Why Learn SQL?

A Day in the Life - Data Analyst

It's Catherine's first day working as a Data Analyst for Codecademy. She's excited to use some of the tools that she learned in **Analyze Data with SQL**, such as:

- Writing basic queries
- Calculating aggregates
- Combining data from multiple tables
- Determining web traffic attribution
- Creating usage funnels
- Analyzing user churn

Exploring Data with SQL

Like most organizations, Codecademy uses **SQL** (**S**tructured **Q**uery **L**anguage) to access its database. A **database** is a set of data stored in a computer. This data is usually structured into *tables*. Tables can grow large and have a multitude of columns and records.

Spreadsheets, like Microsoft Excel and Google Sheets, allow you to view and manipulate data directly: with selecting, filtering, sorting, etc. By applying a number of these operations you can obtain the subset of data you are seeking.

SQL (pronounced "S-Q-L" or "sequel") allows you to write **queries** which define the subset of data you are seeking. Unlike Excel and Sheets, your computer and SQL will handle *how* to get the data; you can focus on *what* data you would like. You can save these queries, refine them, share them, and run them on different databases.

It is a great way to access data and a great entry point to programming because its syntax (the specific vocabulary that gives instructions to the computer) is very human-readable. Without knowing any SQL, you might still be able to guess what each command will do.

On her first day at Codecademy, Catherine wants to become familiar with the company's data, so she connects to the database and uses SQL to explore the database.

1. One of the tables in Codecademy's database is called `browse`. It contains information on each time someone visited the Codecademy's website. Paste the following code into the code editor (middle panel) and click Run.

```sql
SELECT *
FROM browse
LIMIT 10;
```

This code will select all (`*`) columns from `browse` for the first 10 records.

Once you correctly enter the code and click Run, this instruction will turn green, letting you know that you completed this checkpoint.

```
SELECT *
FROM browse
LIMIT 10;

SELECT *
FROM items
LIMIT 10;

SELECT *
FROM checkout
LIMIT 10;

SELECT *
FROM purchase
LIMIT 10;
```

Creating Usage Funnels

Visitors to Codecademy's website follow a simple workflow:

1. Browse items available for sale
2. Click an icon to begin the checkout process
3. Enter payment information to complete their purchase

Not all users who browse on the website will find something that they like enough to checkout, and not all users who begin the checkout process will finish entering their payment information to make a purchase.

This type of multi-step process where some users leave at each step is called a *funnel*.

Catherine wants to determine what percent of users make it through each step of the funnel so that she can recommend improvements to Codecademy's website.

1. Catherine is going to combine data from three different tables:

- `browse` - gives the timestamps of users who visited different item description pages
- `checkout` - gives the timestamps of users who visited the checkout page
- `purchase` - gives the timestamps of when users complete their purchase

Using SQL, she finds that 24% of all users who browse move on to checkout. 89% of those who reach checkout purchase.

Click Run, to see Catherine's analysis.

```
SELECT ROUND(
   100.0 * COUNT(DISTINCT c.user_id) / COUNT(DISTINCT b.user_id)
 ) AS browse_to_checkout_percent,
 ROUND(
   100.0 * COUNT(DISTINCT p.user_id) / COUNT(DISTINCT c.user_id)
 ) AS checkout_to_purchase_percent
 FROM browse b
 LEFT JOIN checkout c
       ON b.user_id = c.user_id
 LEFT JOIN purchase p
       ON c.user_id = p.user_id;
```

Analyzing User Churn

Next, Catherine wants to take a look at the churn rate.

A *churn rate* is the percent of subscribers to a monthly service who have canceled. For example, in January, let's say Codecademy has 1,000 customers. In February, 200 learners sign up, and 250 cancel.

The churn rate for February would be:

$$\frac{cancellations}{total\ subscribers} = \frac{250}{1000 + 200} = 20.8\%$$

Catherine wants to analyze the churn rates for Codecademy for the past few months so she writes another SQL query.

1. Click Run, to see Catherine's analysis for the churn rate in March 2017. What recommendations would you make to Codecademy based on Catherine's analysis? (This query might take some time to load because the `pro_users` table has 118,135 rows!)

```
SELECT COUNT(DISTINCT user_id) AS enrollments,
       COUNT(CASE
              WHEN strftime("%m", cancel_date) = '03'
              THEN user_id
       END) AS march_cancellations,
       ROUND(100.0 * COUNT(CASE
              WHEN strftime("%m", cancel_date) = '03'
              THEN user_id
       END) / COUNT(DISTINCT user_id)) AS churn_rate
FROM pro_users
WHERE signup_date < '2017-04-01'
       AND (
       (cancel_date IS NULL) OR
       (cancel_date > '2017-03-01')
  );
```

Determining Web Traffic Attribution

Catherine's boss asks her to analyze how users are finding Codecademy's websites using *UTM Parameters*. UTM Parameters are special tags that site owners add to their pages to track what website a user was on before they reach the website. For instance:

- If a user found Codecademy's website through Google search, the table `page_visits` might have `utm_source` set to 'google'.
- If a different user clicked a Facebook ad to get to Codecademy's website, then their row in `page_visits` might have `utm_source` as 'facebook'.

1. Catherine wants to know how many visits come from each `utm_source`. Click Run, to see Catherine's analysis. What is the most common source of traffic to Codecademy's website?

```
SELECT utm_source, COUNT(DISTINCT user_id) AS num_users
FROM page_visits
GROUP BY 1
ORDER BY 2 DESC;
```

Begin Your Journey

Catherine learned and applied a lot during the first two weeks of her job. She:

- Wrote basic queries
- Calculated aggregates
- Combined data from multiple tables
- Created usage funnels
- Analyzed user churn
- Determined web traffic attribution

Now it's your turn. By the end of this course, you'll be able to do all of the things that Catherine can do.

Good luck and happy coding!

What is a Relational Database Management System?

What is a Database?

A *database* is a set of data stored in a computer. This data is usually structured in a way that makes the data easily accessible.

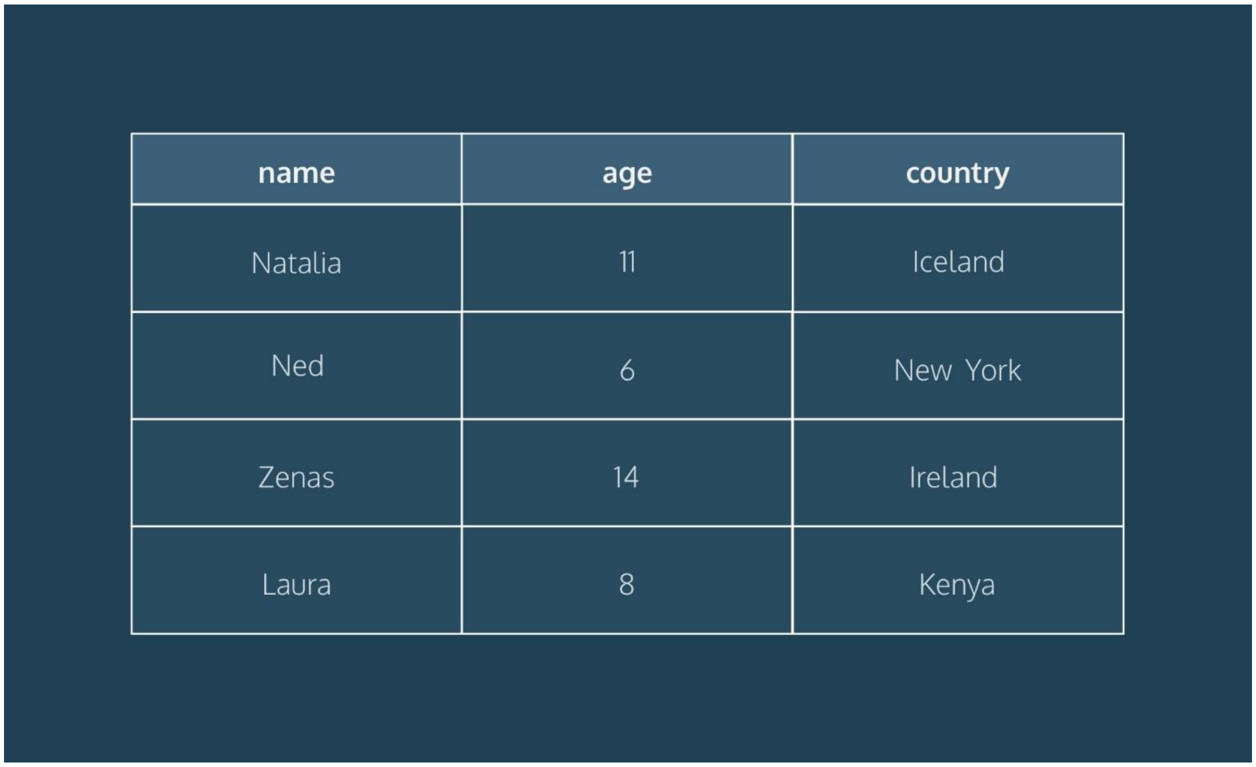
What is a Relational Database?

A *relational database* is a type of database. It uses a structure that allows us to identify and access data *in relation* to another piece of data in the database. Often, data in a relational database is organized into tables.

Tables: Rows and Columns

Tables can have hundreds, thousands, sometimes even millions of rows of data. These rows are often called *records*.

Tables can also have many *columns* of data. Columns are labeled with a descriptive name (say, `age` for example) and have a specific *data type*.
For example, a column called `age` may have a type of `INTEGER` (denoting the type of data it is meant to hold).



name	age	country
Natalia	11	Iceland
Ned	6	New York
Zenas	14	Ireland
Laura	8	Kenya

In the table above, there are three columns (`name`, `age`, and `country`).

The `name` and `country` columns store string data types, whereas `age` stores integer data types. The set of columns and data types make up the schema of this table.

The table also has four rows, or records, in it (one each for Natalia, Ned, Zenas, and Laura).

What is a Relational Database Management System (RDBMS)?

A relational database management system (RDBMS) is a program that allows you to create, update, and administer a relational database. Most relational database management systems use the SQL language to access the database.

What is SQL?

SQL (**S**tructured **Q**uery **L**anguage) is a programming language used to communicate with data stored in a relational database management system. SQL syntax is similar to the English language, which makes it relatively easy to write, read, and interpret.

Many RDBMSs use SQL (and variations of SQL) to access the data in tables. For example, SQLite is a relational database management system. SQLite contains a minimal set of SQL commands (which are the same across all RDBMSs). Other RDBMSs may use other variants.

(SQL is often pronounced in one of two ways. You can pronounce it by speaking each letter individually like "S-Q-L", or pronounce it using the word "sequel".)

Popular Relational Database Management Systems

SQL syntax may differ slightly depending on which RDBMS you are using. Here is a brief description of popular RDBMSs:

MySQL
MySQL is the most popular open source SQL database. It is typically used for web application development, and often accessed using PHP.

The main advantages of MySQL are that it is easy to use, inexpensive, reliable (has been around since 1995), and has a large community of developers who can help answer questions.

Some of the disadvantages are that it has been known to suffer from poor performance when scaling, open source development has lagged since Oracle has taken control of MySQL, and it does not include some advanced features that developers may be used to.

PostgreSQL

PostgreSQL is an open source SQL database that is not controlled by any corporation. It is typically used for web application development.

PostgreSQL shares many of the same advantages of MySQL. It is easy to use, inexpensive, reliable and has a large community of developers. It also provides some additional features such as foreign key support without requiring complex configuration.

The main disadvantage of PostgreSQL is that it is slower in performance than other databases such as MySQL. It is also less popular than MySQL which makes it harder to come by hosts or service providers that offer managed PostgreSQL instances.

Oracle DB

Oracle Corporation owns Oracle Database, and the code is not open sourced.

Oracle DB is for large applications, particularly in the banking industry. Most of the world's top banks run Oracle applications because Oracle offers a powerful combination of technology and comprehensive, pre-integrated business applications, including essential functionality built specifically for banks.

The main disadvantage of using Oracle is that it is not free to use like its open source competitors and can be quite expensive.

SQL Server

Microsoft owns SQL Server. Like Oracle DB, the code is close sourced. Large enterprise applications mostly use SQL Server. Microsoft offers a free entry-level version called *Express* but can become very expensive as you scale your application.

SQLite

SQLite is a popular open source SQL database. It can store an entire database in a single file. One of the most significant advantages this provides is that all of the data can be stored locally without having to connect your database to a server.

SQLite is a popular choice for databases in cellphones, PDAs, MP3 players, set-top boxes, and other electronic gadgets. The SQL courses on Codecademy use SQLite.

For more info on SQLite, including installation instructions, read this article.

Conclusion

Relational databases store data in tables. Tables can grow large and have a multitude of columns and records. Relational database management systems (RDBMSs) use SQL (and variants of SQL) to manage the data in these large tables. The RDBMS you use is your choice and depends on the complexity of your application.