Week 5 SQL

Command-line

1. `mysql`

```
(base) ishtar@eduroam-224-179 ~ % mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 9
Server version: 11.3.2-MariaDB Homebrew

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

2. `SHOW DATABASES;` The prompt gives you a list of databases to which you have access

```
MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| cache              |
| census             |
| data               |
| elections          |
| homebrew           |
| information_schema |
| mysql              |
+--------------------+
7 rows in set (0.001 sec)

MariaDB [(none)]>
```

3. `USE xx;` Use one database

```
MariaDB [(none)]> USE elections;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [elections]>
```

4. `SHOW TABLES;` show the tables in the selected database

```
MariaDB [elections]> SHOW TABLES;
+--------------------+
| Tables_in_elections |
+--------------------+
| Candidate          |
| Party              |
| Ward               |
+--------------------+
3 rows in set (0.001 sec)

MariaDB [elections]>
```

5. `SELECT * FROM (xx table);` shows you the data from the table

```
MariaDB [elections]> SELECT * FROM Party;
+----+-----------------------------------------+
| id | name                                    |
+----+-----------------------------------------+
|  7 | Conservative                            |
|  4 | Green                                   |
|  3 | Independent                             |
|  8 | Independents for Bristol                |
|  6 | Labour                                  |
|  2 | Liberal Democrat                        |
|  9 | Respect                                 |
|  1 | Trade Unionists and Socialists Against Cuts |
|  5 | UKIP                                    |
+----+-----------------------------------------+
9 rows in set (0.001 sec)
```

6. `DESCRIBE (xx table)` Show a list of columns and types in the Party table

```
MariaDB [elections]> DESCRIBE Party;
+-------+-------------+------+-----+---------+----------------+
| Field | Type        | Null | Key | Default | Extra          |
+-------+-------------+------+-----+---------+----------------+
| id    | int(11)     | NO   | PRI | NULL    | auto_increment |
| name  | varchar(100) | NO   | UNI | NULL    |                |
+-------+-------------+------+-----+---------+----------------+
2 rows in set (0.002 sec)
```

7. `SHOW CREATE TABLE Candidate;` shows (more or less) the statement used to create the table

INNER JOIN

WHERE ... AND ...

Entity-Relationship (ER) diagram

a type of structural diagram used in database design to **visually** represent the data or information structure of a system

Components of an ER Diagram:

1. **Entities**:
   the objects or concepts that hold data in the database. (rectangles)
2. **Attributes**:
   the data stored about each entity.
   e.g. "Person" -- attributes --> ID, Name, and Address. (ovals)
3. **Relationships**:
   illustrate how entities are related to one another. (diamonds)
   - one-to-one, one-to-many, or many-to-many, indicating how many instances of one entity can be associated with instances of another entity.
4. **Primary Keys (PK)**:
   a unique attribute or **combination** of attributes that uniquely identifies each instance of an entity
5. **Foreign Keys (FK)**:
   an attribute in one entity that links to the primary key of another entity, establishing a relationship between the two entities.

CREATE/DROP Script

e.g.

Q11. Bob wants to store a table containing students answers in an SQL database. The table should have 3 attributes:

- `student` (an 8 digit number which references the `id` attribute of the `student` table)
- `question` (a 2 digit number which references the `id` attribute of the `question` table)
- `answer` (text containing their answer)

What SQL command would create the most appropriate table?

```
CREATE TABLE answers(student VARCHAR(8), question INTEGER, answer TEXT, FOREIGN
KEY student REFERENCES student(id), FOREIGN KEY question REFERENCES
question(id), PRIMARY KEY (student, question));
```

Q10. Bob wants to store a table containing rankings of different flavours of soup. The table should have 3 attributes:

- `flavor` (text referencing `name` attribute of the flavour table)
- `ranking` (a number)
- `notes` (optional text containing notes)

What SQL command would create the most appropriate table?

```
CREATE TABLE soupranking (flavor TEXT PRIMARY KEY, ranking INTEGER NOT NULL,
notes TEXT, FOREIGN KEY flavor REFERENCES flavor(name));
```

Key Value

```
MariaDB [data]> DESCRIBE Event
    -> ;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| Date        | date         | NO   | PRI | NULL    |       |
| Location    | varchar(255) | NO   | PRI | NULL    |       |
| Name        | varchar(255) | YES  |     | NULL    |       |
| Description | text         | YES  |     | NULL    |       |
| Organizer   | varchar(255) | NO   | MUL | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
5 rows in set (0.004 sec)
```

- `PRI`
  a composite key consisting of `Date` and `Location`.
  enforces uniqueness, meaning no two rows can have the same combination of `Date` and `Location`.
- `MUL`
  means that there is an index on this column (likely due to a foreign key constraint), but the same `Organizer` value can appear in multiple rows.
  This would make sense since one organizer could potentially organize multiple events, hence the foreign key to the `Member` table would not be unique across the `Event` table.

```
+-------+--------------+------+-----+---------+----------------+
| Field | Type         | Null | Key | Default | Extra          |
+-------+--------------+------+-----+---------+----------------+
| id    | int(11)      | NO   | PRI | NULL    | auto_increment |
| name  | varchar(100) | NO   | UNI | NULL    |                |
| party | int(11)      | YES  | MUL | NULL    |                |
| ward  | int(11)      | YES  | MUL | NULL    |                |
| votes | int(11)      | YES  |     | NULL    |                |
+-------+--------------+------+-----+---------+----------------+
```

The first two columns tell you the names and types of columns in this table. The third column (Null) tells you if NULL values are allowed in this column. The Key column tells us that id is the primary key (PRI), name has a unique constraint (UNI), party and ward are foreign keys (MUL) and there are no key constraints at all on votes.

Relational Modelling

refers to the process of structuring and organizing data according to the relational model

Syntax _Keyword

1. `DISTINCT`
   used to remove duplicate rows from a result set. It's applied in the `SELECT` statement, immediately after `SELECT`, to ensure that the query returns only unique rows for the specified columns.
2. `%`
   In SQL, the percent symbol (`%`) is used as a **wildcard character** in `LIKE` clauses to *represent zero, one, or multiple characters in a string*.
   The `LIKE` operator is used in a `WHERE` clause to search for a **specified pattern** in a column.

e.g.

```
SELECT * FROM table WHERE column LIKE '%test';
-- This will find any rows where the column ends with 'test'.

SELECT * FROM table WHERE column LIKE 'test%';
-- This will find any rows where the column starts with 'test'.

SELECT * FROM table WHERE column LIKE '%test%';
-- This will find any rows where the column contains 'test' anywhere in the
string.
```

Normal Forms

Stages in the *normalization process* of database design aimed at reducing data redundancy and improving data integrity. **Normalization** involves ==decomposing tables to minimize duplication ==and designing them such that dependencies are properly enforced.

1. 1NF
   Each column shall contain one (and only one) value
2. 2NF
   a situation where a non-key attribute depends only on part of the composite key rather than the whole key

3. 3NF
   3NF requires that all non-key attributes are directly dependent on the primary key and not on any other non-key attribute

Practice Review

Practice Review for SQL

SQL injection

SQL injection is a **security vulnerability** that allows an attacker to interfere with the queries that an application makes to its database. It is one of the most common web hacking techniques.

- It can allow attackers to fake identity, tamper with existing data, cause repudiation issues such as voiding transactions or changing balances, allow the complete disclosure of all data on the system, destroy the data or make it otherwise unavailable, and become administrators of the database server.

SQL injection happens **when a developer accepts user input that is directly placed into a SQL statement and doesn't properly filter out dangerous characters**. This can allow an attacker to inject their own SQL commands into the SQL statement being executed by the database server.

JDBC

JDBC - Java Database Connectivity is a Java API that manages connecting to a database, issuing queries and commands, and also handling the results returned from the