Lab 08.2 Python and SQLite3

Using databases

Overview

Using SQLite3 in python.

We do not need to install anything, SQLite is built into python

Get a connection/cursor and create a table

1. You only need to create the table once, so lets put this in a separate file called **createbooktable.py**

```
import sqlite3
con = sqlite3.connect("pfda.db") # I will call this database pdfa.db
cur = con.cursor()
#sql = "DROP TABLE IF EXISTS book"
#cur.execute(sql)

cur.execute("CREATE TABLE book(title, author, ISBN)")
# no errors? we should write some code to test this exists
# or even a print to say "done"
con.close()
```

Insert some data into it and see it is there

2. Create program called **insertbook.py**

```
import sqlite3
con = sqlite3.connect("pfda.db")
cur = con.cursor()

# check there is nothing in the database
result = cur.execute("select * from book")
print (result.fetchone())

# insert a book
sql = """
    INSERT INTO book VALUES
        ('Harry Pothead', 'Just Kidding Really', "112344"),
        ('Harry Potter does something profound', 'JK Rowling', "123444")
"""
# DANGER DANGER this can lead to SQL injection
cur.execute(sql)
# something is missing here.... what do you think it is?


result = cur.execute("select * from book")
print (result.fetchone())
con.close()
```

3. run it and you get.

```
None

('Harry Pothead', 'Just Kidding Really', '112344')
```

That looks good, firstly there is nothing in the database and then there is something. Happy days.

4. Run it again….. did you get the same?

```
None

('Harry Pothead', 'Just Kidding Really', '112344')
```

WAIT!!!!!! WHAT??? I thought that databases were supposed to be persistent, why was there nothing in the database when I ran it again… that does not seem useful

Well, we were missing something…. Namely the

```
con.commit()
```

```
import sqlite3
con = sqlite3.connect("pfda.db")
cur = con.cursor()

# check there is nothing in the database
result = cur.execute("select * from book")
print (result.fetchone())

# insert a book
sql = """
    INSERT INTO book VALUES
        ('Harry Pothead', 'Just Kidding Really', "112344"),
        ('Harry Potter does something profound', 'JK Rowling', "123444")
"""
# DANGER DANGER this can lead to SQL injection
cur.execute(sql)
con.commit() # remember to commit your updates


result = cur.execute("select * from book")
print (result.fetchone())
con.close()
```

output the entire table

5. use fetchall to get the entire table, in a file called **getallbooks.py**

```
import sqlite3
con = sqlite3.connect("pfda.db")
cur = con.cursor()


for row in cur.execute("select * from book"):
    print (f"row{row}")
```

SQL Injection

If you are taking any input from a user (directly or indirectly) never put it into an SQL String.

Actually, I would say never put a ANY variable into a SQL string

6. Write a safe program that reads in book details and puts them into the database. (it can output the entire table to check it worked) called **enterbook.py**

```
import sqlite3
con = sqlite3.connect("pfda.db")
cur = con.cursor()

book = {}
book['title'] = input("please enter book title:")
book['author'] = input("please enter book author:")
book['ISBN'] = input("please enter book ISBN:")
#print (book)

data = [book] # should be [], though the docs sometimes have ()
#data = [{"title":"aa","author":"ass","ISBN":"ddd"}]
sql = "insert into book values (:title, :author, :ISBN)"
cur.executemany(sql, data)
con.commit()

for row in cur.execute("select * from book"):
    print (f"row{row}")
```