SQL Intro

Class outline:

- Databases
- SQL
- Querying tables

Databases

Database Management Systems

Database management systems (DBMS) are important, heavily used, and interesting!

Popular DBMSes: MySQL, Oracle, PostgreSQL, MS SQL Server, MongoDB, Redis

A DBMS allows for:

- Storing data in a structured way
- Updating that data
- Querying that data
- Optimizing all of those operations

Relational databases

A relational database consists of tables with data that is often related to each other.

For example, sections.cs61a.org uses MySQL DBMS and has these tables: `attendance`, `course_config`, `section`, `session`, `slot`, `survey`, `user`, `user_session_junction`.

section

id	capacity	staff_id	tag_string
145	35	142	Regular
146	36	188	Zoom
147	36	144	Scholars
148	45	145	Transfer
149	45	174	Regular

user

id	email	name	section_id
192	ana_kerluke@berkeley.edu	Ana Kerluke	149
255	paige_wintheiser@berkeley.edu	Paige Wintheiser	149
270	leanna.feest@berkeley.edu	Leanna Feest	149
387	marcelo35@berkeley.edu	Marcelo Gruno	149
401	baron95@berkeley.edu	Baron Weiss	149

Table terminology

A **table** is a collection of records, which are rows that have a value for each column.

A **row** has a value for each column.

A **column** has a name and a type.

section

id	**capacity**	**staff_id**	**tag_string**
145	35	142	Regular
146	36	188	Zoom
147	36	144	Scholars
148	45	145	Transfer
149	45	174	Regular

How many rows does that table have?
How many columns?

Table terminology

A **table** is a collection of records, which are rows that have a value for each column.

A **row** has a value for each column.

A **column** has a name and a type.

section

id	**capacity**	**staff_id**	**tag_string**
145	35	142	Regular
146	36	188	Zoom
147	36	144	Scholars
148	45	145	Transfer
149	45	174	Regular

How many rows does that table have? 5
How many columns?

Table terminology

A **table** is a collection of records, which are rows that have a value for each column.

A **row** has a value for each column.

A **column** has a name and a type.

section

id	capacity	staff_id	tag_string
145	35	142	Regular
146	36	188	Zoom
147	36	144	Scholars
148	45	145	Transfer
149	45	174	Regular

How many rows does that table have? 5
How many columns? 4

SQL

Structured Query Language

SQL is a declarative programming language for working with relational databases.

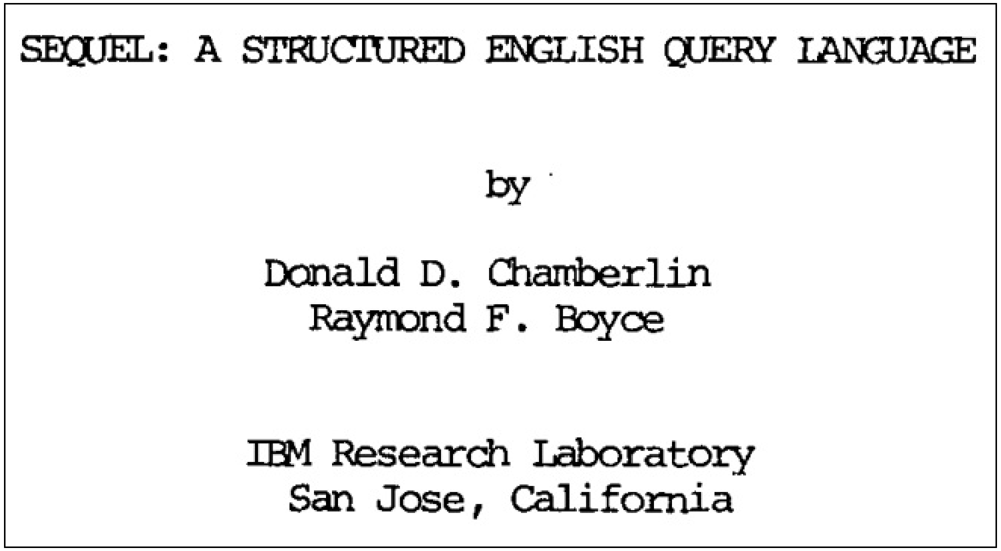
SQL is an ANSI and ISO standard, but DBMS's implement custom variants. We will use SQLite, a lightweight SQL implementation.

SQL variants generally include statements for:

- Querying a table for rows of data (`SELECT ...`)
- Creating new tables (`CREATE ...`)
- Inserting data into tables (`INSERT ...`)
- Updating existing rows in a table (`UPDATE ...`)
- Deleting data from a table (`DELETE ...`)

"S Q L" or "Sequel"?

From the original proposal for the language:



SEQUEL: A STRUCTURED ENGLISH QUERY LANGUAGE

by

Donald D. Chamberlin
Raymond F. Boyce

IBM Research Laboratory
San Jose, California

One trademark dispute later, SEQUEL became SQL.

Pronounce it however you'd like. You do you. 😸

Using SQL

Install sqlite (version 3.8.3 or later):
http://sqlite.org/download.html

Or use SQLite online at code.cs61a.org.

Querying tables

Example table

songs

id	artist	title	release_year	views
1	Lil Nas X	Old Town Road	2018	851
2	Lil Nas X	Panini	2019	423
3	Lil Nas X	Montero (Call Me By Your Name)	2021	391
4	Lil Nas X	Sun Goes Down	2021	35
5	Lil Nas X	Industry Baby	2021	206
6	Lil Nas X	That's What I Want	2021	68
7	Janelle Monáe	Turntables	2021	2
8	Janelle Monáe	Make Me Feel	2018	31
9	Janelle Monáe	Tightrope	2010	28
10	Janelle Monáe	Dirty Computer	2018	3
11	Ben Platt	Grow as We Go	2019	23
12	Ben Platt	Rain	2019	4
13	Ben Platt	Older	2019	6
14	Ben Platt	Imagine	2021	2
15	Ben Platt	I Wanna Love You But I Don't	2021	1
16	Robyn	Dancing with Myself	2010	67
17	Robyn	Indestructible	2011	2
18	Robyn	Hang With Me	2010	7
19	Robyn	Call Your Girlfriend	2011	29
20	P!nk	Try	2012	487
21	P!nk	All I Know So Far	2021	18
22	P!nk	What About Us	2017	320
23	P!nk	Just Like A Pill	2009	137

SELECT statement

The `SELECT` statement queries a database and returns zero or more rows of data.

Return all the rows:

```
SELECT * FROM songs;
```

Return a subset of columns from all rows:

```
SELECT artist, title FROM songs;
```

Rename columns in the returned data:

```
SELECT artist AS singer, title AS song_title FROM songs;
```

Manipulate column values in the returned data:

```
SELECT title, views * 1000000 FROM songs;
```

Try these on code.cs61a.org - the songs table is built-in.

ORDER BY clause

The `SELECT` statement can have multiple optional clauses.

The `ORDER BY` clause specifies the order of rows returned:

Return the rows sorted by a column (highest to lowest):

```
SELECT title, views FROM songs ORDER BY views DESC;
```

Ditto, but ascending (lowest to highest):

```
SELECT title, views FROM songs ORDER BY views ASC;
```

Return the rows sorted by multiple columns:

```
SELECT title, release_year, views  FROM songs
    ORDER BY release_year DESC, views DESC;
```

WHERE clause

The `WHERE` clause can be used to filter rows, and must appear before the `ORDER BY` clause.

```
SELECT [columns] FROM [table] WHERE [condition] ORDER BY [order]
```

Returning rows that match a particular value:

```
SELECT title FROM songs WHERE artist = "Lil Nas X";
```

Using other comparison operators:

```
SELECT artist, title, release_year FROM songs
    WHERE release_year > 2020;
```

Using logical operators:

```
SELECT artist, title, release_year FROM songs
    WHERE release_year > 2020 AND views > 5;
```

Other clauses

The `SELECT` statement supports a wide array of optional clauses. See the full description in the SQLite SELECT documentation.

We will learn a few more clauses in the coming lectures, but not all of them!