**Course Project**

**DeVry University**

**College of Engineering and Information Sciences**

**Course Number: CEIS110**

# Module 5: Querying and Manipulating Data with SQL and Python

# Objectives

* To retrieve data from a database with an embedded SQL query in Python
* To cleanse data and manipulate data from a database in Python
* To save data as a CSV file
* To open data in Excel and create a chart

# Parts List

Equipment:

* PC running Python

# Introduction

**Lists**

One of the most common tasks that a programmer has to do is keep track of a list of objects. We can all think of examples of lists to be maintained on the computer—lists of names, lists of telephone numbers, lists of identification numbers, and so on—and we can use an array to store the list of data items. The array is a simple mechanism used to store a list within a program and allow easy access to any element in the list. The data stored in a database table can be read into a list and processed by Python. The data from the database must be cleansed with spurious or missing data removed. Data cleansing is often performed by companies to ensure the data is complete, valid, and standardized.

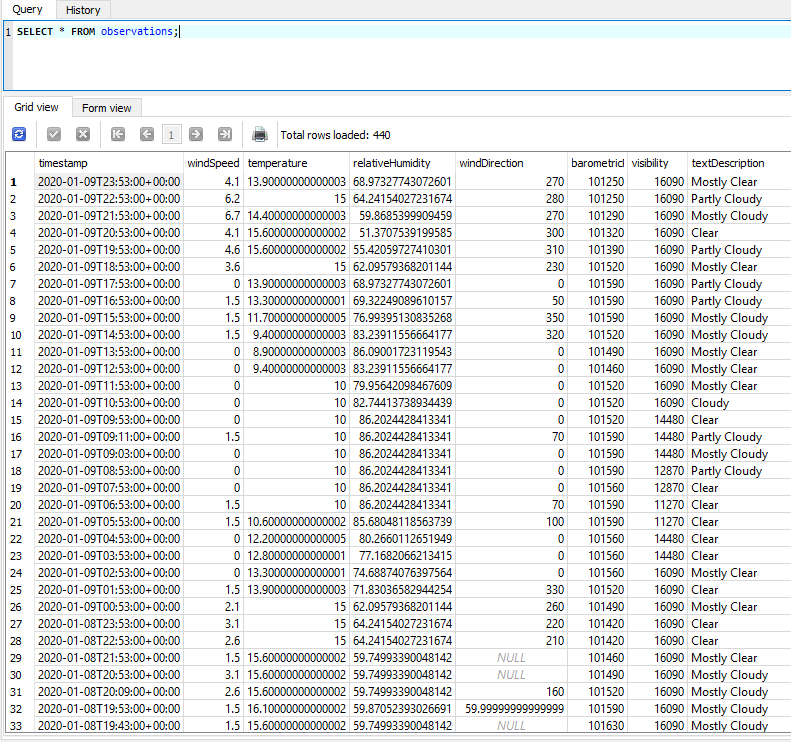


Figure 1: Data from observations table in weather database

Your data may look different than the above figure. The data needs to be read into lists, then cleansed, sorted, and written out to a file in a format that can be easily read by Excel or other data analysis programs.

The database may contain some missing or otherwise invalid data, indicated by the word *NULL* in the query results. These missing or NULL values can be removed by Python so that Excel will be able to work with the resulting file.

The following code retrieves only the temperature and relative humidity values from the database, and writes them to a file in comma-separated values (CSV) format that can be read by Excel. Review this code and note the following:

* It includes a SQL SELECT query, similar to the ones you used in SQLiteStudio in the previous step, to get the temperature and humidity data. The ORDER BY clause ensures data are returned in data and time order.
* To create a smaller data set for easier analysis, only the first half of the rows retrieved from the database are processed; this is done by slicing the list of rows.
* The program transforms the data by converting the temperature from degrees Celsius to degrees Fahrenheit.
* It also cleanses the data by using if-else structures to ensure missing or invalid values (indicated by NULL values in the database and by None in Python) are not written to the file.

***#Purpose: Extract temperature, humidity data from weather database into CSV file***

***#Name: Your name***

***#Date: Your date***

***# Run BuildWeatherDB.py to build weather database before running this program***

***import sqlite3***

***#convert Celsius temperature to Fahrenheit***

***def convertCtoF(tempC):***

***return (tempC\*9.0/5.0) + 32.0***

***#file names for database and output file***

***dbFile = "weather.db"***

***output\_file\_name='formatdata.csv'***

***#connect to and query weather database and***

***dbFile = "weather.db"***

***conn = sqlite3.connect(dbFile)***

***#create cursor to execute SQL commands***

***cur = conn.cursor()***

***selectCmd = """ SELECT temperature, relativeHumidity FROM observations***

***ORDER BY timestamp; """***

***cur.execute(selectCmd)***

***allRows = cur.fetchall()***

***#limit the number of rows output to half***

***rowCount = len(allRows)//2 # double slash does integer division***

***rows = allRows[:rowCount]***

***#write data to output file***

***with open(output\_file\_name,"w+") as outf:***

***outf.write('Celsius,Fahrenheit,Humidity')***

***outf.write('\n')***

***for row in rows:***

***tempC = row[0]***

***if tempC is None: #handle missing temperature value***

***outf.write(',,')***

***else:***

***tempF = convertCtoF(tempC)***

***outf.write(str(tempC)+',')***

***outf.write(str(tempF)+',')***

***humidity = row[1]***

***if humidity is None: #handle missing humidity value***

***outf.write('\n')***

***else:***

***outf.write(str(humidity)+'\n') #print data to file separated by commas***

Copy and paste this code into Spyder, put your own name and the dates in the comments, and save it into your CEIS110 folder as ExtractTempHumidity.py. To work, the program must be saved into the same folder where your weather.db database file is located.

Run the program. It will create an output file named formatdata.csv in the same folder as the program file, containing the extracted, transformed, and cleansed temperature and humidity data.

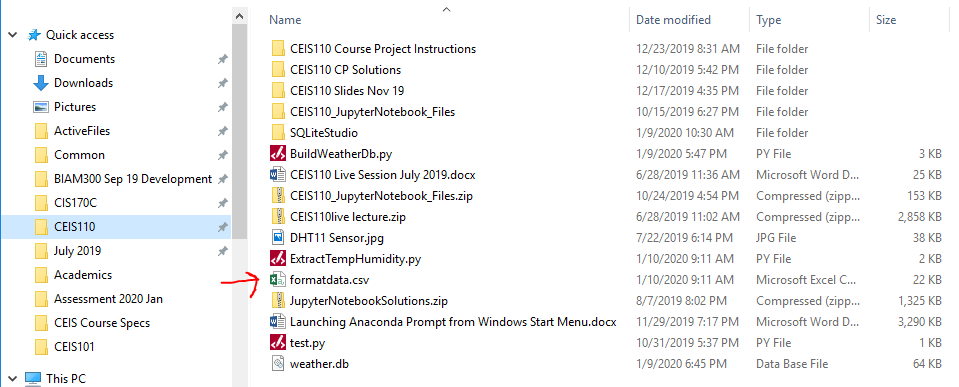


Figure 2: Formatdata.csv in Windows Explorer

Double click on the formatdata.csv file. It should open in Excel. Choose **File > Save as** and be sure to choose the .xlsx file type. Also be sure you remember where it is stored or ensure that it is saved in the same directory as the formatdata.csv.

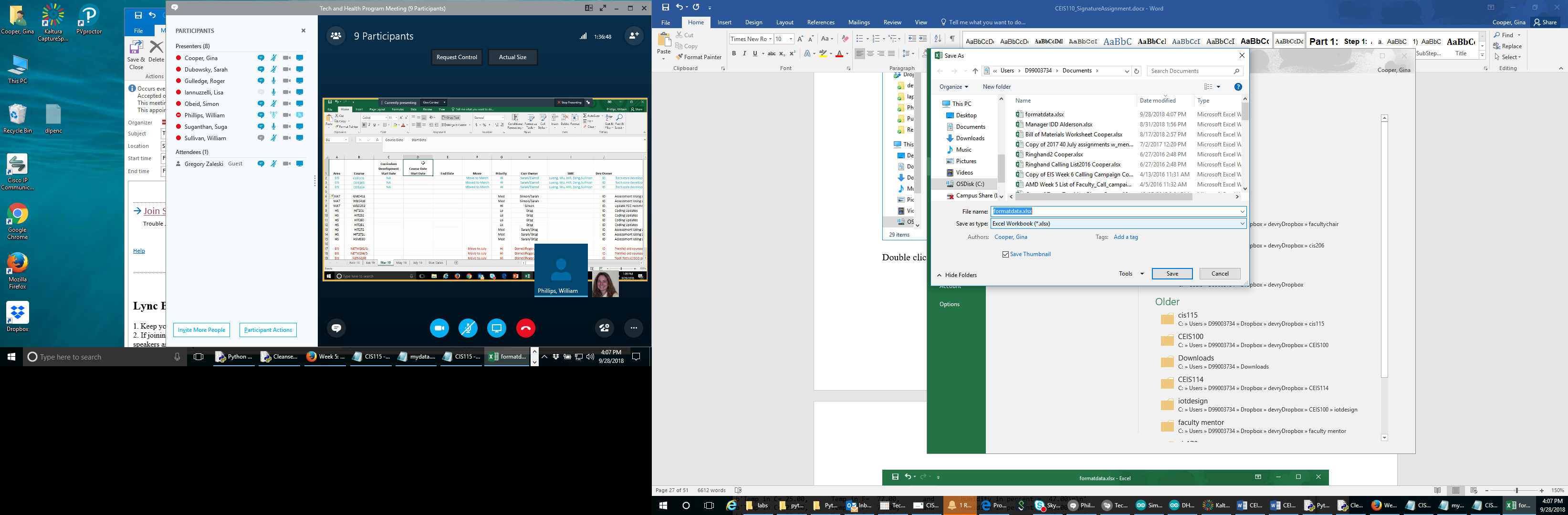


Figure 3: Save file as Excel workbook

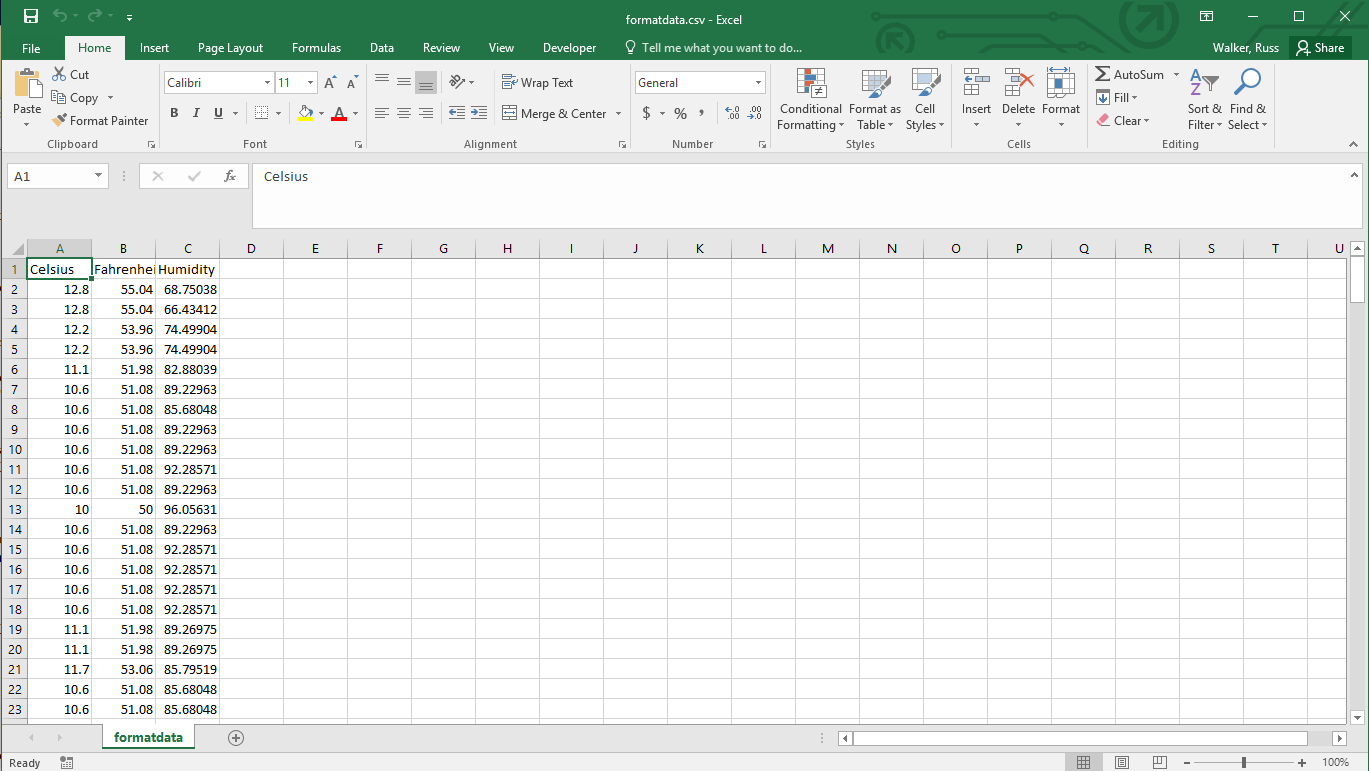
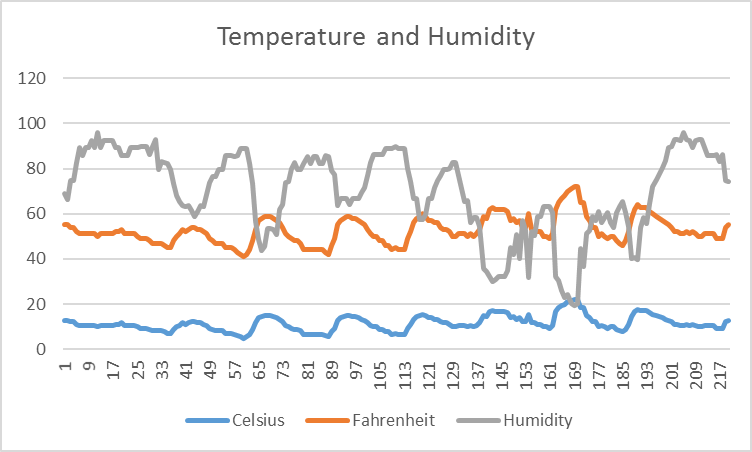


Figure 4: Open formatdata file in Excel

Select all data (it may be from A1: C221--the number of rows in your data set may be different) and create a line chart with a Chart Title of “Temperature and Humidity” Go to Insert > Line chart and select the first option for a 2-D line chart. Change the name of the chart to “Temperature and Humidity” by clicking on the Chart Title.



This data represents the first half of the rows in your weather data set (probably about one week of weather data). To get a second file containing the other half of the data (for the following week), make the changes to the code that are highlighted below, and run the program again to create a second output file named formatdata2.csv. You can also create another line chart for this data and compare them.

***#Purpose: Extract temperature, humidity data from weather database into CSV file***

***#Name: Your name***

***#Date: Your date***

***# Run BuildWeatherDB.py to build weather database before running this program***

***import sqlite3***

***#convert Celsius temperature to Fahrenheit***

***def convertCtoF(tempC):***

***return (tempC\*9.0/5.0) + 32.0***

***#file names for database and output file***

***dbFile = "weather.db"***

***output\_file\_name='formatdata2.csv' #add 2 to file name for 2nd data set***

***#connect to and query weather database and***

***dbFile = "weather.db"***

***conn = sqlite3.connect(dbFile)***

***#create cursor to execute SQL commands***

***cur = conn.cursor()***

***selectCmd = """ SELECT temperature, relativeHumidity FROM observations***

***ORDER BY timestamp; """***

***cur.execute(selectCmd)***

***allRows = cur.fetchall()***

***#limit the number of rows output to half***

***rowCount = len(allRows)//2 # double slash does integer division***

***rows = allRows[rowCount:] #use [rowCount:] instead of [:rowCount] for 2nd data set***

***#write data to output file***

***with open(output\_file\_name,"w+") as outf:***

***outf.write('Celsius,Fahrenheit,Humidity')***

***outf.write('\n')***

***for row in rows:***

***tempC = row[0]***

***if tempC is None: #handle missing temperature value***

***outf.write(',,')***

***else:***

***tempF = convertCtoF(tempC)***

***outf.write(str(tempC)+',')***

***outf.write(str(tempF)+',')***

***humidity = row[1]***

***if humidity is None: #handle missing humidity value***

***outf.write('\n')***

***else:***

***outf.write(str(humidity)+'\n') #print data to file separated by commas***

# Deliverables Module 5

* Complete the Course Project Presentation Deliverable
* Include a screenshot of your Python code with your name and date in the comments
* Include screenshots of your Excel spreadsheet showing the data and the chart