Lecture 11:
Finish Web Technologies &
Begin SQL Databases

Slides by:
Joseph E. Gonzalez
jegonzal@berkeley.edu

Last Two Lectures

- Last Thursday: String manipulation & Regular Expressions
  - guest lecture from the amazing Sam Lau
  - reviewed in section and in future labs & HWs
- Last Tuesday: HTTP, XML, and JSON
  - Pandas web tables support
  - Using the browser developer mode
  - JSON and basics of XML
  - Started HTTP request/response protocol and GET vs POST
  - Didn’t finish REST and web-services …

REST APIs

Example:

GET /website/images
Get all images

POST /website/images
Add an image

GET /website/images/{id}
Get a an image

PUT /website/images/{id}
Update an image

DELETE /website/images/{id}
Delete an image

Client

Server

REST – Representational State Transfer

- A way of architecting widely accessible, efficient, and extensible
  web services (typically using HTTP)
- Client-Server: client and server are able to evolve independently
- Stateless: the server does not store any of the clients session state
- Cacheable: system should clearly define what functionality can be
  cached (e.g., GET vs POST requests)
- Uniform Interface: provide a consistent interface for getting and
  updating data in a system

Scraping Ethics

- Don’t violate terms of use for the service or data
- Scraping can cause result in degraded services for others
- Many services are optimized for human user access patterns
- Requests can be parallelized/distributed to saturate server
- Each query may result in many database requests
- How to scrape ethically:
  - Used documented REST APIs – read terms of service
  - Examine at robots.txt (e.g., https://en.wikipedia.org/robots.txt)
  - Throttle request rates (sleep)
- Avoid getting Berkeley (or your organization) blocked
  from websites & services

Demo

TwitterAPI_REST_Example.ipynb
Databases and SQL

Part 1

Slides by:
Joseph E. Gonzalez & Joseph Hellerstein,
jegonzal@berkeley.edu
jhellerstein@berkeley.edu

What is a database?

Defining Databases

A database is an organized collection of data.

A database management systems (DBMS) is a software system that stores, manages, and facilitates access to one or more databases.

Database Management Systems

Data storage
- Provide reliable storage to survive system crashes and disk failures
- Special data-structures to improve performance

Data management
- Configure how data is logically organized and who has access
- Ensure data consistency properties (e.g., positive bank account values)

Facilitate access
- Enable efficient access to the data
- Supports user defined computation (queries) over data

Is Pandas a Database Management System?

Data Storage?
- Pandas doesn’t store data, this is managed by the filesystem

Data Management?
- Pandas does support changing the organization of data but doesn’t manage who can access the data

Facilitate Access?
- Pandas does support rich tools for computation over data

Pandas is not generally considered a database management system but it often interacts with DBMSs

Why should I use a DBMS?

Why can’t I just have my CSV files?

- DBMSs organize many related sources of information
- DBMSs enforce guarantees on the data
- Can be used to prevent data anomalies
- Ensure safe concurrent operations on data

- DBMSs can be scalable
- Optimized to compute on data that does not fit in memory
- Parallel computation and optimized data structures
- DBMSs prevent data loss from software/hardware failures
Widely Used DBMS Technologies

Relational Database Management Systems
- Relational databases are the traditional DBMS technology
- Logically organize data in relations (tables)

Sales relation:
<table>
<thead>
<tr>
<th>Name</th>
<th>Prod</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sue</td>
<td>iPod</td>
<td>$200.00</td>
</tr>
<tr>
<td>Joey</td>
<td>Bike</td>
<td>$333.99</td>
</tr>
</tbody>
</table>

Tuple (row) Alice Car $999.00 Attribute (column)

Describes relationship: Name purchased Prod at Price.

How is data physically stored?

Physical Data Independence:
Database management systems hide how data is stored from end user applications
- System can optimize storage and computation without changing applications.

Big Idea in Data Structures
Data Systems & Computer Science

It wasn’t always like this ...
Before 1970’s databases were not routinely organized as tables. Instead they exposed specialized data structures designed for specific applications.

Ted Codd and the Relational Model

- [1969] Relational model: a mathematical abstraction of a database as sets
  - Independence of data from the physical properties of stage storage and representation
- [1972] Relational Algebra & Calculus: a collection of operations and a way defining logical outcomes for data transformations
  - Algebra: beginning of technologies like Pandas
  - Calculus: the foundation of modern SQL

Relational Database Management Systems

- Traditionally DBMS referred to relational databases
- Logically organize data in relations (tables)
- Structured Query Language (SQL) to define, manipulate and compute on data.
  - A common language spoken by many data systems
  - Some variations and deviations from the standard...
  - Describes logical organization of data as well as computation on data.

SQL is a Declarative Language

- Declarative: “Say what you want, not how to get it.”
  - Declarative Example: I want a table with columns “x” and “y” constructed from tables “A” and “B” where the values in “y” are greater than 100.00.
  - Imperative Example: For each record in table “A” find the corresponding record in table “B” then drop the records where “y” is less than or equal to 100 then return the “x” and “y” values.
- Advantages of declarative programming
  - Enable the system to find the best way to achieve the result.
  - Often more compact and easier to learn for non-programmers
- Challenges of declarative programming
  - System performance depends heavily on automatic optimization
  - Limited language (not Turing complete)

Review of Relational Terminology

- Database: Set of Relations (i.e., one or more tables)
- Relation (Table):
  - Schema: description of columns, their types, and constraints
  - Instance: data satisfying the schema
- Attribute (Column)
- Tuple (Record, Row)
- Schema of database: is set of schemas of its relations
Two sublanguages of SQL

- DDL – Data Definition Language
  - Define and modify schema
- DML – Data Manipulation Language
  - Queries can be written intuitively.

CAPITALIZATION IS optional but... DATABASE PEOPLE PREFER TO YELL

**Common SQL Types (there are others...)**

- **CHAR(size):** Fixed number of characters
- **TEXT:** Arbitrary number of character strings
- **INTEGER & BIGINT:** Integers of various sizes
- **REAL & DOUBLE PRECISION:** Floating point numbers
- **DATE & DATETIME:** Date and Date+Time formats

See documentation for database system (e.g., Postgres)

**Inserting Records into a Table**

```sql
INSERT INTO students (name, gpa, age, dept, gender) VALUES ('Sergey Brin', 2.8, 40, 'CS', 'M'), ('Danah Boyd', 3.9, 35, 'CS', 'F'), ('Bill Gates', 1.0, 60, 'CS', 'M'), ('Hillary Mason', 4.0, 90, 'DATASCI', 'F'), ('Mark Zuckerberg', 4.0, 30, 'CS', 'M'), ('Sheryl Sandberg', 4.0, 47, 'BUSINESS', 'F'), ('Susan Wojcicki', 4.0, 46, 'BUSINESS', 'F'), ('Marissa Mayer', 4.0, 45, 'BUSINESS', 'F');
```

-- This is a comment.
-- Does the order matter? No
Deleting and Modifying Records

- Records are deleted by specifying a condition:
  
  ```
  DELETE FROM students
  WHERE LOWER(name) = 'sergey brin'
  ```

- Modifying records
  
  ```
  UPDATE students
  SET gpa = 1.0 + gpa
  WHERE dept = 'CS';
  ```

- Notice that there is no way to modify records by location

Querying Tables

SQL DML: Basic Single-Table Queries

```sql
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>]
[HAVING <predicate>]
[ORDER BY <column list>];
```

- Simplest version is straightforward
- Produce all tuples in the table that satisfy the predicate
- Output the expressions in the SELECT list
- Expression can be a column reference, or an arithmetic expression over column refs

Find the name and GPA for all CS Students

```sql
SELECT name, gpa
FROM students
WHERE dept = 'CS';
```
**SELECT DISTINCT**

```sql
SELECT DISTINCT dept
FROM students
WHERE dept = 'CS'
GROUP BY dept
HAVING AVG(gpa) > 3.0
ORDER BY AVG(gpa) DESC;
```

- DISTINCT flag specifies removal of duplicates before output

**ORDER BY**

```sql
SELECT name, gpa, age
FROM students
WHERE dept = 'CS'
GROUP BY name
HAVING AVG(gpa) > 3.0
ORDER BY gpa, name;
```

- ORDER BY clause specifies output to be sorted
  - Lexicographic ordering

**Aggregates**

```sql
SELECT AVG(gpa)
FROM students
WHERE dept = 'CS'
GROUP BY dept
HAVING AVG(gpa) > 3.0
ORDER BY AVG(gpa) DESC;
```

- Before producing output, compute a summary statistic
  - Aggregates include: SUM, COUNT, MAX, MIN, ...
  - Produces 1 row of output
  - Still a table
  - Note: can use DISTINCT inside the agg function
  - SELECT COUNT(DISTINCT name) ...

**GROUP BY**

```sql
SELECT dept, AVG(gpa)
FROM students
WHERE dept = 'CS'
GROUP BY dept
HAVING AVG(gpa) > 3.0
ORDER BY AVG(gpa) DESC;
```

- Partition table into groups with same GROUP BY column values
  - Group By takes a list of columns
  - Produce an aggregate result per group

**What does the following Produce?**

```sql
SELECT name, AVG(gpa)
FROM students
WHERE dept = 'CS'
GROUP BY dept
HAVING AVG(gpa) > 3.0
ORDER BY AVG(gpa) DESC;
```

- An error! (why?)
  - What name should be used for each group?
What if we wanted to only consider departments that have greater than two students?

```
SELECT dept, AVG(gpa)
FROM students
GROUP BY dept
HAVING COUNT(*) > 2
ORDER BY avg_gpa DESC;
```

Doesn’t work ...

- WHERE clause is applied before GROUP BY
- You cannot have aggregation functions in the where clause

**Conceptual SQL Evaluation**

Try Queries Here: http://sqlfiddle.com/#!17/67109/12

### Putting it all together

```
SELECT dept, AVG(gpa) AS avg_gpa, COUNT(*) AS size
FROM students
WHERE gender = 'F'
GROUP BY dept
HAVING COUNT(*) > 2
ORDER BY avg_gpa DESC;
```

What does this compute?

- The average GPA of female students and number of female students in each department where there are at least 3 female students in that department. The results are ordered by the average GPA.
How do you interact with a database?

What is the DBMS?
- Server
- Software
- A library

Answer: It can be all of these.

Interacting with a DBMS

Query

```
SELECT * FROM sales
WHERE price > 100.0
```

Response

```
+-------------------+--------+--------+-------+
| Date              | ID     | Name   | Price |
|-------------------+--------+--------+-------|
| 9/20/2012         | 1234   | Sue    | 200.00|
| 8/21/2012         | 3453   | Joe    | 333.99|
+-------------------+--------+--------+-------+
```

Why are databases drawn as “cans”?

Break

Platters on a Disk Drive
Platters on a Disk Drive

1956: IBM MODEL 350 RAMAC
First Commercial Disk Drive
5MB @ 1 ton

Simple Single Table Query Demo