5c SQLite with Python (Part 1)

Create Datatbase

- **SQL Command** : `CREATE DATABASE airline.db`

```
import sqlite3
db = sqlite3.connect('airline.db’)



db.close()
```

Opens SQLite database file named `airline.db` or automatically creates it if it doesn’t exist

We should always close the file by calling `close()`; however, `close()` itself doesn’t commit any of the changes!

Create Table

- SQL Command :

```
CREATE TABLE flights (
id INTEGER PRIMARY KEY AUTOINCREMENT,
origin VARCHAR(20) NOT NULL,
destination VARCHAR(20) NOT NULL,
duration INTEGER NOT NULL);
```

Create table

```
db = sqlite3.connect('a

c = db.cursor()
c.execute('''CREATE TABLE flights (\
             id INTEGER PRIMARY KEY AUTOINCREMENT,\
             origin VARCHAR(20) NOT NULL,\
             destination VARCHAR(20) NOT NULL,\
             duration INTEGER NOT NULL);''')
db.commit()
db.close()
```

This code will run into error when the table already exists!

For INSERT, UPDATE and DELETE, the changes are not actually saved until you **commit** the changes using `commit()`

Create table

```
db = sqlite3.connect('airline.db’)

try:
    c = db.cursor()
    c.execute('''CREATE TABLE flights (\
           id INTEGER PRIMARY KEY AUTOINCREMENT,\
           origin VARCHAR(20) NOT NULL,\
           destination VARCHAR(20) NOT NULL,\
           duration INTEGER NOT NULL);''')
except:
    print('Table already exist, cannot create.’)

db.commit()
db.close()
```

The use of Exception to avoid Python error message.

Insert one row of data into table

- SQL Command :

```
INSERT INTO flights (origin, destination,
duration) VALUES ('New York', 'London', 415);
```

id	origin	destination	duration
1	New York	London	415
2	Shanghai	Paris	760
3	Istanbul	Tokyo	700
4	New York	Paris	435
5	Moscow	Paris	245
6	Lima	New York	455

Insert multiple rows of data into table

- SQL Command :

```
INSERT INTO flights
     (origin, destination, duration)
VALUES
     ('Shanghai', 'Paris', 760),
     ('Istanbul', 'Tokyo', 700),
     ('New York', 'Paris', 435),
     ('Moscow', 'Paris', 245),
     ('Lima', 'New York', 455);
```

Inserting data into table

- There are 5 ways to insert data into a table using Python code:

#1 : Insert the first row of data directly

```
import sqlite3

db = sqlite3.connect('airline.db')

c = db.cursor()
c.execute('''INSERT INTO flights(origin,
               destination, duration) \
          VALUES ('New York', 'London', 415) ''')

db.commit()
db.close()
```

Important: If ANY part of these data comes from user input, then it cannot be trusted.

#2 : Insert the 2nd row of data using a tuple

```
import sqlite3

db = sqlite3.connect('airline.db')

c = db.cursor()
c.execute('''INSERT INTO flights(origin,
              destination, duration) \
         VALUES(?,?,?)''',
              ('Shanghai', 'Paris', 760))

db.commit()
db.close()
```

Important: If ANY part of these data comes from user input, then it cannot be trusted. Put any untrusted data in a tuple and use this "?" syntax to safely include them in the SQL command.

#3 : Insert the 3rd row using a dictionary

```
import sqlite3
db = sqlite3.connect('airline.db')
c = db.cursor()

c.execute('''INSERT INTO flights\
    (origin, destination, duration) \
    VALUES(:origin, :destination, :duration)''',\
    {'origin':'Istanbul','destination':'Tokyo',
     'duration':700})

db.commit()
db.close()
```

Another way: Using dictionary to insert user supplied data.

#4 : Insert multiple rows of data using a list

```
datalist = [('New York', 'Paris', 435),
            ('Moscow', 'Paris', 245),
            ('Lima', 'New York', 455)]

for data in datalist:
    c.execute('''INSERT INTO flights(origin,
               destination, duration) \
    VALUES(:origin, :destination, :duration)''',
               data)

db.commit()
db.close()
```

#5 : Insert 6 data from csv file

```
import sqlite3
import csv
db = sqlite3.connect('airline.db')
c = db.cursor()

f = open("flights.csv")
reader = csv.reader(f)
for o, dest, dur in reader:
    db.execute('''INSERT INTO flights \
         (origin, destination, duration) \
        VALUES (:origin, :destination,
                :duration)''',
         {"origin":o, "destination":dest,
               "duration":dur})

db.commit()
db.close()
```

View data in the table

- SQL Command : View all the data in the table

```
SELECT * FROM flights;
```

- SQL Command : View some of the fields in the table

```
SELECT origin, destination FROM flights;
```

List the data in the table (12 records)

```
import sqlite3
db = sqlite3.connect('airline.db')
c = db.cursor()
c.execute('''SELECT origin, destination, duration
         FROM flights''')

all_data = c.fetchall()
for data in all_data:
    print(data[0], 'to', data[1], ',', data[2],
          'minutes.')


db.commit()
db.close()
```

Ask cursor c to fetch all the the results and store them in the variable named all_data

Create table and import data from csv

Create the another table `passengers` in the `airline.db` database and import the data from the `passengers.csv` file provided.

- SQL Command :

```
CREATE TABLE passengers (\
id INTEGER PRIMARY KEY AUTOINCREMENT,\
name VARCHAR(10) NOT NULL,\
flight_id INTEGER NOT NULL \
          REFERENCES flights(id));
```

Write the Python code to list the data in the `passengers` table.