CSE 503

Introduction to Computer Science for Non-Majors

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall

Day 28
Databases (part 2)

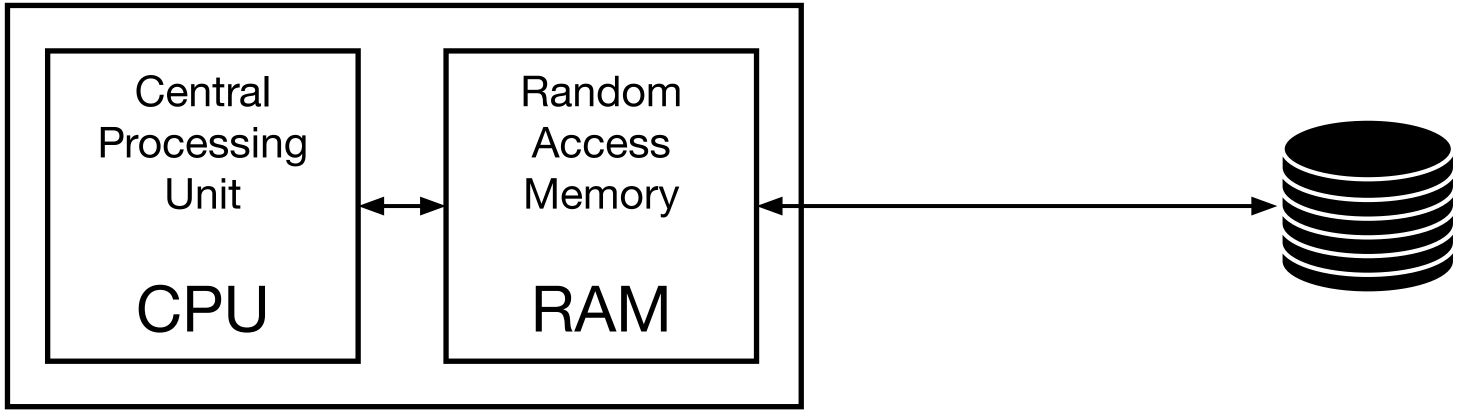
Storing Data

In Memory/CPU

- Transient (exists while program is running)
- Limited size

On Disk

- Persistent
- Larger capacity
- Text files, csv files, databases, etc



Storing Data

Text Files: Streams of characters

CSV Files: Comma separated values

Databases: Tables of data supporting highly efficient operations

(CSE 560 Data Models and Query Languages; CSE 562 Database Systems)

SQLite - Creating and Storing a DB

```python
import sqlite3

conn = sqlite3.connect('test.db')
cur = conn.cursor()

# do things to database

conn.commit()
conn.close()
```

1. Import the SQLite library
2. Open a connection to a DB
   (creates the DB if necessary)
3. Create a cursor object
   (this is how we interact with the DB)
4. Do stuff…
5. Commit our changes and close the DB
   (without commit, changes are lost)

Example Commands

```
cur.execute('CREATE TABLE IF NOT EXISTS movies (title, director, year)')

cur.execute('INSERT INTO movies VALUES ("Jaws", "Spielberg", 1975)')

results = cur.execute('SELECT * FROM movies')

results = cur.execute('SELECT * FROM movies WHERE year = 1975')
```

Parameterized Commands

How can we create commands from variables?

Parameterized Commands

How can we create commands from variables?

We could build up a string using multiple concatenations…but that would be tedious (and cause other issues we will see in the future)

Parameterized Commands

How can we create commands from variables?

We could build up a string using multiple concatenations…but that would be tedious (and cause other issues we will see in the future)

SQLite let us parameterize our commands!

Parameterized Commands

```
def insert(title, director, year):
  cur.execute('INSERT INTO movies VALUES (?,?,?)', (title,director,year))

def get_all_by_year(year):
    return cur.execute('SELECT * FROM movies WHERE year=?',(year,))
```

Parameterized Commands

```
def insert(title, director, year):
  cur.execute('INSERT INTO movies VALUES (?,?,?)', (title,director,year))

def get_all_by_year(year):
    return cur.execute('SELECT * FROM movies WHERE year=?',(year,))
```

? indicates values that will be filled in

Parameterized Commands

```
def insert(title, director, year):
  cur.execute('INSERT INTO movies VALUES (?,?,?)', (title,director,year))

def get_all_by_year(year):
    return cur.execute('SELECT * FROM movies WHERE year=?',(year,))
```

We pass a tuple with the specific values

Parameterized Commands

```
def insert(title, director, year):
  cur.execute('INSERT INTO movies VALUES (?,?,?)', (title,director,year))

def get_all_by_year(year):
    return cur.execute('SELECT * FROM movies WHERE year=?',(year,))
```

Tuples in Python:

```
(x,)    ← Tuples of size one (notice the comma)
(x,y)   ← Tuples of size two
(x,y,z) ← Tuples of size three (or more…as many values as we want)
```

Big Example - Music Rating App

MusicRater1.0

- Python server and JS/HTML client for rating songs
- Songs and ratings stored in CSV files

MusicRater2.0

- Python server and JS/HTML client for rating songs
- Songs and ratings stored in SQLite Database