I see what you mean now! Here's the text with the SQL code directly embedded within the explanation:

In this video, we’re diving into one of SQL’s most powerful features: **filtering**. Filtering helps you select specific data based on a condition, making it a crucial tool for refining queries and extracting just what you need from a database.

**What is Filtering?**

Filtering is the process of selecting only the data that meets certain criteria. Think of it like choosing apples from a fruit cart, where you specify that you only want **fresh** apples. By applying a filter, you exclude the ones that aren’t fresh.

As a **security analyst**, filtering is helpful when working with large datasets. For example, if you're analyzing a **log-in attempts table**, you might filter to only show attempts from a specific country, like Canada. You could write a query like this:

SELECT \* FROM log\_in\_attempts WHERE country = 'Canada';

**SQL Filtering with Operators**

In SQL, filters are applied using operators, which are symbols or keywords that define the condition. One common operator is the **equals (=)** operator. For instance, if you wanted to find records where the **country** column equals **'USA'**, you would use:

SELECT \* FROM log\_in\_attempts WHERE country = 'USA';

The **WHERE** clause specifies the condition, and in this case, it filters the data to show only records where the country is **USA**.

**Pattern Matching with LIKE**

While filtering with an exact match is useful, sometimes you might need to filter based on a pattern. For example, if you're searching for offices in the East building, you could use the **LIKE** operator with a wildcard (%) to match any text starting with "East". The query might look like:

SELECT \* FROM employees WHERE office LIKE 'East%';

This query will return all records where the office column starts with "East" (e.g., **East-120**, **East-290**, **East-435**).

**Handling Inconsistencies**

What if there are inconsistencies in how data is stored? For instance, your database might use both **US** and **USA** to represent the United States. In this case, you can use LIKE to account for these variations. To filter for both **US** and **USA**, the query would look like:

SELECT \* FROM log\_in\_attempts WHERE country LIKE 'US%';

This query would return entries where the **country** starts with "US", catching both "US" and "USA".

By using these filtering techniques, you can make your SQL queries much more precise and tailor them to retrieve exactly the data you need. You've now learned how to refine queries using both exact matches and patterns in SQL. Exciting stuff ahead as we continue to explore more advanced filtering options!

This should now be easier to read with the code properly integrated within the text. Let me know if you'd like any further adjustments!

# The WHERE clause and basic operators

Previously, you focused on how to refine your SQL queries by using the WHERE clause to filter results. In this reading, you’ll further explore how to use the WHERE clause, the LIKE operator and the percentage sign (%) wildcard. You’ll also be introduced to the underscore (\_), another wildcard that can help you filter queries.

## How filtering helps

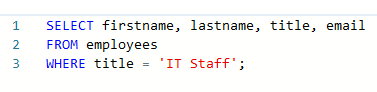
As a security analyst, you'll often be responsible for working with very large and complicated security logs. To find the information you need, you'll often need to use SQL to filter the logs.

In a cybersecurity context, you might use filters to find the login attempts of a specific user or all login attempts made at the time of a security issue. As another example, you might filter to find the devices that are running a specific version of an application.

## WHERE

To create a filter in SQL, you need to use the keyword WHERE. WHERE indicates the condition for a filter.

If you needed to email employees with a title of IT Staff, you might use a query like the one in the following example. You can run this example to examine what it returns:



Rather than returning all records in the employees table, this WHERE clause instructs SQL to return only those that contain 'IT Staff' in the title column. It uses the equals sign (=) operator to set this condition.

**Note:** You should place the semicolon (;) where the query ends. When you add a filter to a basic query, the semicolon is after the filter.

## Filtering for patterns

You can also filter based on a pattern. For example, you can identify entries that start or end with a certain character or characters. Filtering for a pattern requires incorporating two more elements into your WHERE clause:

* a wildcard
* the LIKE operator

### ****Wildcards****

A **wildcard** is a special character that can be substituted with any other character. Two of the most useful wildcards are the percentage sign (%) and the underscore (\_):

* The percentage sign substitutes for any number of other characters.
* The underscore symbol only substitutes for one other character.

These wildcards can be placed after a string, before a string, or in both locations depending on the pattern you’re filtering for.

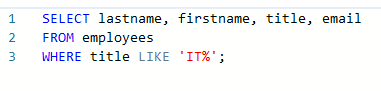
The following table includes these wildcards applied to the string 'a' and examples of what each pattern would return.

|  |  |
| --- | --- |

### ****LIKE****

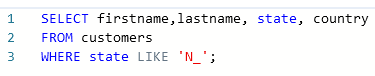
To apply wildcards to the filter, you need to use the LIKE operator instead of an equals sign (=). LIKE is used with WHERE to search for a pattern in a column.

For instance, if you want to email employees with a title of either 'IT Staff' or 'IT Manager', you can use LIKE operator combined with the % wildcard:



This query returns all records with values in the title column that start with the pattern of 'IT'. This means both 'IT Staff' and 'IT Manager' are returned.

As another example, if you want to search through the invoices table to find all customers located in states with an abbreviation of 'NY', 'NV', 'NS' or 'NT', you can use the 'N\_' pattern on the state column:



This returns all the records with state abbreviations that follow this pattern.

## Key takeaways

Filters are important when refining what your query returns. WHERE is an essential keyword for adding a filter to your query.  You can also filter for patterns by combining the LIKE operator with the percentage sign (%) and the underscore (\_) wildcards.