Here’s a **complete breakdown** with **definitions + examples** — all using our familiar **CollegeDB (Student–Department)** setup.

**🌟 1️⃣ DISTINCT**

Removes duplicate values from results — used for **unique filtering**.

-- Show unique department IDs from students

SELECT DISTINCT DeptID FROM Student;

-- Show unique combinations of DeptID and Gender

SELECT DISTINCT DeptID, Gender FROM Student;

✅ **Purpose:** Filter out duplicates from query results.

**🌟 2️⃣ TOP / LIMIT / FETCH (Row Filtering)**

Used to **limit number of rows** returned.

-- MySQL syntax

SELECT \* FROM Student

ORDER BY Marks DESC

LIMIT 3;

-- SQL Server syntax

SELECT TOP 3 \* FROM Student ORDER BY Marks DESC;

-- Oracle / PostgreSQL

SELECT \* FROM Student ORDER BY Marks DESC FETCH FIRST 3 ROWS ONLY;

✅ **Purpose:** Retrieve top N records — e.g., top 3 students by marks.

**🌟 3️⃣ CASE (Conditional Filtering Logic)**

Used for **conditional filtering or computed filtering** in SELECT.

-- Assign grade categories based on marks

SELECT Name, Marks,

CASE

WHEN Marks >= 85 THEN 'Distinction'

WHEN Marks BETWEEN 70 AND 84 THEN 'First Class'

ELSE 'Average'

END AS Grade

FROM Student;

✅ **Purpose:** Create filters or computed columns based on conditions.

**🌟 4️⃣ Subqueries (Nested Filtering)**

A **query inside another query** to filter based on another table or computed result.

-- Students having marks greater than the average marks

SELECT \* FROM Student

WHERE Marks > (SELECT AVG(Marks) FROM Student);

-- Students who belong to departments located in 'Block A'

SELECT \* FROM Student

WHERE DeptID IN (SELECT DeptID FROM Department WHERE Location='Block A');

✅ **Purpose:** Filter using **results of another query**.

**🌟 5️⃣ HAVING (Filter after Aggregation)**

Used to filter **groups** (after GROUP BY), not individual rows.

-- Show departments where average marks > 75

SELECT DeptID, AVG(Marks) AS AvgMarks

FROM Student

GROUP BY DeptID

HAVING AVG(Marks) > 75;

✅ **Purpose:** Filters **aggregated results** (unlike WHERE which filters rows before grouping).

**🌟 6️⃣ EXISTS / NOT EXISTS**

Used for **correlated subqueries** (row-by-row filtering).

-- Departments having at least one student

SELECT DeptName FROM Department d

WHERE EXISTS (SELECT 1 FROM Student s WHERE s.DeptID = d.DeptID);

-- Departments having no students

SELECT DeptName FROM Department d

WHERE NOT EXISTS (SELECT 1 FROM Student s WHERE s.DeptID = d.DeptID);

✅ **Purpose:** Filters data based on **existence or absence** of related records.

**🌟 7️⃣ ALL / ANY**

Used with subqueries for **comparative filtering**.

-- Students with marks greater than all students in Mechanical

SELECT \* FROM Student

WHERE Marks > ALL (SELECT Marks FROM Student WHERE DeptID=2);

-- Students with marks greater than any student in Electronics

SELECT \* FROM Student

WHERE Marks > ANY (SELECT Marks FROM Student WHERE DeptID=3);

✅ **Purpose:** Compare one value with a **set of values**.

**🌟 8️⃣ Window Filtering (Using RANK, DENSE\_RANK, ROW\_NUMBER)**

Advanced way to filter using ranking functions.

-- Top student per department

WITH Ranked AS (

SELECT Name, DeptID, Marks,

RANK() OVER (PARTITION BY DeptID ORDER BY Marks DESC) AS rnk

FROM Student

)

SELECT \* FROM Ranked WHERE rnk = 1;

✅ **Purpose:** Filter based on **ranking inside groups** (without removing rows before ranking).

**🌟 9️⃣ CONDITIONAL AGGREGATE FILTERING**

Use conditional logic inside aggregates.

-- Count male vs female students per department

SELECT DeptID,

COUNT(CASE WHEN Gender='M' THEN 1 END) AS MaleCount,

COUNT(CASE WHEN Gender='F' THEN 1 END) AS FemaleCount

FROM Student

GROUP BY DeptID;

✅ **Purpose:** Apply **conditional filters** inside COUNT() or SUM().

**🌟 🔟 Combining Multiple Filters**

You can mix advanced and basic filters for complex queries.

-- Female students in 'Block A' departments

-- whose marks are above average of their own department

SELECT s.Name, s.Marks, d.DeptName

FROM Student s

JOIN Department d ON s.DeptID = d.DeptID

WHERE s.Gender = 'F'

AND d.Location = 'Block A'

AND s.Marks > (

SELECT AVG(Marks)

FROM Student

WHERE DeptID = s.DeptID

);

✅ **Purpose:** Complex real-world data filtering across multiple tables.

**💡 Summary Table**

| **Category** | **Command / Clause** | **Used For** |
| --- | --- | --- |
| **Basic** | WHERE, AND, OR, NOT | Row-level filtering |
| **Set Comparison** | IN, ANY, ALL, EXISTS | Compare with multiple values/subquery |
| **Pattern / Range** | LIKE, BETWEEN | Match patterns or numeric ranges |
| **Aggregate Filtering** | HAVING | Filter group results |
| **Distinct Filtering** | DISTINCT | Remove duplicates |
| **Top-N Filtering** | LIMIT / TOP / FETCH | Restrict number of rows |
| **Conditional Logic** | CASE | Filter or compute conditionally |
| **Window Filtering** | RANK, ROW\_NUMBER | Filter based on rankings |
| **Subquery Filtering** | Subqueries in WHERE or HAVING | Dynamic comparisons |