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:

```
SELECT firstname, lastname, title, email
FROM employees
WHERE title = 'IT Staff';
```

Reset

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.

Pattern	Results that could be returned
'a%'	**apple123, art, a**
'a_'	**as, an, a7**
'a__'	**ant, add, a1c**
'%a'	**pizza, Z6ra, a**
'_a'	**ma, 1a, Ha**
'%a%'	**Again, back, a**
'_a_'	**Car, ban, ea7**

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:

```
SELECT lastname, firstname, title, email
FROM employees
WHERE title LIKE 'IT%';
```

Reset

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:

```
SELECT firstname,lastname, state, country
FROM customers
WHERE state LIKE 'N_';
```

Reset

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.