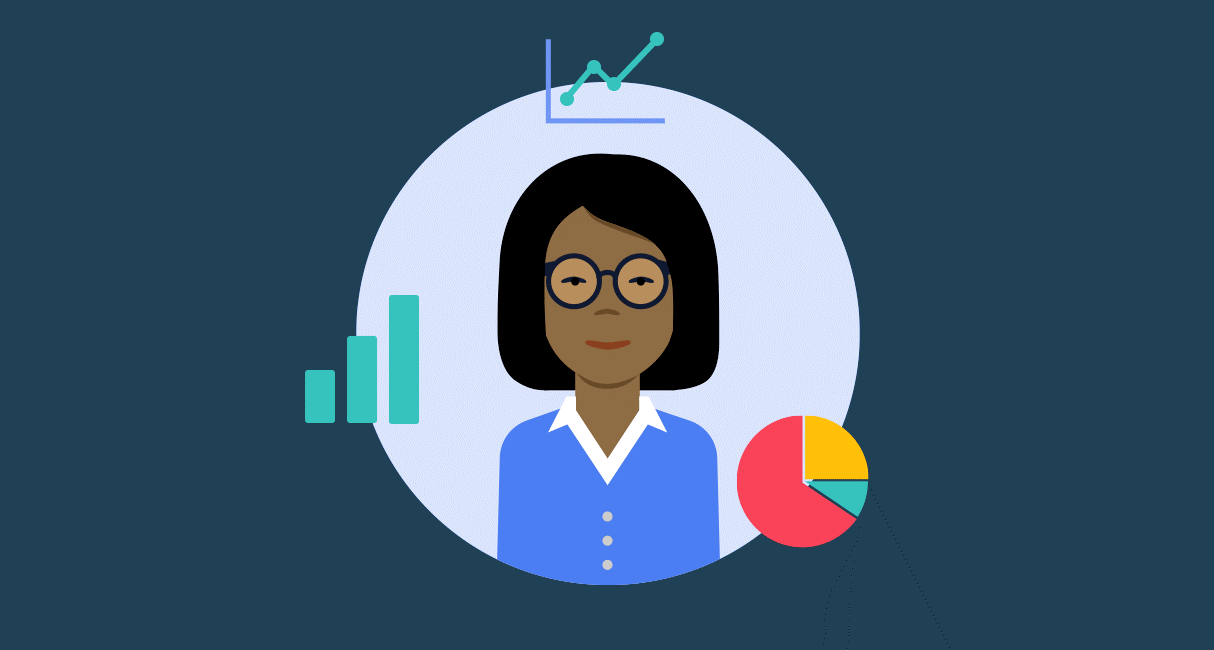
**A Day in the Life - Data Analyst**

It’s Catherine’s first day working as a Data Analyst for ACME. She’s excited to use some of the tools that she learned in **Learn SQL from Scratch**, 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**

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.

Many databases use SQL (**S**tructured **Q**uery **L**anguage). 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 ACME, Catherine wants to become familiar with the company’s data, so she connects to the database and uses SQL to explore the database.

**Instructions**

**1.**

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

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.

Hint

Examine the data in the far right panel:

* What columns are there?
* What kinds of questions do you have about the ACME’s website?

The columns are:

* user\_id
* browse\_date
* item\_id

SELECT \* FROM browse LIMIT 10;

**Creating Usage Funnels**

Visitors to ACME’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 ACME’s website.

**Instructions**

**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.

Hint

In the result, there should be two columns:

* browse\_to\_checkout\_percent
* checkout\_to\_purchase\_percent

 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**

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

The churn rate for February would be:

Catherine wants to analyze the churn rates for ACME for the past few months.

**Instructions**

**1.**

Click Run, to see Catherine’s analysis for the churn rate in March 2017.

What recommendations would you make to ACME based on Catherine’s analysis?

Hint

In the result, there should be three columns:

* enrollments
* march-cancellations
* churn\_rate

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 ACME’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 ACME’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 ACME’s website, then their row in page\_visits might have utm\_source as ‘facebook’.

**Instructions**

**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 ACME’s website?

Hint

The most common source is The New York Times!

 SELECT utm\_source,

  COUNT(DISTINCT user\_id) AS num\_users

FROM page\_visits

GROUP BY 1

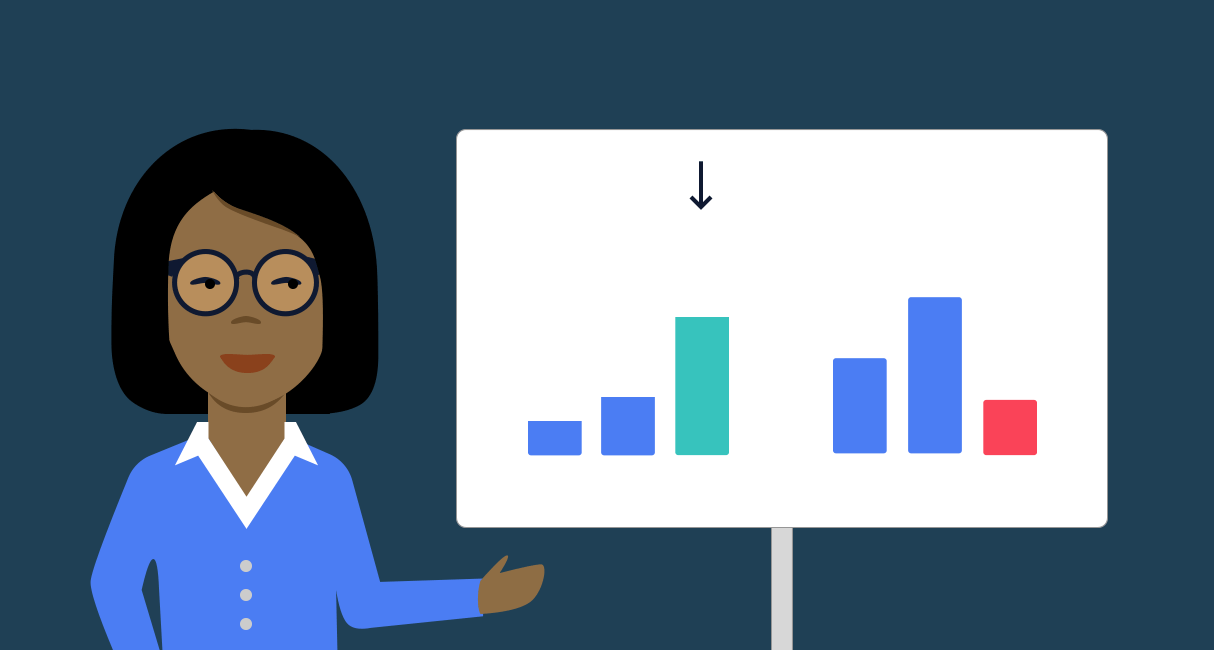
ORDER BY 2 DESC;

# Begin Your Journey

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

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

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



**Introduction**

In this lesson, we will be learning different SQL commands to **query** a single table in a database.

One of the core purposes of the SQL language is to retrieve information stored in a database. This is commonly referred to as querying. Queries allow us to communicate with the database by asking questions and returning a result set with data relevant to the question.

We will be querying a database with one table named movies.

Let’s get started!

Fun fact: IBM started out SQL as SEQUEL (**S**tructured **E**nglish **QUE**ry **L**anguage) in the 1970’s to query databases.

**Instructions**

**1.**

We should get acquainted with the movies table.

In the editor, type the following:

SELECT \* FROM movies;

What are the column names?

Hint

Suppose we want to see all values in a table. We will use the syntax:

SELECT \* FROM table\_name;

1. SELECT \* indicates that we want to *select* *all* columns.
2. FROM table\_name indicates which table we are interested in.
3. ; ends a SQL command.

Take a look at the result table:

* Scroll right to look at the columns.
* Scroll down to look at the rows.

The columns are id, name, genre, year, and imdb\_rating.

SELECT \* FROM movies;

**Select**

Previously, we learned that SELECT is used every time you want to query data from a database and \* means *all* columns.

Suppose we are only interested in two of the columns. We can select individual columns by their names (separated by a comma):

SELECT column1, column2

FROM table\_name;

To make it easier to read, we moved FROM to another line.

Line breaks don’t mean anything specific in SQL. We could write this entire query in one line, and it would run just fine.

**Instructions**

**1.**

Let’s only select the name and genre columns of the table.

In the code editor, type the following:

SELECT name, genre

FROM movies;

Hint

Instead of selecting all the columns:

SELECT \*

FROM movies:

We are now just selecting the name and genre column:

SELECT name, genre

FROM movies;

Double-check your query, character by character:

* There is a comma between name and genre.
* SQL commands end with a ;.

SELECT and FROM are clauses. Clauses perform specific tasks in SQL. By convention, clauses are written in capital letters.

**2.**

Now we want to include a third column.

Edit your query so that it returns the name, genre, *and* year columns of the table.

Hint

The syntax for selecting three individual columns:

SELECT column1, column2, column3

FROM table\_name;

Following this format, the query below selects name, genre, and year columns (in that order) from the movies table:

SELECT name, genre, year

FROM movies;

* We have to separate the column names with a comma.
* SQL commands end with a ;.