SQL Server SELECT

In SQL Server, the SELECT statement is used to query data from a database. It is one of the most important commands in SQL, allowing you to retrieve rows from one or more tables. Below are some commonly used variations and their different purposes:

1. Basic SELECT Statement
- Use: Retrieve specific columns or all columns from a table.

Syntax :
```
  SELECT column1, column2 FROM table_name;
```
Or to get all columns:
Syntax :
```
  SELECT * FROM table_name;
```

2. SELECT DISTINCT
- Use: Return only unique values, removing duplicates.

Syntax :
```
  SELECT DISTINCT column1 FROM table_name;
```

3. SELECT with WHERE Clause
- Use: Retrieve rows that meet specific conditions.

Syntax :
```
  SELECT column1, column2 FROM table_name WHERE condition;
```
Example:
Syntax :
```
  SELECT first_name, age FROM employees WHERE age > 30;
```

4. SELECT with ORDER BY
- Use: Sort the result set based on one or more columns, either in ascending (ASC) or descending (DESC) order.

Syntax :
```
  SELECT column1, column2 FROM table_name ORDER BY column1 ASC;
```

5. SELECT with GROUP BY
- Use: Aggregate data based on one or more columns and group the results. Often used with aggregate functions like COUNT(), SUM(), AVG(), etc.

Syntax :
```
  SELECT column1, COUNT(*) FROM table_name GROUP BY column1;
```

6. SELECT with HAVING
- Use: Filter aggregated results (works in conjunction with GROUP BY).

Syntax :
```
  SELECT column1, COUNT(*) FROM table_name GROUP BY column1 HAVING
COUNT(*) > 1;
```

7. SELECT with JOIN
- Use: Combine rows from two or more tables based on a related column.

INNER JOIN:
  Syntax :
```
    SELECT a.column1, b.column2
```

```
  FROM table1 a
  INNER JOIN table2 b ON a.id = b.id;
```

LEFT JOIN (or LEFT OUTER JOIN):

Syntax :

```
  SELECT a.column1, b.column2
  FROM table1 a
  LEFT JOIN table2 b ON a.id = b.id;
```

RIGHT JOIN (or RIGHT OUTER JOIN):

Syntax :

```
  SELECT a.column1, b.column2
  FROM table1 a
  RIGHT JOIN table2 b ON a.id = b.id;
```

8. SELECT with Subquery

- Use: Use a query inside another query, often used for filtering or calculations.

Syntax :

```
 SELECT column1
 FROM table_name
 WHERE column2 = (SELECT MAX(column2) FROM another_table);
```

9. SELECT with TOP

- Use: Retrieve only a certain number of rows from the result set, usually used for paging or limiting results.

Syntax :

```
 SELECT TOP 10 column1 FROM table_name;
```

10. SELECT INTO

- Use: Create a new table from the result of a query.

Syntax :

```
 SELECT column1, column2 INTO new_table FROM old_table;
```

11. SELECT with UNION

- Use: Combine results from multiple SELECT queries into a single result set. Removes duplicates unless UNION ALL is used.

Syntax :

```
 SELECT column1 FROM table1
 UNION
 SELECT column1 FROM table2;
```

12. SELECT with CASE Expression

- Use: Return different values depending on conditions.

Syntax :

```
 SELECT column1,
        CASE
          WHEN condition1 THEN 'Result1'
          WHEN condition2 THEN 'Result2'
          ELSE 'Other'
        END AS alias_name
 FROM table_name;
```

13. SELECT with Aggregate Functions
- Use: Perform calculations on a set of values and return a single value.

COUNT:

Syntax :

```
SELECT COUNT(*) FROM table_name;
```

SUM:

Syntax :

```
SELECT SUM(column1) FROM table_name;
```

AVG:

Syntax :

```
SELECT AVG(column1) FROM table_name;
```

MAX/MIN:

Syntax :

```
SELECT MAX(column1) FROM table_name;
```

14. SELECT with OFFSET-FETCH (for Pagination)
- Use: Retrieve a specific range of rows, commonly used for paginating results.

Syntax :

```
SELECT column1 FROM table_name
ORDER BY column1
OFFSET 10 ROWS FETCH NEXT 10 ROWS ONLY;
```