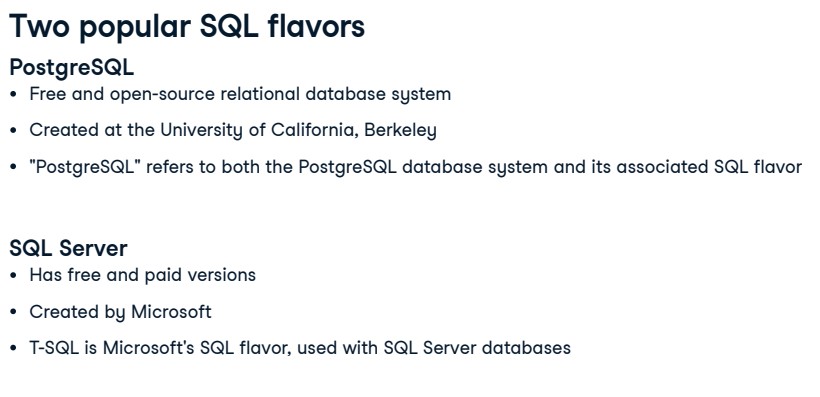
**1. SQL flavors**

Our last topic in this short course is SQL flavors.

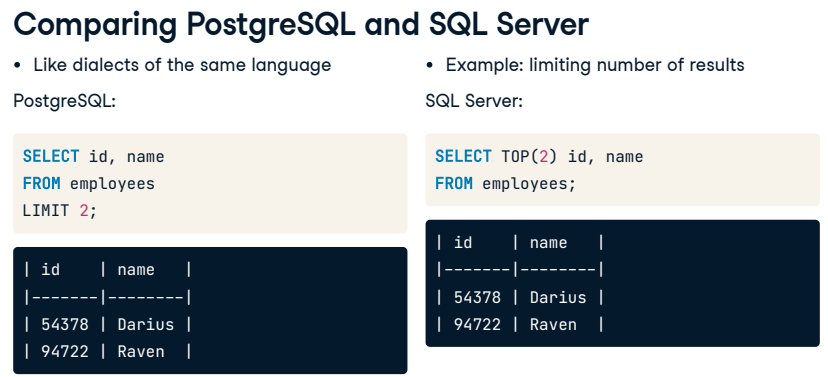
**2. SQL flavors**

SQL has a few different versions, or flavors. Some are free, while others have customer support and are made to complement major databases such as Microsoft's SQL Server or Oracle Database, which are used by many companies. All SQL flavors are used with table-based relational databases like the ones we've seen, and the vast majority of keywords are shared between them! In fact, all SQL flavors must follow universal standards set by the International Organization for Standards and the American National Standards Institute. Only additional features on top of these standards result in different SQL flavors.

**3. Two popular SQL flavors**

Let's take a look at two of the most popular SQL flavors. PostgreSQL is a free and open-source relational database system which was originally created at the University of California, Berkeley, and was sponsored by America's famous Defense Advanced Research Projects Agency, or DARPA. DARPA also sponsored research leading to creating the internet, the computer mouse, and GPS! The name "PostgreSQL" is used to refer to both the database system itself as well as the SQL flavor used with it. SQL Server is also a relational database system which comes in both free and enterprise versions. It was created by Microsoft, so it pairs well with other Microsoft products. T-SQL is Microsoft's proprietary flavor of SQL, used with SQL Server databases.

**4. Comparing PostgreSQL and SQL Server**

Think of SQL flavors as dialects of the same language. If Claudia speaks American English, she will have no trouble understanding people on a trip to London, even though most people in London speak British English and there are some small differences. Here's an example of a small difference between SQL Server and PostgreSQL: when we want to limit the number of records returned, we use the LIMIT keyword in PostgreSQL. Here, we limit the number of employee names and ids selected to only the first two records. The exact same results are achieved in SQL Server using the TOP keyword instead of LIMIT. Notice that this keyword is the only difference between the two queries! Limiting results is useful when testing code, since many result sets can have thousands of results! It's best to write and test code using just a few results before removing the LIMIT for the final query.

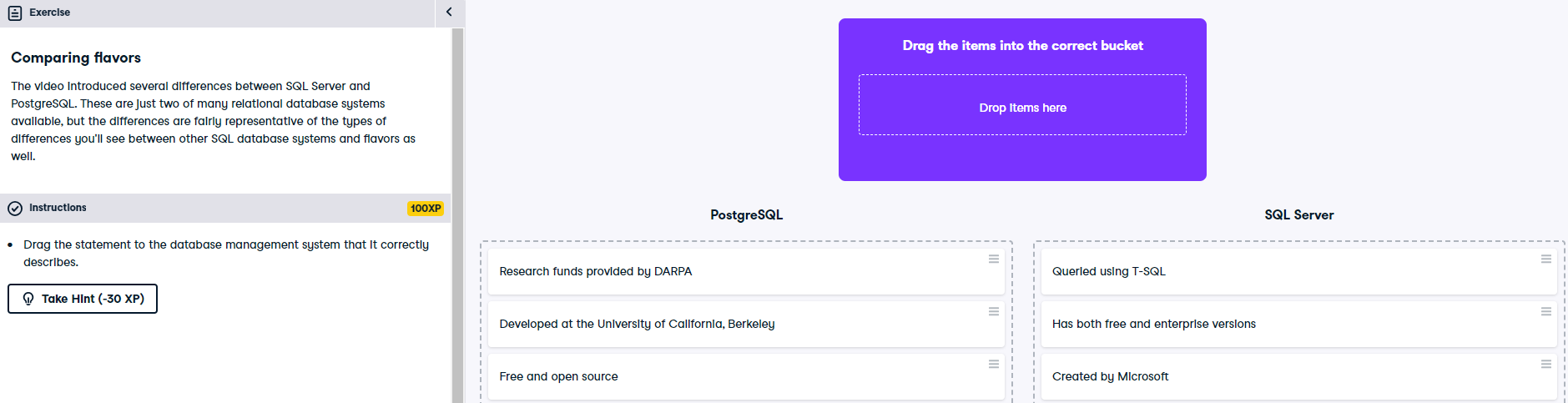
**5. Choosing a flavor**



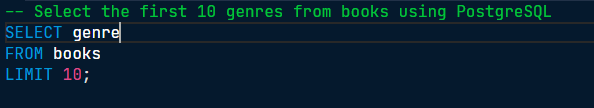
New SQL learners may wonder which flavor they should learn. This may be an easy decision if a learner knows that her employer uses Microsoft's SQL Server, for example. Or it might be a hard one for a job seeker or student who doesn't know what database management system a future employer might use. Don't worry too much about what flavor to learn. As we've seen, the differences are small. A PostgreSQL wizard can become a SQL Server wizard by learning a handful of different keywords!

**6. Let's practice!**

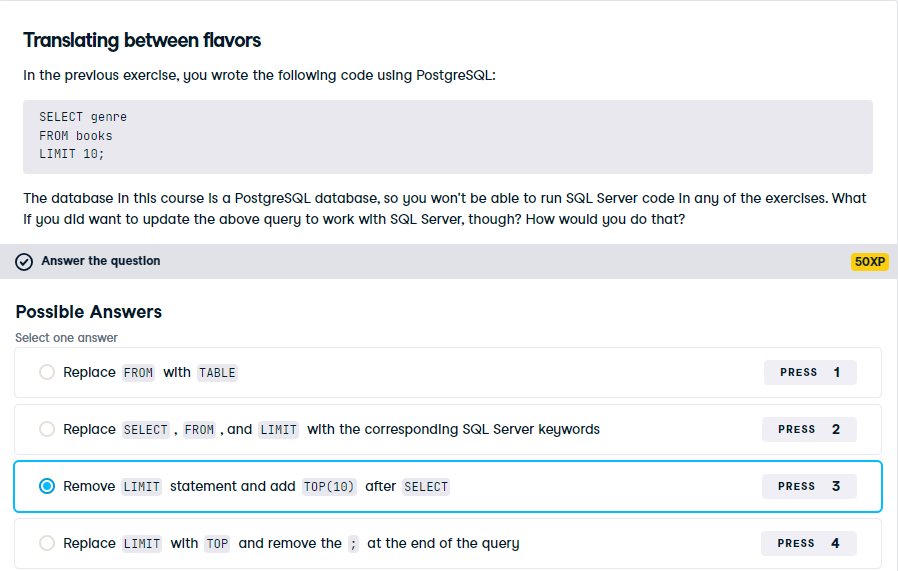
Now that we've sampled a few SQL flavors, let's practice!



Excellent work! Both SQL Server and PostgreSQL are very popular SQL flavors. As your SQL journey continues, you're sure to see their names pop up!



There's no LIMIT to your SQL skills! Great work. You can see from this exercise how it's nice to work with small result sets by limiting the number of results.



Looks like you've got the flavor of things! Great work. Only a few keyword differences exist between PostgreSQL and SQL Server—most syntax is the same!

PostgreSQL and SQL Server are both relational database management systems (RDBMS), but they have some differences in terms of features, licensing, and platform support. Here are some key distinctions:

1. **Licensing:**
   * **PostgreSQL:** Released under the PostgreSQL License, which is a permissive open-source license. It allows users to use, modify, and distribute the software freely.
   * **SQL Server:** Developed by Microsoft and is available in different editions, including a free version called SQL Server Express. However, some editions, like SQL Server Enterprise, require a commercial license.
2. **Platform Support:**
   * **PostgreSQL:** Known for its cross-platform compatibility and is available on various operating systems, including Linux, Windows, and macOS.
   * **SQL Server:** Primarily designed for the Windows platform, although there is a Linux version available. macOS support is limited, and it may be used in certain scenarios.
3. **Community and Development:**
   * **PostgreSQL:** Has a strong open-source community, and development is driven by a global group of contributors. It often leads in terms of supporting the latest SQL standards.
   * **SQL Server:** Developed by Microsoft, which has its own development and support team. It may have features and integrations that are specific to the Microsoft ecosystem.
4. **Performance:**
   * **PostgreSQL:** Known for its extensibility and support for complex queries. It performs well in large and complex datasets.
   * **SQL Server:** Generally performs well and includes various optimization features. Performance can depend on the specific use case and configuration.
5. **Feature Set:**
   * **PostgreSQL:** Offers a broad range of features, including support for JSON data types, advanced indexing, and extensibility through custom functions and procedural languages.
   * **SQL Server:** Provides a comprehensive set of features, including advanced analytics, integration with Microsoft tools, and support for various data types.
6. **Extensions and Stored Procedures:**
   * **PostgreSQL:** Allows the creation of custom functions using multiple programming languages and supports stored procedures.
   * **SQL Server:** Supports stored procedures, triggers, and functions, often using Transact-SQL (T-SQL).
7. **Replication and High Availability:**
   * **PostgreSQL:** Offers various replication methods, including streaming replication and logical replication, to support high availability.
   * **SQL Server:** Provides features like Always On Availability Groups for high availability and disaster recovery.

When choosing between PostgreSQL and SQL Server, it's important to consider factors such as licensing costs, platform requirements, and specific features needed for your project. Both databases are powerful and capable, and the choice often depends on the preferences and requirements of the development team and the organization.

While both PostgreSQL and SQL Server are SQL-compliant relational database management systems, there are some differences in the usage of keywords and syntax. Below are a few examples of differences in keyword usage:

1. **LIMIT/OFFSET vs. TOP:**
   * **PostgreSQL:** Uses **LIMIT** and **OFFSET** for pagination. For example: **SELECT \* FROM table LIMIT 10 OFFSET 20;**
   * **SQL Server:** Uses **TOP** for limiting results. For example: **SELECT TOP 10 \* FROM table;**
2. **Boolean Data Type:**
   * **PostgreSQL:** Uses **BOOLEAN** for boolean data type.
   * **SQL Server:** Uses **BIT** for boolean data type.
3. **Auto-incrementing Columns:**
   * **PostgreSQL:** Uses **SERIAL** or **BIGSERIAL** for auto-incrementing columns.
   * **SQL Server:** Uses **IDENTITY** for auto-incrementing columns.
4. **String Concatenation:**
   * **PostgreSQL:** Uses the **||** operator for string concatenation.
   * **SQL Server:** Uses the **+** operator for string concatenation.
5. **Date and Time Functions:**
   * **PostgreSQL:** Has functions like **CURRENT\_DATE**, **CURRENT\_TIME**, and **CURRENT\_TIMESTAMP** for date and time.
   * **SQL Server:** Uses **GETDATE()** for the current date and time.
6. **String Comparison:**
   * **PostgreSQL:** Case-sensitive by default, but you can use **ILIKE** for case-insensitive comparison.
   * **SQL Server:** Case-insensitive by default, but you can use **COLLATE** for case-sensitive comparison.
7. **Quotes for String Literals:**
   * **PostgreSQL:** Allows both single quotes (**'**) and double quotes (**"**) for string literals.
   * **SQL Server:** Uses single quotes (**'**) for string literals.
8. **Common Table Expressions (CTE):**
   * **PostgreSQL:** Uses **WITH** clause for defining CTEs.
   * **SQL Server:** Also uses **WITH** clause for defining CTEs.
9. **String Functions:**
   * **PostgreSQL:** Uses functions like **LENGTH()**, **POSITION()**, and **SUBSTRING()** for string manipulation.
   * **SQL Server:** Uses functions like **LEN()**, **CHARINDEX()**, and **SUBSTRING()** for similar purposes.

It's important to consult the documentation for each database system when writing queries to ensure that you are using the correct syntax and keywords. While many SQL concepts are standardized, there are still differences in implementation and specific functionalities between different database systems.