* SQL code resources - <https://www.w3schools.com/sql/>
* Method 1: SQL Databases and Python (How to connect SQL to Python) - <https://learnpython.com/blog/python-libraries-for-sql/#:~:text=SQL%20stands%20for%20Structured%20Query,from%20within%20your%20Python%20script>
  + MySQL is an open-source relational database management system that’s widely used by many companies and web applications. It is written in C and C++ and typically runs on an external server dedicated to storing the data. This means you need to connect to the server to access the data from your local machine by writing SQL queries. This can all be done from within Python.
    - Ex.) pip install mysql-connector-python
  + Using the connect() function allows you to create a database in your environment. Now you can use Python code to connect to the external server and access the data. You’ll need to have a username and password already set up and know the IP address of the server.
    - >>> import mysql.connector
    - >>> db = mysql.connector.connect(user='julia',
      * password='supertopsecretpassword',
      * host='111.1.1.1',
      * database='employees')
  + To access a table in the database, you need to create a cursor object to help navigate through the database:
    - >>> cur = db.cursor()
  + You can now execute SQL queries using this cursor object. For example, you can use the same query we saw in the previous section:
    - >>> cur.execute('select name, age from employee\_data')
  + Use the execute method to run any SQL command here. You can also use it to create new tables and manipulate them. To view the data in the cursor object, you can use a loop to print out the rows:
    - >>> for data in cur:
    - ...     print(data)
    - ('andrea', 32)
    - ('phillip', 25)
* Method 2: SQL Databases and Python (MySQL Workbench) (How to connect SQL to Python) - <https://towardsdatascience.com/starting-with-sql-in-python-948e529586f2#:~:text=A%20quick%20and%20easy%20way,engine%20in%20the%20test%20environment>.
  + A quick and easy way to be able to run SQL queries with Python is using [SQLite](https://www.sqlite.org/index.html). SQLite is a library that utilizes an SQL database engine. It performs relatively fast and has been proven to be highly reliable. [SQLite is the most commonly used database engine in the test environment.](https://pynative.com/python-sqlite/)
    - import sqlite3  
      import pandas as pd
  + Using the connect() function allows you to create a database in your environment.
    - Ex.) sql\_connect = sqlite3.connect('factbook.db') (For this example, I will be referencing a [CIA database](https://www.cia.gov/library/publications/the-world-factbook/) called “factbook.db”.)
  + The cursor() function is used to assist with executing our SQL queries.
    - Ex.) cursor = sql\_connect.cursor()
  + Save SQL Query as a string
    - Ex.) query = "SELECT \* FROM factbook;"
  + Execute the query using the cursor variable from earlier. This will convert the results to tuples and store it as a local variable. To get all the results we use fetchall().
    - Ex.) results = cursor.execute(query).fetchall()
  + To run the query, we saved early with pandas we do the following.
    - Ex.) pd.read\_sql\_query(query,sql\_connect)
  + Remember it is important to close your connection when you are finished. Closing the connection will grant others access to the database.
    - Ex.) sql\_connect.close()
* What environment to code SQL databases
  + MySQL Workbench - <https://dev.mysql.com/downloads/workbench/>
  + sqlite3
  + mysql.connector
* NoSQL and Python
  + Method 1: MongoDB
    - Examples
    - <https://www.tutorialspoint.com/python_data_science/python_nosql_databases.htm#:~:text=As%20more%20and%20more%20data,is%20interacts%20with%20Relational%20databases>.
    - <https://realpython.com/introduction-to-mongodb-and-python/>