SQL CLAUSES

SQL (Structured Query Language) is a programming language used to manage and manipulate data stored in relational databases. SQL queries typically involve one or more clauses that are used to specify the conditions and actions of the query.

Here are some common SQL clauses:

SELECT

The SELECT clause is used to specify the columns that you want to retrieve from the database. It is typically the first clause used in a SQL query.

Example:

SELECT first_name, last_name, email

FROM customers;

This query selects the first name, last name, and email columns from the customers table.

FROM

The FROM clause specifies the table or tables from which to retrieve the data.

Example:

SELECT * FROM orders;

This query selects all columns from the orders table.

WHERE

The WHERE clause is used to filter the data based on specified conditions.

Example:

SELECT *

FROM orders

WHERE customer_id = 123;

This query selects all columns from the orders table where the customer_id is equal to 123.

GROUP BY

The GROUP BY clause is used to group rows that have the same values in a specified column or columns.

Example:

SELECT product_id, COUNT(*)

FROM order_items

GROUP BY product_id;

This query groups the order_items table by product_id and returns the number of times each product appears in the table.

HAVING

The HAVING clause is used to filter the results of a GROUP BY query.

Example:

```
SELECT product_id, COUNT(*)

FROM order_items

GROUP BY product_id

HAVING COUNT(*) > 10;
```

This query groups the order_items table by product_id and returns only the results where the count is greater than 10.

ORDER BY

The ORDER BY clause is used to sort the results of a query by one or more columns.

Example:

```
SELECT first_name, last_name, email

FROM customers

ORDER BY last_name, first_name;
```

This query selects the first name, last name, and email columns from the customers table and sorts the results by last name and then by first name.

LIMIT

The LIMIT clause is used to limit the number of rows returned by a query.

Example:

```
SELECT *

FROM orders

LIMIT 10;
```

This query selects all columns from the orders table but returns only the first 10 rows.

INSERT INTO

The INSERT INTO clause is used to add new rows to a table.

Example:

```
INSERT INTO customers (first_name, last_name, email)

VALUES ('John', 'Doe', 'johndoe@email.com');
```

This query adds a new row to the customers table with the specified values.

UPDATE

The UPDATE clause is used to modify existing rows in a table.

Example:

```
UPDATE customers

SET first_name = 'Jane', last_name = 'Doe'

WHERE id = 123;
```

This query updates the first_name and last_name columns of the row with id 123 in the customers table.

DELETE

The DELETE clause is used to remove rows from a table.

Example:

```
DELETE FROM customers

WHERE id = 123;
```

This query removes the row with id 123 from the customers table.

JOIN

The JOIN clause is used to combine rows from two or more tables based on a related column.

Example:

```
SELECT customers.first_name, orders.order_date

FROM customers

JOIN orders ON customers.id = orders.customer_id;
```

This query selects the first name from the customers table and the order date from the orders table, and joins the tables on the customer_id and id columns, respectively.

LEFT JOIN/RIGHT JOIN

The LEFT JOIN and RIGHT JOIN clauses are used to combine rows from two or more tables, but retain all rows from one table even if there is no match in the other table.

Example:

```
SELECT customers.first_name, orders.order_date

FROM customers

LEFT JOIN orders ON customers.id = orders.customer_id;
```

This query selects the first name from the customers table and the order date from the orders table, and joins the tables on the customer_id and id columns, respectively. However, all rows from the customers table will be returned even if there is no match in the orders table.

UNION

The UNION clause is used to combine the results of two or more SELECT statements into a single result set.

Example:

```
SELECT first_name, last_name, email

FROM customers

UNION

SELECT first_name, last_name, email

FROM leads;
```

This query selects the first name, last name, and email columns from the customers and leads tables, and combines the results into a single result set.

SQL FUCNTIONS

SQL functions are pre-defined formulas or calculations that can be used to perform various tasks on data stored in a relational database. These functions are used to transform, manipulate, and analyze data in a variety of ways. In this article, we'll discuss some of the most commonly used SQL functions and provide examples of how they can be used.

COUNT()

The COUNT() function is used to count the number of rows that match a specified condition in a table.

Example:

```
SELECT COUNT(*)

FROM customers;
```

This query counts the total number of rows in the customers table.

SUM()

The SUM() function is used to calculate the sum of a column in a table.

Example:

```
SELECT SUM(price)

FROM orders;
```

This query calculates the sum of the price column in the orders table.

AVG()

The AVG() function is used to calculate the average of a column in a table.

Example:

```
SELECT AVG(price)

FROM orders;
```

This query calculates the average of the price column in the orders table.

MIN()

The MIN() function is used to find the minimum value in a column in a table.

Example:

```
SELECT MIN(price)

FROM orders;
```

This query finds the minimum value in the price column in the orders table.

MAX()

The MAX() function is used to find the maximum value in a column in a table.

Example:

```
SELECT MAX(price)

FROM orders;
```

This query finds the maximum value in the price column in the orders table.

CONCAT()

The CONCAT() function is used to concatenate two or more strings into a single string.

Example:

```
SELECT CONCAT(first_name, ' ', last_name) as full_name

FROM customers;
```

This query combines the first_name and last_name columns into a single column called full_name.

SUBSTRING()

The SUBSTRING() function is used to extract a substring from a string.

Example:

SELECT SUBSTRING(email, 1, 5) as email_prefix

FROM customers;

This query extracts the first five characters from the email column in the customers table.

DATEPART()

The DATEPART() function is used to extract a specific part of a date.

Example:

SELECT DATEPART(year, order_date) as order_year

FROM orders;

This query extracts the year from the order_date column in the orders table.

DATEADD()

The DATEADD() function is used to add a specific amount of time to a date.

Example:

SELECT DATEADD(day, 7, order_date) as new_order_date

FROM orders;

This query adds seven days to the order_date column in the orders table.

ROUND()

The ROUND() function is used to round a number to a specified number of decimal places.

Example:

SELECT ROUND(price, 2) as rounded_price

FROM orders;

This query rounds the price column in the orders table to two decimal places.

In conclusion, SQL functions are an essential tool for working with data in a relational database. By using these functions, you can easily transform, manipulate, and analyze data to extract valuable insights and make informed decisions.