**Got It!**

## 1. SQL databases

Great job on these exercises! Let's pursue. We've mentioned SQL several times by now, so how about we spend a bit more time on this language that is so fundamental in data engineering?

## 2. SQL

SQL stands for Structured Query Language. SQL is to databases what English is to pop music. It's the preferred language to query RDBMS or Relational Database Management System - basically systems that gather several tables like the Employees table from the previous lesson, where all tables are related to each other. More on that in a moment. SQL has two main advantages: it allows you to access many records at once, and group, filter or aggregate them. Most programming languages let you do that, but SQL was the first, which is why it's been so influential. It's a little bit like the Beatles and pop music. It's also very close to English, which makes it easy to write and understand. As you already know data engineers use SQL to create and maintain databases, while data scientists use SQL to query databases.

## 3. Remember the employees table

We're not going to learn SQL in this course, nor test you on it: we have great courses and tracks that cover this topic. However, looking at some examples will help your understanding. Let's look at a data engineering example first, creating a table. Take a moment to refresh your memory of Spotflix's employee table. Remember the first columns holds non-decimal numbers, the penultimate one stores logical values, and the others hold text.

## 4. SQL for data engineers

We can create such a table using SQL. We type the command CREATE TABLE, and declare the name of the table, "employees". Then we proceed to create the first column, employee\_id, and specify the type of data expected, integers - which mean this column will only accept whole numbers, without any decimal. We then create the second column, first\_name, and specify it should be text (VARCHAR stands for "variable characters"). Two-hundred fifty-five here means that the value entered can't be more than Two-hundred fifty-five characters long. And we do the same for last name, role and team. We declare full\_time as a Boolean, which is the type for logical values. This column can only hold zero for false or one for true. Office is declared as VARCHAR as well because it's text. Data engineers then run other statements to update the table and write records into it.

## 5. SQL for data scientists

Data scientists will then use SQL to query the table. For example, if Julian wants to get the first and last name of all the employees whose role title contains the keyword data, he can select the first and last name, FROM the employees table, WHERE the role title contains data. The percentage signs on each side of "Data" mean "Data" can appear anywhere in the role title.

## 6. Database schema

So far, we've looked at tables individually; but databases are made of many tables. The database schema governs how tables are related.

## 7. Database - albums

In the Spotflix's database, we have a table for albums, containing columns for the album's unique ID, the artist's unique ID, the title of the album, etc.

## 8. Database - artists

We also have an artist's table, containing columns for the artist unique ID, the artist name and their biography.

## 9. Database - linking albums and artists

The artist table can the linked to the album's table through the artist ID.

## 10. Database - songs

We also have a songs table with columns for the song unique ID, album ID, song title, etc.

## 11. Database - linking albums and songs

The songs table can be linked to the the album's table through the album ID.

## 12. Database - playlists

We also have a playlists table with columns for a playlist unique ID, the ID of the user that created it, the song ID, etc.

## 13. Database - linking songs and playlists

We can link the playlists table to the songs table through the song ID. We could have other tables for labels, genres, users, etc. See why these are called "relational" databases?

## 14. Several implementations

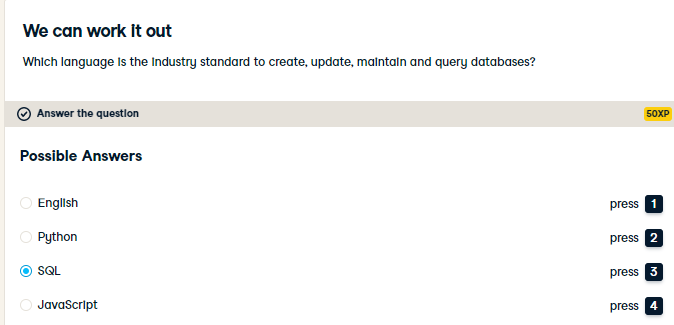
Finally, there are several implementations of SQL. How they differ is out of the scope of this course, but they are pretty similar. Switching from one to the other is like switching from a QWERTY keyboard to an AZERTY one, or switching from British English to American English. A few things change, but most things stay the same.

## 15. Summary

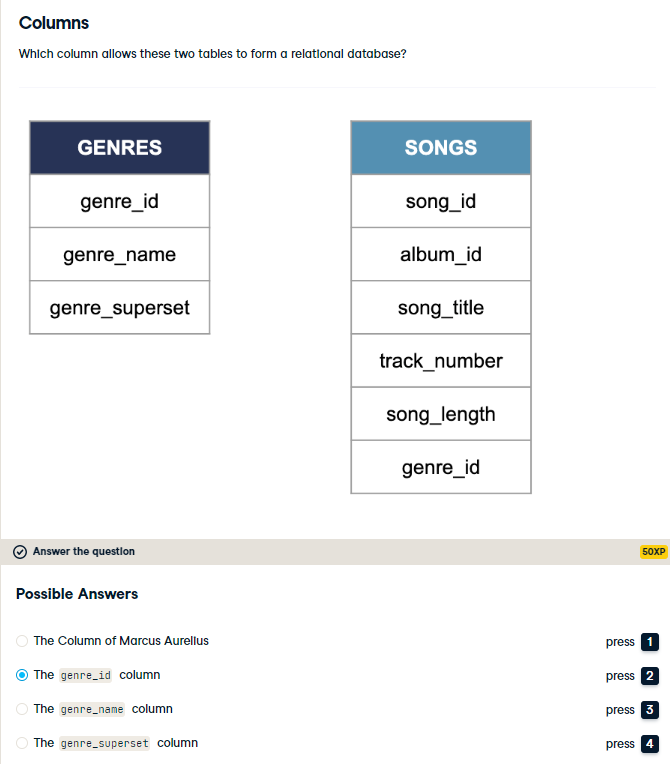
You now understand why SQL is the language or reference for RDBMS, how data engineers and data scientists use it differently, can give an example of a database schema, and can cite several SQL implementations.

## 16. Let's practice!

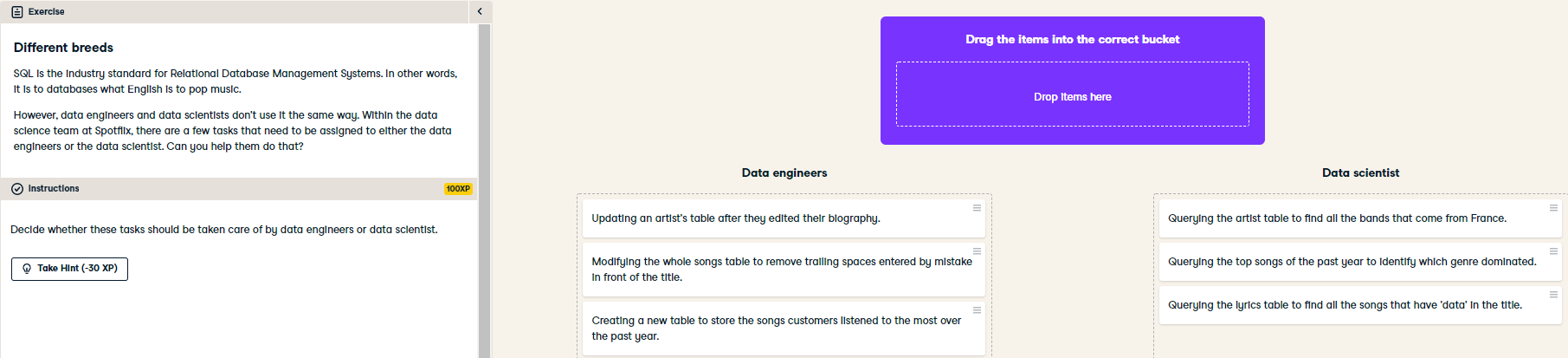
Time for some exercises!



Absolutely! Structured Query Language was developed in the 70s at IBM for the exact purpose of managing and retrieving data from structured/relational databases.



Exactly! The genre\_id column appear in both tables, so it can be used to connect them.



Kaboom! Now you can tell if someone is a data engineer or a data scientist just by asking how they use SQL!