Overview

In this module, we'll be using:

MySQL

MySQL Workbench

In the next lesson, **Database concepts**, we'll go through the **steps of downloading**, **installing**, and **launching** MySQL Workbench and MySQL Community Server on **Windows** and **Mac**.

The various **CSVs**, **SQL files**, and **queries** that we'll need to create and transform our database are provided with this document. However, they can also be **downloaded from the walk-throughs** at the point we will need them.

Throughout the module, we will need to use the provided CSVs, SQL files, and queries to create or recreate our database. We will look at **how we use** these files and queries.

Overview

In this module, we'll also be using:

In the **SQL in production** lesson, we'll introduce Jupyter Notebooks and go through the **steps of downloading**, **installing**, and **launching** Anaconda and Jupyter Notebooks on **Windows** and **Mac**.

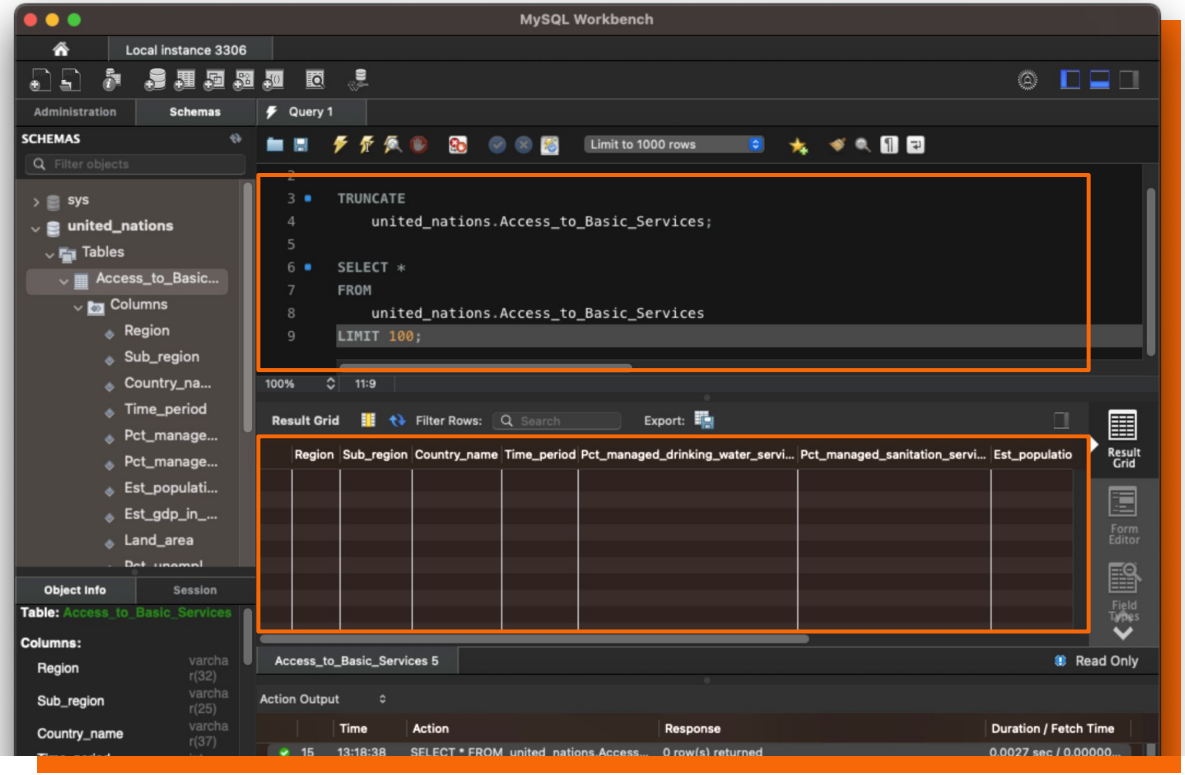
Included in the lessons beyond this point are **[Walk-through]s** in which queries are run in **MySQL Workbench,** as well as the accompanying **[Notebook]s** in which we can execute the same queries in **Jupyter Notebooks.**

Throughout the module, we will be able to interact with and manipulate our database **the same way** using either **MySQL Workbench or Jupyter Notebooks.**

The effect of database transformations

For example, we will learn how to remove all data from a table in SQL.

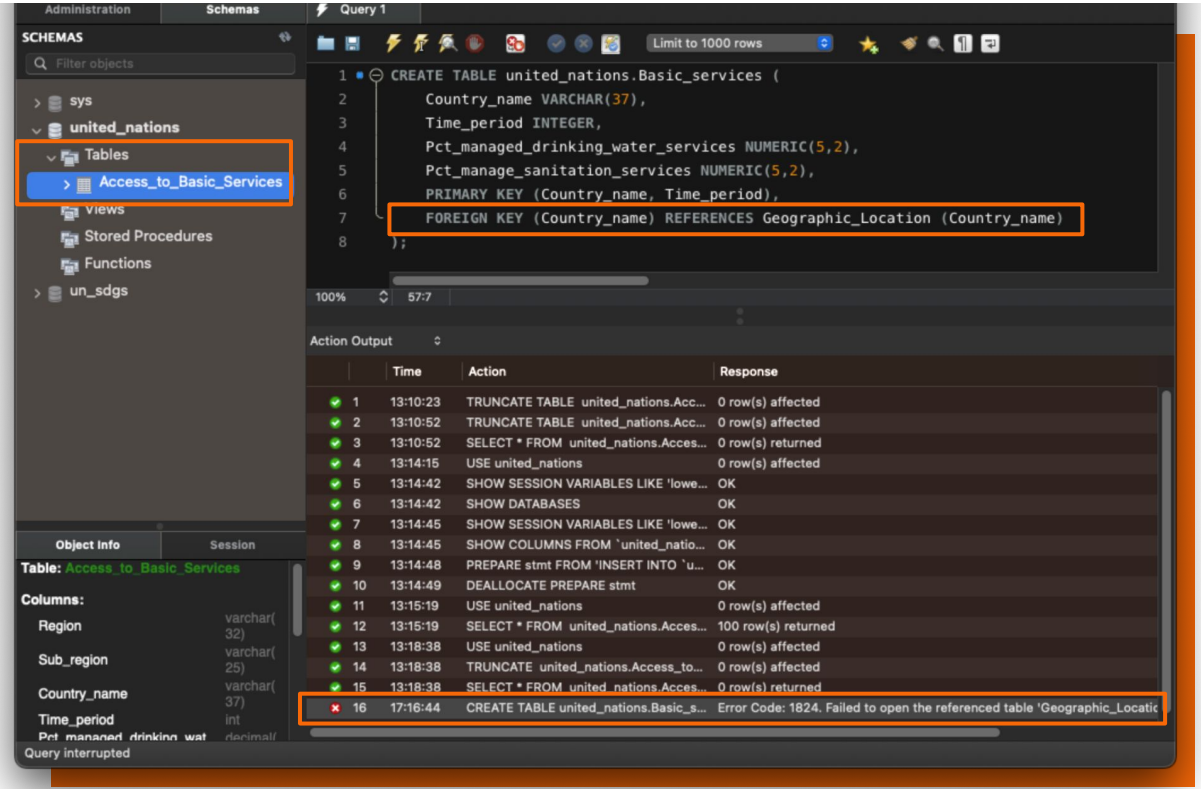
Once we've **removed the data** from the table, any query thereafter will **return an empty table**. In these cases, we'll be prompted with the necessary steps to recreate the database. It's crucial to **follow these instructions closely** so that future queries work.



The effect of database transformations

For example, we want to divide a larger dataset into smaller tables and link them using relationships.

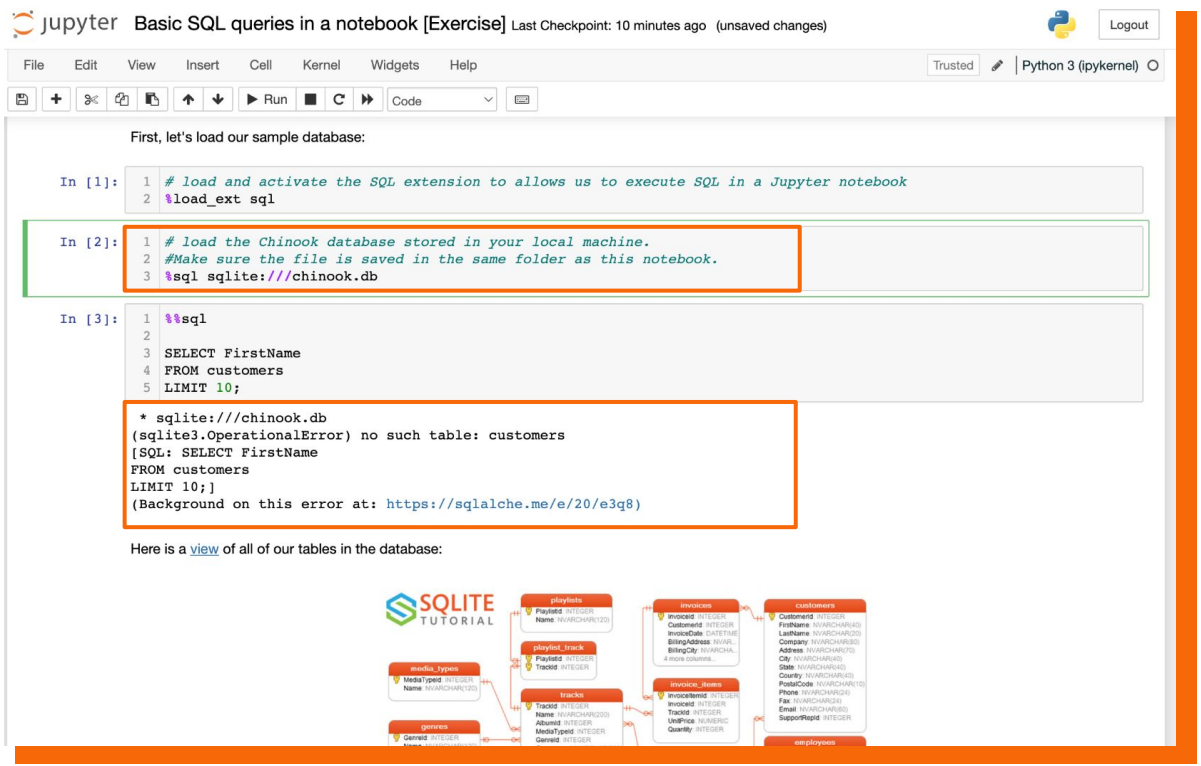
The **walk-throughs often build on one another**, so if we don't complete them in order, we may end up in a situation like this:
We created the **`Geographic_location` table in a previous walk-through**, and therefore, if we **skipped** it, we would get an **error**.



Using notebook files and dependencies

We will receive database files that we need to query in a notebook.

The notebook environment will **look for the database file in the current folder***. If the file is not in the same folder as the notebook, the system won't be able to locate it and we will get a "no such table" or similar error message. We need to **make sure we have all the notebook dependencies saved in the same folder as the notebook**.



Using notebook files and dependencies

It is possible to specify the correct file path in our code if the file is not saved in the same folder.

Should the **database be saved in a different folder** we will need to use the absolute or relative file path to locate the folder.

- We use three slashes `sqlite:///` in order to use a relative path.
- We use four slashes `sqlite:////` in order to use an absolute path.