Intro to Computer Science

Previous

- Processing

Next

- SQL
  - select
  - join

Site	URL
SQL Zoo	http://sqlzoo.net/wiki/SQL_Tutorial
SQLite	https://docs.python.org/3.4/library/sqlite3.html

SQL

- Structured query language (SQL) is a way of extracting information from relational databases
  - query
  - define
  - modify
- Allows the user to specify what data they want, not necessarily how
- Several standards and implementations
  - ANSI SQL, SQL-92, SQL-99, SQL-03, vendor specific, …
  - *Concepts* remain the same

SQL: Concepts

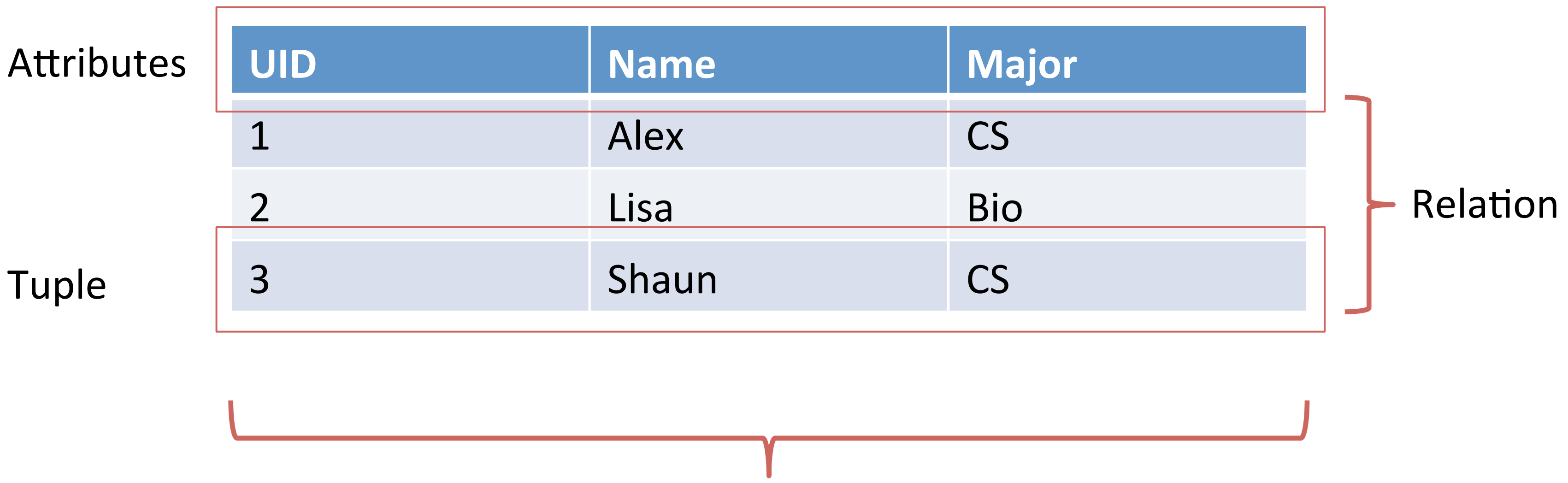
- SQL is predicated on relational algebra
- If you understand relational algebra, SQL will become easy *and make sense*

Relational databases

- A *tuple* is a set of data
  - student = (name, id, major)
- A *relation* is a set of tuples
  - student_1, student_2, …, student_3
- A *database* is a set of relations

- Attributes give meaning to data within a tuple
- *Relational algebra* is a way to combine and refine the contents of one or more relations

Relational database



UID	Name	Major
1	Alex	CS
2	Lisa	Bio
3	Shaun	CS

Collections of relations are a database

Concepts

- Relational algebra has some fundamental concepts:
  - selection
  - projection
  - joins

Selection

- Selection is a means of gathering tuples that satisfy a given predicate
- Produces a new relation, which is a subset of the starting relation(s)

Selection

UID	Name	Major
1	Alex	CS
2	Lisa	Bio
3	Shaun	CS

Select students whose first name is ‘Alex’

UID	Name	Major
1	Alex	CS

Projection

- Projection is a means of refining the result set to only contain specified attributes
- Whereas selection operates of tuples, projection operates over attributes

Projection

UID	Name	Major
1	Alex	CS
2	Lisa	Bio
3	Shaun	CS

Select all the data, but only show the UID and name

UID	Name
1	Alex
2	Lisa
3	Shaun

Can be combined with selection

Select students whose first name is 'Alex', but only give me the name and major

Name	Major
Alex	CS

In SQL

- SELECT [attributes]
  - SELECT uid, name, major
  - SELECT *
- FROM [relations]
  - FROM students
- WHERE [predicates]
  - WHERE uid > 2 AND major = ‘CS’
- GROUP BY [attribute]
- HAVING [predicate]
- ORDER

- Predicate gotchas
  - Not equal is ‘<>’
  - Special comparisons for NULL values
    - a NOT NULL
    - a IS NULL
  - String values go between single quotes (‘ ’), not double quotes (“ ”)

Multiple relations

- Sometimes we need to separate data
  - For organizational/philosophical reasons
  - For practicality (to avoid duplication)
- Relation databases allow us to do so
  - And to combine the data as needed

Joins

- Combining relations is known as a *join*
- There are three common joins
  1. Cross join
  2. Inner join
  3. Outer join

Cross join

- The *cross join* combines two or more relations
  - Sometimes referred to as the *Cartesian product*

UID	Name	Major
1	Alex	CS
2	Lisa	Bio
3	Shaun	CS

PID	Name	Dept
1	Bob	EE
2	Paula	Chem

Cross join

- Combine every tuple from one set with every tuple from the other
- This can be done with an arbitrary number of relations (tables) and an arbitrary number of tuples (rows)!

UID	Name	Major	PID	Name	Dept
1	Alex	CS	1	Bob	EE
1	Alex	CS	2	Paula	Chem
2	Lisa	Bio	1	Bob	EE
2	Lisa	Bio	2	Paula	Chem
3	Shaun	CS	1	Bob	EE
3	Shaun	CS	2	Paula	Chem

Inner join

- Cross joins give no regard to the whether rows “should” be combined
  - Inner joins allow us to specify the linking data
- Concerned with tuples that have matching attributes

Natural inner join

- The natural join of R_1 and R_2 is the cross join in which the common attributes in R_1 and R_2 are equal

UID	Name	Dept
1	Alex	CS
2	Lisa	Bio
3	Shaun	EE
4	Hillary	ME

Dept	Head
CS	Lucy
Chem	Diane
Bio	Roger

UID	Name	Dept	Head
1	Alex	CS	Lucy
2	Lisa	Bio	Roger

Equi-join

- The natural join of R_1 and R_2 is the cross join in which the specified attributes in R_1 and R_2 are equal

UID	Name	Major
1	Alex	CS
2	Lisa	Bio
3	Shaun	EE
4	Hillary	ME

ID	Name	Dept
3	Lucy	CS
4	Diane	Chem
5	Roger	Bio

UID	Name	Major	Dept	Head
3	Shaun	EE	CS	Lucy
4	Hillary	ME	Chem	Diane

Outer join

- Concerned with tuples that *do not* have matching similar attributes
  - A much different concept to the inner joins
- We introduce the existence of a NULL value

Left-outer join

- The left-outer join is
  - The natural join of relations R_1 and R_2
  - Tuples in R_1 that have no matching tuple in R_2

UID	Name	Dept
1	Alex	CS
2	Lisa	Bio
3	Shaun	EE
4	Hillary	ME

Head	Dept
Lucy	CS
Diane	Chem
Roger	Bio

UID	Name	Dept	Head
1	Alex	CS	Lucy
2	Lisa	Bio	Roger
3	Shaun	EE	NULL
4	Hillary	ME	NULL

Right-outer join

- The left-outer join is
  - The natural join of relations R_1 and R_2
  - Tuples in R_2 that have no matching tuple in R_1

UID	Name	Dept
1	Alex	CS
2	Lisa	Bio
3	Shaun	EE
4	Hillary	ME

Name	Dept
Lucy	CS
Diane	Chem
Roger	Bio

UID	Name	Dept	Head
1	Alex	CS	Lucy
NULL	NULL	Chem	Diane
2	Lisa	Bio	Roger

Full-outer join

- Combines the results of the left and right joins
- The full-outer join is
  - The natural join of relations R_1 and R_2
  - Tuples in R_1 that have no matching tuple in R_2
  - Tuples in R_2 that have no matching tuple in R_1
- Note: not actually supported in MySQL
  - Must use the union of the right and left outer joins

Full-outer join

UID	Name	Dept
1	Alex	CS
2	Lisa	Bio
3	Shaun	EE
4	Hillary	ME

Name	Dept
Lucy	CS
Diane	Chem
Roger	Bio

UID	Name	Dept	Head
1	Alex	CS	Lucy
2	Lisa	Bio	Roger
3	Shaun	EE	NULL
4	Hillary	ME	NULL
NULL	NULL	Chem	Diane