It provides commands to create, modify, or delete database objects such as tables, indexes, and constraints.

Common DDL Commands:

- **CREATE**: Used to create objects like tables, views, or indexes.

```
CREATE TABLE users (id INT, name VARCHAR(50));
```

Copy

Explain this code

- **ALTER**: Modifies existing database objects. For example, it can be used to add, delete, or modify columns in an existing table.

```
ALTER TABLE users ADD email VARCHAR(100);
```

Copy

Explain this code

- **DROP**: Deletes an existing database object.

```
DROP TABLE users;
```

Copy

Explain this code

- **TRUNCATE**: Deletes all records from a table without removing the table structure itself.

```
TRUNCATE TABLE users;
```

Copy

Explain this code

- **RENAME**: Renames database objects, such as tables.

```
RENAME TABLE old_table_name TO new_table_name;
```

Copy

Explain this code

2. DML (Data Manipulation Language)

DML deals with the manipulation of data stored in the database. It allows for the insertion, retrieval, modification, and deletion of data.

Common DML Commands:

- **SELECT**: Retrieves data from one or more tables.

```
SELECT name, email FROM users WHERE id = 5;
```

Copy

Explain this code

- **INSERT**: Adds new records into a table.

```
INSERT INTO users (name, email) VALUES ('John Doe', 'john@example.com');
```

Copy

Explain this code

- **UPDATE**: Modifies existing records in a table.

```
UPDATE users SET email = 'john.doe@example.com' WHERE id = 5;
```

Copy

Explain this code

- **DELETE**: Removes records from a table.

```sql
DELETE FROM users WHERE id = 5;
```

3. DCL (Data Control Language)

DCL deals with permissions and access control. It allows database administrators to grant or revoke access rights and privileges to or from database users.

We won't be using these in our lectures.

Common DCL Commands:

- **GRANT**: Provides specific privileges to users or roles.
- **REVOKE**: Removes specific privileges from users or roles.

4. TCL (Transaction Control Language)

TCL deals with the management of transactions within the database. It ensures that multiple operations are treated as a single unit of work, maintaining data integrity.

We won't be using these in our lectures.

Common TCL Commands:

- **COMMIT**: Saves all the modifications made during the current transaction.
- **ROLLBACK**: Undoes any changes made during the current transaction.

And more...

More on CREATE, INSERT INTO, DELETE, UPDATE

Create a database

To create a new database in MySQL, you use the CREATE DATABASE statement with the following syntax:

```sql
CREATE DATABASE IF NOT EXISTS `[my_db_name]`;
USE `[my_db_name]`;
```

Analogously, if we want to delete a database

```sql
DROP database `[my_db_name]`;
```

In this syntax:

- Specify in the name of the database after the `CREATE DATABASE` keywords (in [my_db_name]). The database name must be unique within a MySQL server instance. If you attempt to create a database with a name that already exists, MySQL will issue an error.
- Use the `IF NOT EXISTS` option to conditionally create a database if it doesn't exist.
- Specify the character set and collation for the new database. If you skip the `CHARACTER SET` and `COLLATE` clauses, MySQL will the default character set and collation for the new database.

Create tables

We specify the table's name, the variables and their data type.

Also the primary key and we set up the union between the company table and stocks table: symbol is the union field.

Copy

```sql
create table `stocks`(
`stock_id` int,
`date` date,
`open` float,
`high` float,
`low` float,
`close` float,
`volume` float,
`symbol` char(4) not null,
PRIMARY KEY (`stock_id`),
CONSTRAINT `stock_ibfk_1` FOREIGN KEY (`symbol`) REFERENCES `company` (`symbol`)
)
```

Explain this code

Populate a database - INSERT INTO command

The following `INSERT` statement allows you to insert multiple rows into the books table:

Copy

```sql
INSERT INTO company(symbol, company_name, sec_fillings, sector, sub_industry, Headquarters,        ,
VALUES('MMM', '3M', 'reports', 'Industrials', 'Industrial Conglomerates', 'Saint Paul, Minnesota', '1976
      ('MSFT', 'Microsoft', 'reports', 'Software', 'Software', 'Saint Paul, Minnesota', '1976-09-08', '6
```

Explain this code

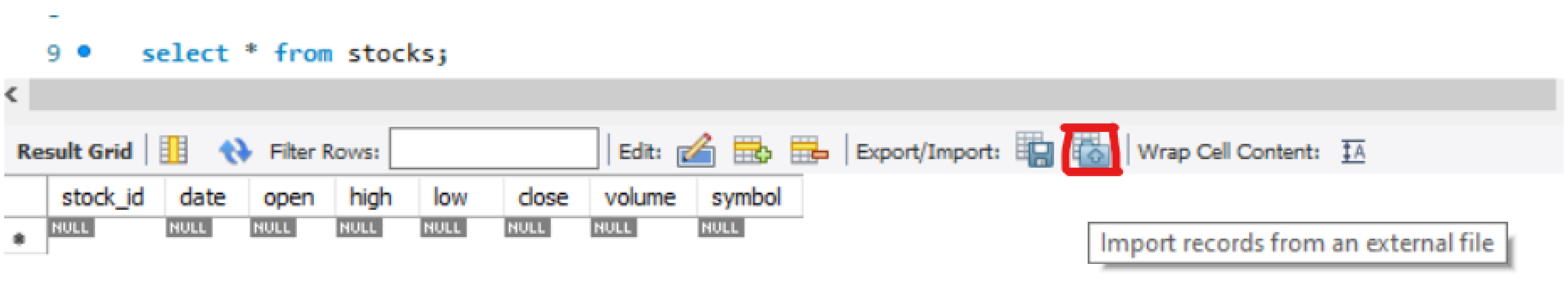
Note: If we are adding values for all the columns of the table, we do not need to specify the names of the columns in the SQL query. However, make sure that the order of the values is in the same order as the columns in the table.

Populate database massively

As we can see with the `INSERT INTO` command we insert row by row. However, sometimes this is not the best way. There are two other ways of doing it massively.

Insert data massively with the assistant

Click on this button and follow the instructions of the assistant. It could take several minutes



```sql
select * from stocks;
```

stock_id	date	open	high	low	close	volume	symbol
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Updating data

We use the `UPDATE` statement to modify the values of existing records in a table.

Copy

```sql
UPDATE table_name
SET column1 = value1, column2 = value2, ... -- specifies the changes we want to make
WHERE condition; -- defines which records we want to modify.
```

Explain this code

NOTE: We need to be careful when updating records in a table. If we omit the `WHERE` clause, all records in the table will be updated."

Deleting data

The `DELETE` statement is used to remove existing records from a table.

Copy

```sql
DELETE FROM table_name  -- specifies the table from which we want to delete data
WHERE condition; -- matches the records based on our condition
```

Explain this code

⚠️ ⚠️ ⚠️ Caution!!! If we don't include the `WHERE` clause, we will delete the entire table ⚠️ ⚠️ ⚠️

Importing a SQL file

To import a SQL file into a MySQL database, you can use both the terminal (command-line interface) and MySQL Workbench. Here's how you can do it using each method:

1. Using Terminal:

1. **Using the `mysql` Command:**

   You can use the `mysql` command to import an SQL file directly into a database. Replace `[username]`, `[database_name]`, and `[path_to_sql_file]` with your actual MySQL username, the target database name, and the path to your SQL file, respectively.

   ```
   mysql -u [username] -p [database_name] < [path_to_sql_file]
   ```

   Copy

   Explain this code

   After running the above command, you'll be prompted to enter the password for the given username. Once entered, the SQL file will start importing.

2. Using MySQL Workbench:

1. **Open MySQL Workbench** and establish a connection to your database server by double-clicking on the connection or creating a new one.
2. **Open the SQL Script File:**
   - Go to `File` in the top menu and select `Open SQL Script...`
   - Navigate to your SQL file and open it. The content of the SQL file will be displayed in a new query tab.
3. **Run the Script:**
   - Ensure that the target database (the one you want to import into) is selected from the dropdown list of schemas on the left side.
   - Click on the lightning bolt icon (or `Query` → `Execute (All or Selection)`) to run the script. This will execute all the SQL statements in the opened file.
4. Monitor the **Action Output** panel at the bottom of the Workbench window to see the progress and ensure there are no errors.

Summary of basic SQL queries

Delete the database if it exists:

```sql
drop database if exists db_name;
```

Copy

Explain this code

Create a database:

```sql
create database db_name;
```

Copy

Explain this code

Access the database:

```sql
use db_name;
```

Copy

Explain this code

Create a table if it doesn't exist:

```sql
create table if not exists table_name (
  id int primary key,
  attribute_1 varchar(40) not null,
  attribute_2 varchar(3) not null,
  attribute_3 int,
  attribute_4 date,
  attribute_5 boolean
);
```

Copy

Explain this code

Create relationships:

```sql
alter table table_a
add foreign key(table_b_column)
references table_b(table_b_id)
on delete set null;
```

Copy

Explain this code

Insert data:

```sql
insert into table_name (id, column_a, column_b,..)
values (1, 'A',  3.14,..);
```

Copy

Explain this code

Update data:

```sql
update table_name
set column = 'value'
where id = 1;
```

Copy

Explain this code

Delete data:

```sql
delete from table_name where column = 'value';
```

Copy

Explain this code

Bonus: ERDs and MySQL

In MySQL Workbench, the ERD functionality is part of the broader data modeling toolset. An ERD in MySQL Workbench visually represents the structure of your database: tables, relationships, and the fields within each table.

Key Features:

- **Visual Database Design**: You can drag and drop database elements onto the canvas to create a visual representation of your database.

- **Forward Engineering**: Convert your ERD into SQL scripts to create the database schema.
- **Reverse Engineering**: Generate an ERD from an existing MySQL database schema. This is especially useful if you inherit a database and want to visualize its structure.
- **Synchronization**: Synchronize your ERD with your database, so changes in one are reflected in the other.

Creating an ERD in MySQL Workbench:

1. **Launch MySQL Workbench** and open a new model by clicking on the "+" icon in the toolbar or selecting `File > New Model`.
2. In the model editor, right-click on the canvas and choose "Add Diagram" to create a new ERD.
3. Use the toolbox on the left side to drag tables, views, and other database elements onto the canvas.
4. Double-click on any element to edit its properties. For a table, this lets you add columns, set data types, define primary/foreign keys, and more.
5. To define relationships, use the relationship tools in the toolbox. Drag from one table to another to define a relationship, then adjust its properties (e.g., one-to-one, one-to-many).

Reverse Engineering with MySQL Workbench:

1. Connect to a MySQL server by setting up a connection in the home screen of MySQL Workbench.
2. Once connected, choose `Database > Reverse Engineer` from the top menu.
3. Follow the wizard steps. MySQL Workbench will analyze the connected database, retrieve its schema, and generate an ERD for you.

PREVIOUS
← Introduction to Databases

NEXT
Lab | MySQL Database Creation →