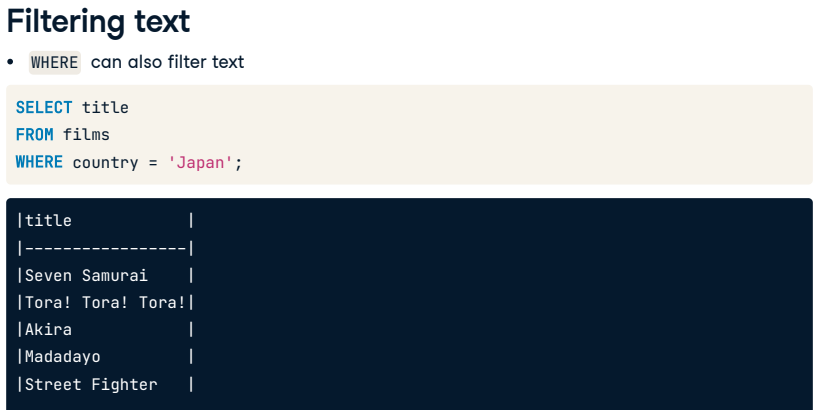
## 1. Filtering text

We're making excellent progress! We will now switch our focus away from filtering numbers to filtering textual data.

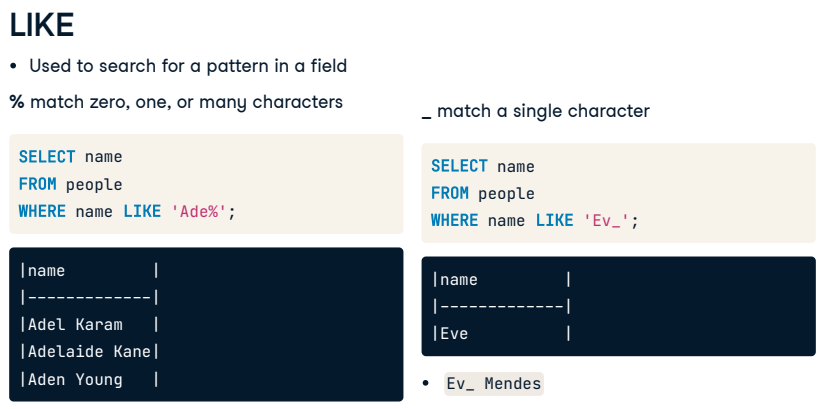
## 2. Filtering text

As we've briefly seen, we can use the WHERE clause to filter text data. However, so far, we've only been able to filter by specifying the exact text we're interested in.

## 3. Filtering text

We'll often want to search for a pattern rather than a specific text string in the real world. We'll be introducing three more SQL keywords into our vocabulary to help us achieve this: LIKE, NOT LIKE, and IN.

## 4. LIKE

In SQL, we can use the LIKE operator with a WHERE clause to search for a pattern in a field. We use a wildcard as a placeholder for some other values to accomplish this. There are two wildcards with LIKE, the percent, and the underscore. The percent wildcard will match zero, one, or many characters in the text. For example, the query on the left matches people like Adel, Adelaide, and Aden. The underscore wildcard will match a single character. For example, the query on the right matches only three-letter names like Eve. We'd also see names like Eva if it were in our dataset. Eva Mendes, however, would not be visible unless the search criteria looked like this.

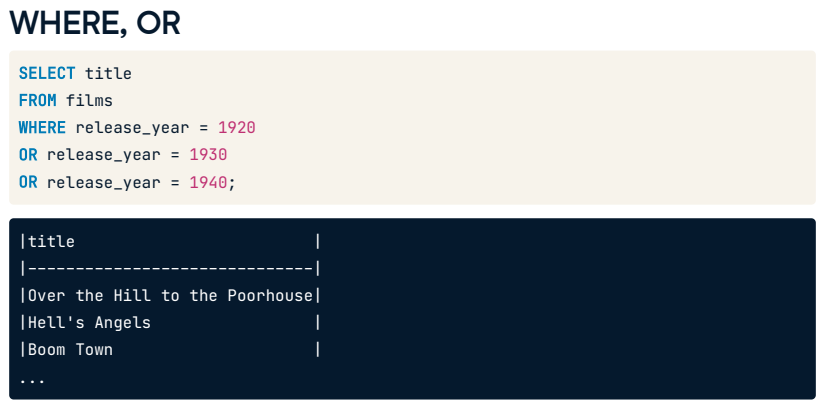
## 5. NOT LIKE

We can also use the NOT LIKE operator to find records that don't match the specified pattern. In this query, we are finding records for people who do not have A-dot as part of their first name. It's important to note that this operation is case-sensitive, so we must be mindful of what we are querying.

## 6. Wildcard position

We've reviewed one example of where to position each wildcard, but we can actually put them anywhere and combine them! We can find values that start, end, or contain characters in any position, as well as find records of a certain length. For example, this code on the left will find all people whose name ends in r. The code on the right will find records where the third character is t.

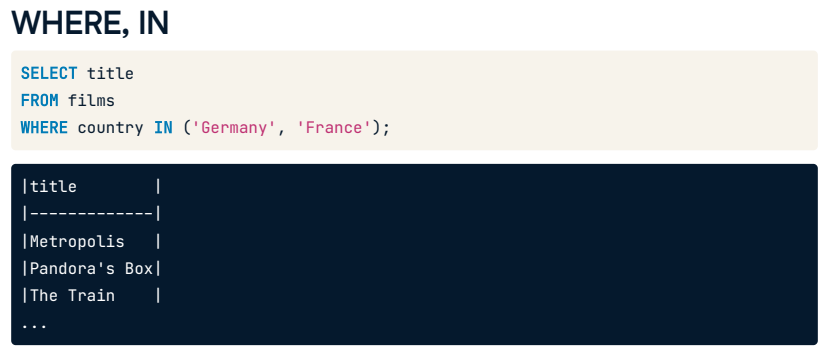
## 7. WHERE, OR

What if we want to filter based on many conditions or a range of numbers? We could chain several ORs to the WHERE clause based on what we know, but that can get messy. We can see an example here where we select the film titles released in 1920, 1930, or 1940.

## 8. WHERE, IN

A helpful operator here is IN. The IN operator allows us to specify multiple values in a WHERE clause, making it easier and quicker to set numerous OR conditions. Neat, right? So, the example shown on the previous slide would simply become WHERE release\_year IN 1920, 1930, 1940, where the years are enclosed in parentheses.

## 9. WHERE, IN

Here is another example using a text field where we want to find the title WHERE the associated country is either Germany or France.

## 10. Let's practice!

Let's practice!

## Exercise

# LIKE and NOT LIKE

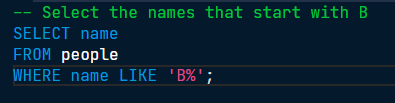
The LIKE and NOT LIKE operators can be used to find records that either match or do not match a specified pattern, respectively. They can be coupled with the wildcards % and \_. The % will match zero or many characters, and \_ will match a single character.

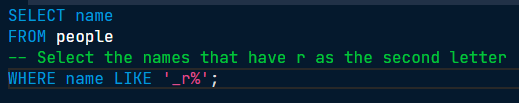
This is useful when you want to filter text, but not to an exact word.

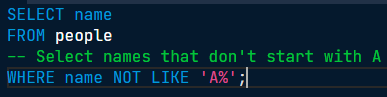
Do the following exercises to gain some practice with these keywords.

## Instructions

* + Select the names of all people whose names begin with 'B
  + Select the names of people whose names have 'r' as the second letter.
  + Select the names of people whose names don't start with 'A'.







I LIKE to see the progress we're making! Filtering your data to find specified patterns is vital to your skillset. Our results still had names that started with Á with an accent, showing that we need to be specific with our filtering criteria.

## Exercise

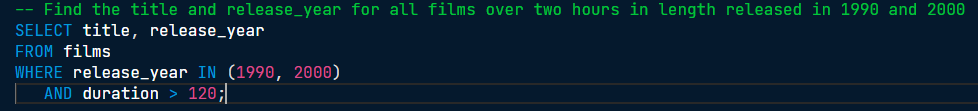
# WHERE IN

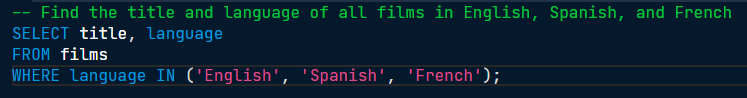
You now know you can query multiple conditions using the IN operator and a set of parentheses. It is a valuable piece of code that helps us keep our queries clean and concise.

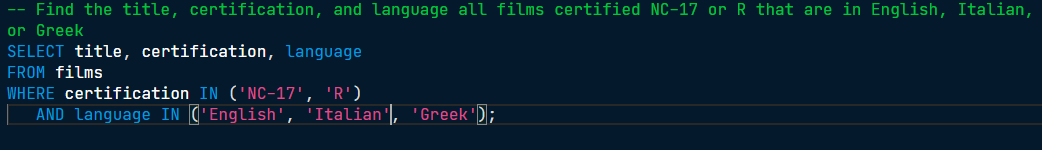
Try using the IN operator yourself!

## Instructions

* + Select the title and release\_year of all films released in 1990 or 2000 that were longer than two hours.
  + Select the title and language of all films in English, Spanish, or French using IN.
  + Select the title, certification and language of all films certified NC-17 or R that are in English, Italian, or Greek.







our SQL vocabulary is growing by the minute! Interestingly, A Fistful of Dollars starring Clint Eastwood is listed as Italian.

## Exercise

# Combining filtering and selecting

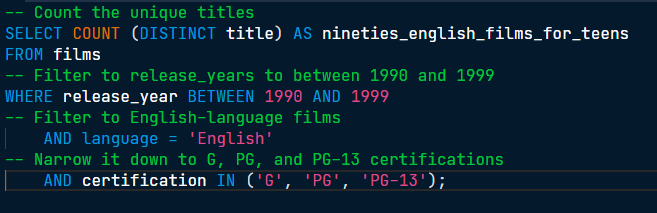
Time for a little challenge. So far, your SQL vocabulary from this course includes COUNT(), DISTINCT, LIMIT, WHERE, OR, AND, BETWEEN, LIKE, NOT LIKE, and IN. In this exercise, you will try to use some of these together. Writing more complex queries will be standard for you as you become a qualified SQL programmer.

As this query will be a little more complicated than what you've seen so far, we've included a bit of code to get you started. You will be using DISTINCT here too because, surprise, there are two movies named 'Hamlet' in this dataset!

Follow the instructions to find out what 90's films we have in our dataset that would be suitable for English-speaking teens.

## Instructions

* Count the unique titles from the films database and use the alias provided.
* Filter to include only movies with a release\_year from 1990 to 1999, inclusive.
* Add another filter narrowing your query down to English-language films.
* Add a final filter to select only films with 'G', 'PG', 'PG-13' certifications.



You've got a natural flair for filtering! Nice work, this filter tells us we have 310 films that the 90's obsessed teenagers can enjoy.