3) An aggregate function is a function that performs a calculation on a set of values, and returns a single value.

Aggregate functions are often used with the GROUP BY clause of the SELECT statement. The GROUP BYclause splits the result-set into groups of values and the aggregate function can be used to return a single value for each group.

The most commonly used SQL aggregate functions are:

* MIN() - returns the smallest value within the selected column.

Syntax: SELECT MIN(column\_name)  
FROM table\_name  
WHERE condition;

* MAX() - returns the largest value within the selected column.

Syntax: SELECT MAX(column\_name) FROM table\_name  
WHERE condition;

* COUNT() - returns the number of rows that matches a specified criteria.

Syntax: SELECT COUNT(*column\_name*) FROM *table\_name* WHERE *condition*;

* SUM() - returns the total sum of a numerical column

Syntax: SELECT SUM(*column\_name*) FROM *table\_name*  
WHERE *condition*;

* AVG() - returns the average value of a numerical column

Syntax: SELECT AVG(*column\_name*) FROM *table\_name*  
WHERE *condition*;

USAGE:

* When you use MIN() or MAX(), the returned column will not have a descriptive name. To give the column a descriptive name, use the AS keyword.

Example: SELECT MIN(Price) AS SmallestPrice FROM Products;

* We can use the MIN() or MAX(), function and the GROUP BY clause, to group the result-set by one or more columns.

Example: SELECT MAX(Price) AS SmallestPrice, CategoryID FROM Products GROUP BY CategoryID;

* We can ignore duplicates by using the DISTINCT keyword in the COUNT() function. If DISTINCT is specified, rows with the same value for the specified column will be counted as one.

Example: SELECT COUNT(DISTINCT Price)  
FROM Products;

* The parameter inside the SUM() function can also be an expression.

Example: SELECT SUM(Quantity \* 10) FROM OrderDetails;

* To list all records with a higher price than average, we can use the AVG() function in a sub query.
* SELECT \* FROM Products WHERE price > (SELECT AVG(price) FROM Products);

4) Pivot tables in SQL are a powerful tool used for data summarization, analysis, and reporting. They allow you to transform rows into columns and aggregate data to make it easier to understand and analyze.

That is a pivot table in SQL reorganizes data from multiple rows into columns, enabling users to view data from different perspectives.

To create pivot tables in SQL, it's essential to comprehend the basic syntax and structure of the PIVOT function. A pivot table query's fundamental structure comprises:

* SELECT Statement: This defines the columns to be included in the final output.
* FROM Clause: Specifies the source table where the data is coming from.
* PIVOT Clause: This is where the magic happens. The PIVOT clause is used to define the pivot table operation. It includes:
  + The aggregation function (such as SUM, AVG, COUNT) is to be applied.
  + The column that will be turned into a set of output columns (the pivot column).
  + The column(s) that will be used for grouping the data (the value column).
* WHERE Clause (Optional): This filters the data based on the given criteria (or condition).

5) SQL JOIN clause is used to combine rows from two or more tables, based on a related column between them.

The different types of the JOINs in SQL:

* (INNER) JOIN: Returns records that have matching values in both tables
* LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table
* RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table
* FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table