CMPT 321
FALL 2017

Embedding SQL into Python applications

Lecture 03.07

By Marina Barsky

- SQL systems are useful for efficient interaction with on-disk systems
- There are many important things which cannot be achieved with pure SQL:
  - Try to write SQL query to compute factorial n!
  - Format output of a query into a graphics
  - Find all relatives of Fred
- Thus, real database programming requires both SQL and a host language.

Impedance mismatch

- The basic problem of connecting SQL statements with conventional programming language is ***impedance mismatch***:
  - SQL – relational model – tables
  - Programming languages: integers, strings, arrays, pointers

Establishing connection between host language and SQL

- Database *driver*: software distributed by a DBMS vendor which exposes operations on database to a particular language
- Shared variables to transfer data between two types of data structures: *cursors*
  - A cursor can be viewed as a pointer to one row in a set of rows
  - The cursor can only reference one row at a time, but can move to other rows of the result set as needed

Host language: Python
DBMS: SQLite (SQLite drive built-in)

- Establishing connection

```
import sqlite3 as lite

SQLITE_DB = 'pizza.db'

try:
    con = lite.connect(SQLITE_DB)

except lite.Error as e:
    if con:
        con.rollback()
```

Reading table into a cursor

```
con.row_factory = lite.Row

cur = con.cursor()
if params:
    cur.execute(sql, params)
else:
    cur.execute(sql)

rows = cur.fetchall()
```

Example of transfer from cursor into a Python list

```
sql = '''SELECT distinct (pizzeria)
      FROM SERVES
      ORDER BY pizzeria’’’

con.row_factory = lite.Row

cur = con.cursor()
cur.execute(sql)
rows = cur.fetchall()

pizzerias = []
for row in rows:
    pizzerias.append(row["pizzeria"])
```

Example with params

```
sql = '''SELECT *
         FROM Serves
         WHERE pizzeria =:name
         ORDER BY pizza'''
params = {"name": val}


cur = con.cursor()
cur.execute(sql, params)
```

Executing update

```
try:
    cur = con.cursor()
    result = cur.execute(sql, params)
    con.commit()

except lite.Error as e:
    if con:
        con.rollback()

    print("Transaction failed: {0}".format(e))
```

Example of deletion

```
sql = '''DELETE FROM Serves
    WHERE pizzeria = ?
    AND pizza=?'''
params = (pizzeria, pizza)

cur = con.cursor()
result = cur.execute(sql, params)
con.commit()
```

Example of update

```
sql = '''UPDATE Serves
    SET price =?
    WHERE pizzeria = ?
    AND pizza = ?'''
params = (price, pizzeria, pizza)

cur = con.cursor()
result = cur.execute(sql, params)
con.commit()
```