An Informal Introduction to Queries in SQL

Tables

- Data in a relational database lives in tables:
  - Tables, also called relations, usually have a name
  - Rows, also called tuples, contain one data point
  - Each row is divided into columns giving different attributes of the data point
    - Each column has its own data type and name
    - All rows have the same set of columns
  - Tables are just one kind of object that lives in a relational database.

Tables

- Data in a relational database lives in tables:
  - Tables, also called relations, usually have a name
  - Rows, also called tuples, contain one data point
  - Each row is divided into columns giving different attributes of the data point
    - Each column has its own data type and name
    - All rows have the same set of columns
  - Tables are just one kind of object that lives in a relational database.

Table Example

- In the "cs403" database (the one we’ll use for this class), there is a table named mines_cs_faculty

Here is some sample data from mines_cs_faculty:

<table>
<thead>
<tr>
<th>name</th>
<th>office</th>
<th>email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mehta, Dinesh</td>
<td>BB 280J</td>
<td><a href="mailto:dmehta@mines.edu">dmehta@mines.edu</a></td>
</tr>
<tr>
<td>Paone, Jeffrey</td>
<td>BB 280C</td>
<td><a href="mailto:jpaone@mines.edu">jpaone@mines.edu</a></td>
</tr>
<tr>
<td>Fisher, Wendy</td>
<td>BB 280D</td>
<td><a href="mailto:wfisher@mines.edu">wfisher@mines.edu</a></td>
</tr>
<tr>
<td>Rader, Cynthia</td>
<td></td>
<td><a href="mailto:crader@mines.edu">crader@mines.edu</a></td>
</tr>
</tbody>
</table>

Getting Data with SQL: SELECT

To get the contents of a table using SQL, you use a SELECT query:

```
SELECT * FROM mines_cs_faculty;
```

We’ll demonstrate this now using psql.

Note, the semicolon at the end of the query is required in psql, but not in SQuirreL.

Ordering

- In the relational model, there is no intrinsic ordering of rows in a table.
  - Unless you specify an order, the database is allowed to give you rows in any order
  - Typically order has to do with how the data is stored, and won’t change until the data is modified
  - You can specify an order with the ORDER BY clause:
    ```
    SELECT * FROM mines_cs_faculty ORDER BY name;
    ```
Filtering in SQL: WHERE

Often we only want one or a selection of rows, not all rows from a table. In SQL, this is achieved using a WHERE clause.

Suppose we only want Dr. Mehta's information:

```
SELECT * FROM mines_cs_faculty
WHERE name = 'Mehta, Dinesh';
```

<table>
<thead>
<tr>
<th>name</th>
<th>office</th>
<th>email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mehta, Dinesh</td>
<td>BB 280J</td>
<td><a href="mailto:dmehta@mines.edu">dmehta@mines.edu</a></td>
</tr>
</tbody>
</table>

(1 row)

WHERE Condition

- The WHERE condition can be any logical expression on column names:
  - `name != 'Han, Qi'`
  - `name <> 'Han, Qi'` (same as above)
  - `office >= 'BB 300'`
  - `1 = 1`

- Conditions can also be compound using AND, OR, NOT:
  - `office >= 'BB 300' AND name < 'D'`

Projecting in SQL

Often we only want some columns, not all. The SELECT clause lets us list what we want:

```
SELECT name, email FROM mines_cs_faculty;
```

<table>
<thead>
<tr>
<th>name</th>
<th>email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Han, Qi</td>
<td><a href="mailto:qhan@mines.edu">qhan@mines.edu</a></td>
</tr>
<tr>
<td>Painter-Wakefield, Christopher</td>
<td><a href="mailto:cpainter@mines.edu">cpainter@mines.edu</a></td>
</tr>
<tr>
<td>Paone, Jeffrey</td>
<td><a href="mailto:jpaone@mines.edu">jpaone@mines.edu</a></td>
</tr>
<tr>
<td>Wang, Hua</td>
<td><a href="mailto:huwang@mines.edu">huwang@mines.edu</a></td>
</tr>
<tr>
<td>Yang, Dejun</td>
<td><a href="mailto:djyang@mines.edu">djyang@mines.edu</a></td>
</tr>
</tbody>
</table>

Putting It Together

We can use SELECT, WHERE, ORDER BY, and more all in one query:

```
SELECT name
FROM mines_cs_faculty
WHERE office >= 'BB 300'
AND name < 'D'
ORDER BY name;
```

<table>
<thead>
<tr>
<th>name</th>
<th>office</th>
<th>email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baldwin, Mark</td>
<td>BB 312</td>
<td><a href="mailto:baldwin@mines.edu">baldwin@mines.edu</a></td>
</tr>
</tbody>
</table>

(1 row)

NULL

- NULL is a special value in SQL:
  - It represents the absence of any value
  - It can symbolize many things:
    - The data is unknown
    - The data is missing
    - The data is irrelevant
- In `mines_cs_faculty`, Dr. Rader's office entry is NULL:

<table>
<thead>
<tr>
<th>name</th>
<th>office</th>
<th>email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rader, Cynthia</td>
<td></td>
<td><a href="mailto:crader@mines.edu">crader@mines.edu</a></td>
</tr>
</tbody>
</table>

Note: unfortunately, NULL looks like an empty string in most query tool results.

More About Nothing (NULL)

- NULL cannot be compared, e.g.
  - `WHERE office = NULL` yields NULL rather than true or false
- Note that NULL is neither true nor false
  - `TRUE AND NULL` is NULL
  - `TRUE OR NULL` is TRUE
  - `FALSE AND NULL` is FALSE
  - `FALSE OR NULL` is NULL
  - NULL in most expressions gives back NULL:
    - `X + NULL` → NULL
- To detect in a condition, use IS NULL or IS NOT NULL:
  - `SELECT * FROM mines_cs_faculty WHERE office IS NULL;`
Other Tables To Explore

- mines_courses, mines_courses_meetings – data about courses at Mines in a previous semester
- department, project, employee, etc. – data used in the textbook examples

Up Next

- Next lecture: SQL functions and operators