Objectives

- PostgreSQL DB
- Elasticsearch

Review

- Databases
  - What language do we use to query databases?

Database Overview

- Store data in such a way to allow *efficient* storage, search, and update
- **Relational Data Model** - currently most popular type of database
  - Different vendors: PostgreSQL, Oracle, MySQL, DB2, MSSQL
  - Data is stored in **tables**
  - ***Attributes***: column names (one word)
  - ***Entities***: table
  - Often contain ***primary key***:
    a set of columns that uniquely identify a row

Example Students Table

- id is the primary key
- What are the attributes?

Attributes

id	lastName	firstName	gradYear	major
10011	Aaronson	Aaron	2018	CSCI
43123	Brown	Allison	2017	ENGL

Review: SELECT Command

- Queries the database
- Returns a result—a ***virtual table***
- Syntax:

Optional

```
SELECT column_names
FROM table_names [WHERE condition];
```

  - Columns, tables separated by commas
  - Can select all columns with *
  - Where clause specifies constraints on what to select from the table

ORDER BY

- Last operation performed, last in query
- Orders:
  - **ASC** = ascending
  - **DESC** = descending
- Example

```
SELECT firstName, lastName
FROM Students WHERE gradYear=2017
ORDER BY GPA DESC;
```


Aggregates

- Standard SQL aggregate functions: **COUNT, SUM, AVG, MIN, MAX**
- Can only used in the **SELECT** part of query
- Example

```
SELECT COUNT(*), AVG(GPA)
FROM students WHERE gradYear=2017;
```


JOIN Queries

- Join two tables on an attribute

Majors:

id	name	department
1	ART-BA	ART
2	ARTH-BA	ART

Students:

id	lastName	firstName	gradYear	majorID
10011	Aaronson	Aaron	2018	123
43123	Brown	Allison	2017	157

```
SELECT lastName, name
FROM Students, Majors
WHERE Students.majorID=Majors.id;
```

What if Students Have Multiple Majors?

- We don't necessarily want to add another column to Students table
  - What if student has 2 majors?
- Example of Many to Many Relationship
- Solution: Create **StudentsToMajors** table:

studentID	majorID
435	243
435	232

Primary Key: (StudentID, MajorID)
Foreign Keys from Students, Majors Tables

What if Students Have Multiple Majors?

- We don't necessarily want to add another column to Students table
  - What if student has 2 majors?
- Example of Many to Many Relationship
- Solution: Create **StudentsToMajors** table:

studentID	majorID
435	243
435	232

Primary Key: (StudentID, MajorID)
Foreign Keys from Students, Majors Tables

How will we find out a students name and their major?

INSERT Statements

- You can add rows to a table

```
INSERT INTO Majors VALUES
( 354, 'CSCI', 'Computer Science');
```

Assumes filling in all values, in column order

- Preferred Method: include column names
  - Don't depend on order

```
INSERT INTO Majors (id, name, department)
VALUES ( 354, 'CSCI', 'Computer Science');
```


INSERT Statements

- Automatically create ids

```
INSERT INTO Majors (id, name, department)
VALUES ( nextval('majors_sequence'),
'CSCI', 'Computer Science' );
```

- If table is set up appropriately, let the DB handle creating unique ids:

```
INSERT INTO Majors (name, department)
VALUES ('CSCI', 'Computer Science' );
```

UPDATE Statement

- You can modify rows of a table
- Use **WHERE** condition to specify which rows to update
- Example: Update a student's married name

```
UPDATE Students SET
LastName='Smith-Jones' WHERE id=12;
```

- Example: Update all first years to undeclared

```
UPDATE Students SET majorID=345
WHERE gradYear=2020;
```


DELETE Statement

- You can delete rows from a table

```
DELETE FROM table [ WHERE condition ];
```

- Example

```
DELETE FROM EnrolledStudents WHERE
hasPrerequisites=False AND course_id=456;
```


Using a Database

- DBMS: Database management system
- Using PostgreSQL in this class
  - Free, open source
- Slight differences in syntax between DBMSs
- DBMS can contain multiple databases
  - Need to say which DB you want to use

Designing a DB

- Design tables to hold your data
  - Data's name and types
- Similar to OO design
  - No duplication of data
  - Have pointers to info in other tables
- Main difference: no lists
  - If you think "list", think of a OneToMany or a ManyToMany table that contains the relationships between the data

Standard Data Types

- Standard to SQL
  - CHAR - fixed-length character
  - VARCHAR - variable-length character
    - Requires more processing than CHAR
  - INTEGER - whole numbers
  - NUMERIC
  - Names for types in specific DB may vary
- More data types available in each DB

PostgreSQL Data Types

- Names for standard data types
  - Numeric: `int, smallint, real, double precision`
  - Strings
    - `char(N)` - fixed length (padded)
    - `varchar(N)` - variable length, with a max
    - `text` - variable unlimited length
- Additional useful data types
  - `date, time, timestamp,` and `interval`
  - `Timestamp` includes both date and time

Constraints

- **PRIMARY KEY** may not have null values
- **UNIQUE** may have null values
  - Example: username when have a separate id
- **FOREIGN KEY**
  - Use key from another ("foreign") table
  - Example: shopping cart has its own id; references the user's id as owner
- **CHECK**
  - value in a certain column must satisfy a Boolean (truth-value) expression
  - Example: GPA >= 0

Creating a Table

- Example:

```
CREATE TABLE weather (
    city            varchar(80),
    temp_lo         int,        -- low temperature
    temp_hi         int,        -- high temperature
    prcp            real,       -- precipitation
    date            date
);
```

Databases Course Overview

- Planned CSCI317 in Winter 2018
- How do you solve more complex problems/write more complicated queries?
- How do you organize data into relational databases?
  - Design data
- How do you store data so that you can access it and manipulate it efficiently?
  - Underlying data structures
- How do you handle concurrent transactions correctly and efficiently?

Practicing SQL with PostgreSQL

- `ssh to carl`
- `psql cs397`
- View the tables and sequences
  - `\d`
- View just the tables
  - `\dt`
- View the info for one table
  - `\d students`

Practicing SQL with PostgreSQL

- View the contents of a table
  - `SELECT * FROM students;`
- Find out how many students there are
  - `SELECT count(*) FROM students;`
- Add yourself into the students table
  - `INSERT INTO students (firstname, lastname, graduation_year) values ('Sara','Sprenkle',1999);`
- Add a non-CSCI397 friend into the table
- Check how many students there are now

Practicing SQL

- How many students are graduating this year?
  - Start with this to make sure you're working with the correct data.
  - `SELECT * FROM students WHERE graduation_year = 2017;`
- What information is in the other tables?

`http://cswiki.wlu.edu/dokuwiki/doku.php/labs/postgresql`

Practicing SQL

- Update the studentstodegrees and studentstomajors tables with your info
  - `INSERT INTO studentstodegrees values (studentid, degreesid);`

Practicing SQL

- Update the studentstodegrees and studentstomajors tables with your info
  - `INSERT INTO studentstodegrees values (studentid, degreesid);`
- Subqueries:
  - `INSERT INTO studentstodegrees values ( (SELECT id FROM students WHERE lastname='Sprenkle'), (SELECT id FROM degrees WHERE abbreviation = 'BS') );`

SQL scripts

- We can create .sql files that contain commands and run them on the database
- Example in handouts/postgresql_practice
- `psql cs397 < practice.sql`