Tutorial Week 3

Part 1: Relational Databases and Relational Algebra Queries

Consider the following relations:

Accounts

acctNo	type	balance
12345	savings	12000
23456	chequing	1000
34567	savings	0

Customers

firstName	lastName	idNo	account
Eugene	Krabs	420-699	12345
Pearl	Krabs	805-123	12345
Pearl	Krabs	805-123	23456

Indicate the following:

1. The attributes of each relation
2. The tuples of each relation
3. The components of one tuple from each relation
4. The relation schema for each relation
5. The database schema
6. A suitable domain for each attribute
7. Another equivalent way to present each relation

Consider the following database consisting of the following four relations:

Product(maker, model, type)

PC(mode, speed, ram, hd, price)

Laptop(model, speed, ram, hd, screen, price)

Printer(mode, color, type, price)

Write relational algebra expressions for the following queries. Assume for convenience purposes that the model numbers are unique across all the different manufacturers and across all product types

1. Find all PC models that have a speed of at least 3.00
2. Find all manufacturers that make laptops with a hard disk of at least 100GB
3. Find the model number and price of all products (of any type) made by manufacturer B
4. Find the model numbers of all colour laser printers
5. Find the manufacturers that sell laptops but not PCs
6. Find those hard-disk sizes that occur in two or more PCs
7. Find those pairs of PC models that have both the same speed and RAM. A pair should be listed only once (e.g. if you list the pair (i,j) do not list the pair (j,i))
8. Find those manufacturers of at least two different computers (PCs or laptops) with speeds of at least 2.80
9. Find the manufacturers the computer (PC or laptop) with the highest available speed
10. Find the manufacturer of PCs with at least three different speeds
11. Find the manufacturers who sell exactly three different models of PC

Part 2: SQL Queries

SQLite Documentation and Download:
https://www.sqlite.org/index.html

The SELECT statement

The SELECT statement is probably the most used statement in SQL. It is used to select data from a **database**. The data returned from the query is stored in a result table that is referred to as the **result-set**

Basic syntax:

```
SELECT column1, column2, ...
FROM table1, table2, ...
[WHERE clause]
[OFFSET m] [LIMIT n]

-- selecting individual columns
SELECT column1, column2, ...
FROM table_name

-- wildcard (selects all columns of a table)
SELECT * FROM table_name
```

Some additional notes:

- You can select from more than one table using comma separated after the keyword `FROM`
- The `WHERE` clause is an optional part of the `SELECT` query that can be used to specify various conditions to select by
- You can specify an offset using the optional `OFFSET` attribute
    - Doing this will make the `SELECT` query return records offset by the given value (i.e. after the given value)
    - By default, the offset starts at 0
- You can limit the number of returns using the optional `LIMIT` attribute
    - This is particularly useful if you are selecting a very large result-set which would result in a very slow query

Exercises

Suppose we have the following relational database:

```
CREATE TABLE branch(branch_name, branch_city, assets)
CREATE TABLE customer(customer_name, customer_street, customer_city)
CREATE TABLE loan(loan_number, branch_name, amount)
CREATE TABLE borrower(customer_name, loan_number)
CREATE TABLE bank_account(account_number, branch_name, balance)
CREATE TABLE depositor(customer_name, account_number)
```

Complete the following tasks:

1. List the appropriate primary keys for each table
2. Given your choice of primary keys, identify the appropriate foreign keys
3. Find the names of all branches located in 'Chicago'
4. Find the names of all borrowers who have a loan branch in 'Down-town'
5. Find all loan numbers with a loan value greater than $10,000
6. Find the names of all depositors who have an account with a value greater than $6000

In your own time, play this incredibly fun game with a friend where one person makes up a query and have the other person write relational algebra to retrieve the data