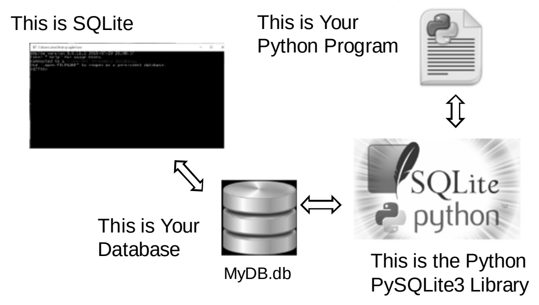
Python and SQLite3

Dr. John Artz

Overview

- Database Concepts
- Inserting, Deleting and Updating Data
- Select Statements in PySQL
- Project Tip
- Ideas in Progress

Python and SQLite Configuration


This is SQLite
This is Your Python Program
This is Your Database
MyDB.db
SQLite python
This is the Python PySQLite3 Library

Database Concepts

- Database Connection – You May Have Multiple Databases, So You Have to Let SQLite3 Know Which One You Want to Use Via a Database Connection
- Cursor – A Database Cursor is an Object That Allows You To Execute an SQL Statement or Traverse A Set of Rows Returned By a Query

Database Connection

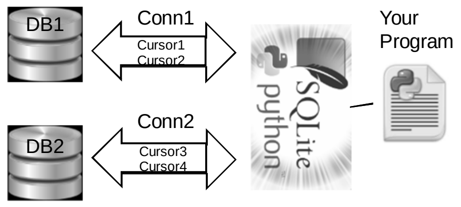
```
import sqlite3  #Import PySQLite3 Library
conn = sqlite3.connect("MyDB.db") #Create Connection
print "conn is of type", type(conn)
```

This Creates a Connection to a Database File

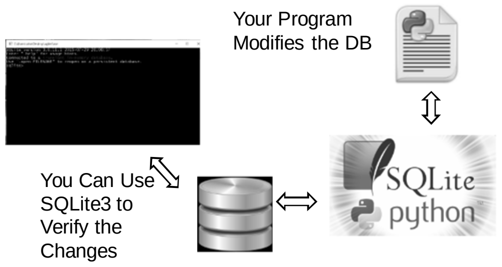
Creating a Cursor

- cursor = conn.cursor()
- A cursor is Created By the Connection Object to Allow Commands to Be Executed
- A Program May Have Multiple Cursors Connected to the Database Through the Connection

Many Connections, Many Cursors



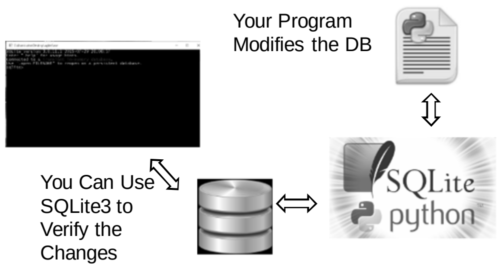
Using SQLite to Verify


Your Program Modifies the DB
You Can Use SQLite3 to Verify the Changes
SQLite
python

Using a Cursor to Create a Table

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
cur = conn.cursor()
cur.execute("Drop Table if Exists Days") ←Why This
cur.execute("create table Days (DoY int not null,DoW
text,Holiday int,Weather text);")
print "Table Created"
```

Verify With .tables


Your Program Modifies the DB
You Can Use SQLite3 to Verify the Changes

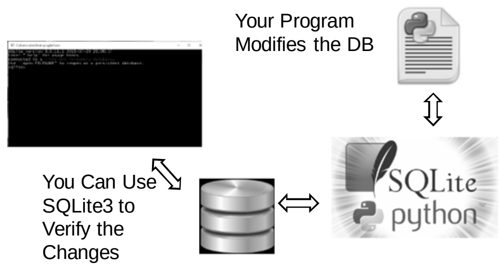
Using a Cursor to Drop a Table

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
cur = conn.cursor()
cur.execute("Drop Table if Exists Days")
print "Table Dropped"
```

Inserting Data

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
print type(conn)
cur = conn.cursor()
cur.execute("insert into Days (DoY, DoW, Holiday,
Weather) values (1, 'Thurs', 1, 'Snow')")
```

Verify With Select - What Went Wrong?



Inserting Data With Commit

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
print type(conn)
cur = conn.cursor()
cur.execute("insert into Days (DoY, DoW, Holiday,
Weather) values (1, 'Thurs', 1, 'Snow')")
conn.commit()
```

Inserting Data With Rollback

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
print type(conn)
cur = conn.cursor()
cur.execute("insert into Days (DoY, DoW, Holiday,
Weather) values (1, 'Thurs', 1, 'Snow')")
conn.rollback()
```

A Brief Digression – Transaction Processing

- Delete Contents of Days Table

Begin Transaction

 insert into Days (DoY, DoW, Holiday, Weather) values (1, 'Thurs', 1, 'Snow');

   insert into Days (DoY, DoW, Holiday, Weather) values (2, 'Fri', 0, 'Snow');

   insert into Days (DoY, DoW, Holiday, Weather) values (3, 'Sat', 0, 'Clear');

Show Query Rollback, Commit

Getting Fancy with Primary Key

- import sqlite3
- conn = sqlite3.connect("MyDB.db")
- cur = conn.cursor()
- cur.execute("Drop Table if Exists Days")
- cur.execute("create table Days (DoY int Primary Key not null,DoW text,Holiday int,Weather text);")
- print "Table Created"

Inserting Data Twice

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
print type(conn)
cur = conn.cursor()
cur.execute("insert into Days (DoY, DoW, Holiday, Weather) values (1, 'Thurs', 1, 'Snow')")
cur.execute("insert into Days (DoY, DoW, Holiday, Weather) values (1, 'Thurs', 1, 'Snow')")
conn.commit()
```

Ignoring Duplicates

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
print type(conn)
cur = conn.cursor()
cur.execute("insert or ignore into Days (DoY, DoW, Holiday, Weather)
values (1, 'Thurs', 1, 'Snow')")
cur.execute("insert or ignore into Days (DoY, DoW, Holiday, Weather)
values (1, 'Thurs', 1, 'Snow')")
conn.commit()
```

Inserting Multiple Rows

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
cur = conn.cursor()
cur.execute("drop table if exists Days")
cur.execute("create table Days (DoY int not null,DoW text,Holiday int,Weather text);")
cur.execute("insert into Days (DoY, DoW, Holiday, Weather) values (1, 'Thurs', 1, 'Snow')")
cur.execute("insert into Days (DoY, DoW, Holiday, Weather) values (2, 'Fri', 0, 'Snow')")
cur.execute("insert into Days (DoY, DoW, Holiday, Weather) values (3, 'Sat', 0, 'Clear')")
conn.commit()
print "Rows Inserted and  Committed"
conn.close()
```

Triple Quoted Strings in Python

```
print """we have single quotes '
A new line
and double quotes " in this string and it doesn't
matter"""
```

- Triple Quoted Strings Allow Us to Print Out Strings Ignoring Internal Quotes

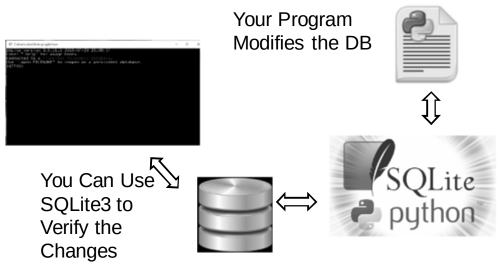
Inserting with Script

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
cur = conn.cursor()
cur.executescript("""
   drop table if exists Days;
   create table Days (DoY int not null,DoW text,Holiday int,Weather text);
   insert into Days (DoY , DoW, Holiday, Weather) values (1, 'Thurs', 1, 'Snow');
   insert into Days (DoY , DoW, Holiday, Weather) values (2, 'Fri', 0, 'Snow');
   insert into Days (DoY , DoW, Holiday, Weather) values (3, 'Sat', 0, 'Clear');
   """)
conn.commit()
print "Rows Inserted and  Committed"
conn.close()
```

Inserting Data With Wildcards

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
days = ( (1, "Thurs", 1, "Snow"), (2, "Fri", 0, "Snow"), (3, "Sat", 0, "Clear"))
cur = conn.cursor()
cur.execute("drop table if exists Days")
cur.execute("create table Days (DoY int not null,DoW text,Holiday
int,Weather text);")
cur.executemany("INSERT INTO Days VALUES(?, ?, ?,?)", days)
conn.commit()
print "Rows Inserted and  Committed"
conn.close()
```

Verify Inserts With Select


Your Program Modifies the DB
You Can Use SQLite3 to Verify the Changes
SQLite
python

Using Files

- We Can Read Insert Statements from Files
- We Can Read Data From Files and Put the Values into Insert Statements
- We Can Read Data From Files, Create a List of Lists and Use executemany
- Or, We Can Parse a File of Data Which We Will Get To Later

Update

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
cur = conn.cursor()
cur.execute("Update Days Set Weather = 'Rain'
Where DoY = '2' ")
conn.commit()
print cur.rowcount, "Rows Updated"
```

Delete

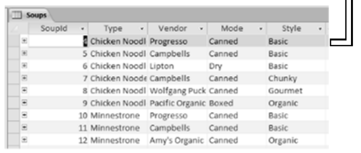
```
import sqlite3
conn = sqlite3.connect("MyDB.db")
cur = conn.cursor()
cur.execute("Delete from Days Where DoY = '1'")
conn.commit()
print cur.rowcount, "Rows Deleted"
```

Select

- Selects Work Similar to Insert, Update and Delete Except That the Select Returns One or More Rows That Need to Be Dealt With Somehow
- We Will Learn More About the Cursor Objects to Deal With Selects

List of Returned Row Values

[4, “Chicken Noodle”, “Progresso”. “Canned”, “Basic”]



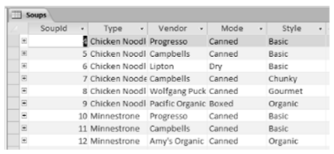
Soups

SoupId	Type	Vendor	Mode	Style
4	Chicken Noodl	Progresso	Canned	Basic
5	Chicken Noodl	Campbells	Canned	Basic
6	Chicken Noodl	Lipton	Dry	Basic
7	Chicken Noode	Campbells	Canned	Chunky
8	Chicken Noodl	Wolfgang Puck	Canned	Gourmet
9	Chicken Noodl	Pacific Organic	Boxed	Organic
10	Minnestrone	Progresso	Canned	Basic
11	Minnestrone	Campbells	Canned	Basic
12	Minnestrone	Amy's Organic	Canned	Organic

Each Returned Row is a List of Values

Rowset as a List of Lists

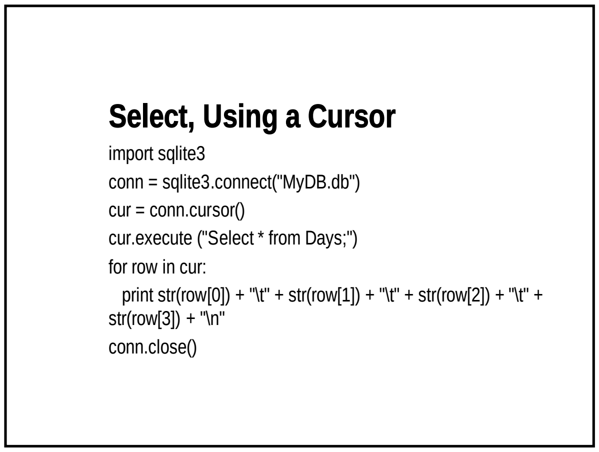
Each Returned Rowset is a List of Lists



Soups

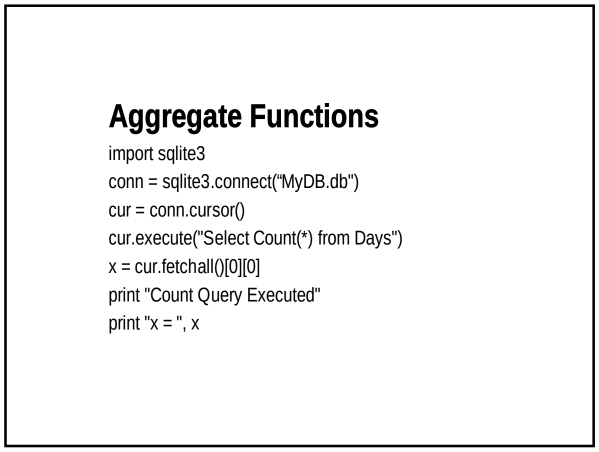
SoupId	Type	Vendor	Mode	Style
4	Chicken Noodl	Progresso	Canned	Basic
5	Chicken Noodl	Campbells	Canned	Basic
6	Chicken Noodl	Lipton	Dry	Basic
7	Chicken Noode	Campbells	Canned	Chunky
8	Chicken Noodl	Wolfgang Puck	Canned	Gourmet
9	Chicken Noodl	Pacific Organic	Boxed	Organic
10	Minnestrone	Progresso	Canned	Basic
11	Minnestrone	Campbells	Canned	Basic
12	Minnestrone	Amy's Organic	Canned	Organic

[[4, “Chicken Noodle”, “Progresso”. “Canned”, “Basic”]
[5, “Chicken Noodle”, “Campbell's”, “Canned”, “Basic”]]



Select, Using a Cursor

```
import sqlite3
conn = sqlite3.connect("MyDB.db")
cur = conn.cursor()
cur.execute ("Select * from Days;")
for row in cur:
   print str(row[0]) + "\t" + str(row[1]) + "\t" + str(row[2]) + "\t" +
str(row[3]) + "\n"
conn.close()
```



Aggregate Functions

```
import sqlite3
conn = sqlite3.connect(“MyDB.db")
cur = conn.cursor()
cur.execute("Select Count(*) from Days")
x = cur.fetchall()[0][0]
print "Count Query Executed"
print "x = ", x
```

PROJECT 2 TIPS

Creating Tables

```
def CreateTables():
    cur.execute("Drop Table if Exists Sales")
    cur.execute("create table sales
(TrxId int not null,DoY int,StoreID int,SoupId int,PromoId int,Sales number);")
    print "Table Created"
```

We Need One of These for Each Table in the DB

Inserting Data

```
def InsertSales(record):#We Need One of These Too
    TrxId = int(record[0])
    DoY = int(record[1])
    StoreId = int(record[2])                The Order May Change,
    SoupId = int(record[3])                 So, Consult the File Definition
    PromoId = int(record[4])
    Sales = float(record[5])
    row = [TrxId,DoY,StoreId,SoupId, PromoId, Sales]
    cur.execute("INSERT or ignore INTO Sales VALUES(?, ?, ?,?,?,?)", row)
    print "Row Inserted: ", row
```

Q You Will Need One of These for Each Table

Parsing the File

```
conn = sqlite3.connect("MyDB.db")
cur = conn.cursor()
CreateTables()
f = open("SalesParseData.txt", "r")
linecount = 0
line = f.readline()
while line != "" and linecount < 5:
    linecount = linecount + 1
    line = line.replace("\n","")
    linelist = line.split("\t")
#    InsertDays(linelist), etc
    InsertSales(linelist)
    line = f.readline()
conn.commit()
f.close()
```

I'm Working on the Following Ideas

FetchOne, FetchMany & FetchAll

- Allows You to Retrieve Portions of a Returned Rowset

Offset and Limit

- Use Offset and Limit to Step Through Large Dataset

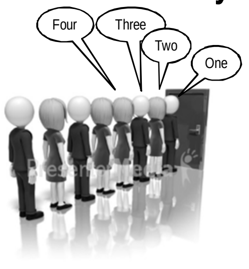
Graph Sales Using Matplotlib

- Maybe in Last Week
- See Sqlite3 tutorial 4 on Youtube by sentdex
- [print row for row in c.fetchall()] creates list
- Watch Beautiful Soup Tutorials on YouTube

Looking Ahead to Database Design

- The Next Two Weeks Will Focus on Database Design
- To Illustrate the Problem of Database Design I Will Provide Two Examples
- You Know What a Faculty Member is
- And You Know What a Course Is
- Or Do You?

What Do You Mean By Faculty?



What Do You Mean By Course?

Course	Section	Description	Day	Time
ISTM6202	10	Database	F	4-6
ISTM6202	11	Database	M	6-8
ISTM6202	12	Database	R	6-8
ISTM6203	10	Telecom	W	6-8
ISTM6203	11	Telecom	M	8-10
ISTM6204	10	Proj. Mgmt	R	8-10
ISTM6207	10	IRM	T	6-8

How Many Courses Are Offered?

The Answer Could Be 4 or 7 Depending On What You Mean By "Course"

Recap

- Database Concepts
- Inserting, Deleting and Updating Data
- Select Statements in PySQL
- Project Tip
- Ideas in Progress