1. COUNT Function

Purpose: Counts the number of rows in a database table or the number of non-NULL values in a column.

Example:

```sql
SELECT COUNT(*) AS TotalRows FROM Students; -- Counts all rows in the Students table.
SELECT COUNT(StudentID) AS TotalStudents FROM Students WHERE Age > 18; -- Counts rows where Age is greater than 18.
```

---

2. MAX Function

Purpose: Finds the maximum value in a column.

Example:

```sql
SELECT MAX(Marks) AS HighestMarks FROM Students; -- Retrieves the highest marks in the Students table.
```

---

3. MIN Function

Purpose: Finds the minimum value in a column.

Example:

```sql
SELECT MIN(Marks) AS LowestMarks FROM Students; -- Retrieves the lowest marks in the Students table.
```

---

4. AVG Function

Purpose: Calculates the average value of a numeric column.

Example:

```sql
SELECT AVG(Salary) AS AverageSalary FROM Employees; -- Calculates the average salary of all employees.
```

---

5. SUM Function

Purpose: Calculates the total sum of a numeric column.

Example:

```sql
SELECT SUM(Salary) AS TotalSalaries FROM Employees WHERE Department = 'HR'; -- Sums up salaries of employees in the HR department.
```

---

6. SQRT Function

Purpose: Calculates the square root of a numeric value.

Example:

```sql
SELECT SQRT(16) AS SquareRoot; -- Returns 4, which is the square root of 16.
```

---

7. RAND Function

Purpose: Generates a random number.

Example:

```sql
SELECT RAND() AS RandomNumber; -- Generates a random decimal number between 0 and 1.
```

```sql
SELECT RAND(5) AS RandomSeed; -- Generates a random number based on seed 5.
```

---

8. CONCAT Function

Purpose: Concatenates multiple string values into one.

Example:

sql Copy code

```sql
SELECT CONCAT(FirstName, ' ', LastName) AS FullName FROM Employees; -- Combines the first and last names of employees.
```

---

9. RANK Function

Purpose: Assigns a rank to each row within a result set, with gaps in ranks if there are ties.

Example:

sql Copy code

```sql
SELECT RANK() OVER (ORDER BY Marks DESC) AS Rank, StudentName FROM Students; -- Assigns ranks based on marks in descending order, with gaps for ties.
```

---

10. DENSE_RANK Function

Purpose: Similar to `RANK`, but without gaps in ranks for ties.

Example:

sql Copy code

```sql
SELECT DENSE_RANK() OVER (ORDER BY Marks DESC) AS Rank, StudentName FROM Students; -- Assigns ranks without gaps in case of ties.
```

---

11. ROW_NUMBER Function

Purpose: Assigns a unique row number to each row within a result set.

Example:

```sql
SELECT ROW_NUMBER() OVER (ORDER BY Marks DESC) AS RowNo, StudentName FROM Students; -- Assigns a unique row number based on descending marks.
```

Practical Scenario Combining Functions

Imagine you have a **Students** table with columns: `StudentID`, `StudentName`, `Marks`, `Age`.

Example Query:

```sql
SELECT StudentID, StudentName, Marks, RANK() OVER (ORDER BY Marks DESC) AS Position, DENSE_RANK() OVER (ORDER BY Marks DESC) AS DensePosition, ROW_NUMBER() OVER (ORDER BY Marks DESC) AS RowNumber FROM Students;
```

Result:

StudentID	StudentName	Marks	Position	DensePosition	RowNumber
1	Alice	95	1	1	1
2	Bob	90	2	2	2
3	Charlie	90	2	2	3

This highlights the differences between `RANK`, `DENSE_RANK`, and `ROW_NUMBER`.