COSC 311: Introduction to Data Visualization and Interpretation

(1) Python + SQLite3

Dr. Shuangquan (Peter) Wang
(spwang@salisbury.edu)

Department of Computer Science

Salisbury University

About this note

- The contents of this note refer to:
  - Prof. Fang Li's Teaching Materials (Shanghai Jiao Tong University)
    - https://www.cs.sjtu.edu.cn/~li-fang/DB.htm
    - **The database access part refers to Weiming BAO’s slides (revised a little)**

Install DB Browser

- Go to https://sqlitebrowser.org/ (or https://sqlitebrowser.org/dl/)
- Click download
- Find the file for your OS and download
- Install it

Download chinook & iris databases

- Chinook database
  - Go to https://www.sqlitetutorial.net/sqlite-sample-database/
  - Find the download place (https://www.sqlitetutorial.net/wp-content/uploads/2018/03/chinook.zip) and download the package
  - Unzip it, you will have "chinook.db"
- Iris database
  - Go to https://github.com/Xorlev/csci568/blob/master/project05/iris.sqlite3.db
  - Download the iris database "iris.sqlite3.db"

Use SQL to query the database

- Launch DB Browser
- Open the database
- Check tables, data
- Write SQL statements to try

```
select customerid, firstname, lastname, country
from customers
where not country = 'USA';

select sepal_length, sepal_width
from iris
where id < 50;
```

Database access using Python

- **SQLite3**
  - SQLite is a self-contained, high-reliability, embedded, full-featured, public-domain, SQL database engine.
  - SQLite is the most used database engine in the world.
  - SQLite3 is integrated in Python (from 2.5.x) by default.

https://www.sqlite.org/index.html

General Steps

```
# import sqlite3 module
import sqlite3
# create a connection object
conn = sqlite3.connect('<DBname>.<db/sqlite>')
# create a cursor object
cur = conn.cursor()
# call the cursor's execute() method to perform
SQL commands
cur.execute('<SQL statement>')
# save (commit) the changes
conn.commit()
# close the connection
conn.close()
```

Step-by-Step

Create a Table

```
cur.execute('''DROP TABLE IF EXISTS students;''')
cur.execute('''CREATE TABLE students(
              sid int PRIMARY KEY,
              name char[10] NOT NULL,
              dept char[2],
              age int DEFAULT 20); ''')
conn.commit()
```

- Triple-quotes – for string literals that span multiple lines.
- We need to commit once we are certain about the changes.

  In SQLite3, before committing, a *journal* file is maintained.

Step-by-Step

Select

```
cur.execute('''SELECT * FROM students
               WHERE age < 25 ORDER BY age
            ''')
for row in cur:
      print(row)
```

- To retrieve data after executing a SELECT statement, the cursor can be treated as an iterator.

```
cur.execute('''SELECT dept, MAX(age)
               FROM students GROUP BY dept
            ''')
for row in cur:
      print("dept. = ", row[0])
      print("MAX(age) = ", row[1], "\n")
```

Salisbury UNIVERSITY

Step-by-Step

Insert

```
cur.execute('''INSERT INTO students VALUES(?,?,?,?)''',
               (1,'smith','cs',35))
conn.commit()
```

- Use “?” as a placeholder and provide the values in a tuple

Step-by-Step

Insert

```
tmp = [(2,'martin','cs',20),
       (3,'Bern','cs',28),
       (4,'Hone','cs',23),
       (5,'Lihong','ee',18),
       (6,'John','ee',29),
       (7,'Flower','ee',25),
       (8,'martin','ee',20),
       (9,'smith','ma',21),
       (10,'wang','ma',19)]
cur.executemany('INSERT INTO students VALUES(?,?,?,?)', tmp)

conn.commit()
```

- Batch insert multiple rows, preparing the tuples of values in a list and call **`cur.executemany()`**

Step-by-Step

Delete

```
cur.execute('DELETE FROM students WHERE age > 30')
conn.commit()
```

Step-by-Step

Update

```
cur.execute('''UPDATE students
               SET age = 17
               WHERE name = 'martin' AND dept = “cs”
            ''')
conn.commit()
```

Thanks