Ch-13: Grouping Records

Operators in SQL

Mathematical: +, -, *, /
Relational: =, !=, >, <, >=, <=
Logical: AND, OR, NOT

1. Mathematical Operators

Used to perform **arithmetic operations** on numeric columns

Operator	Description	Example Query
+	Addition	SELECT Marks + 5 FROM Student;
-	Subtraction	SELECT Marks - 2 FROM Student;
*	Multiplication	SELECT Marks * 2 FROM Student;
/	Division	SELECT Marks / 2 FROM Student;

Example:
Increase all students' marks by 5

```
SELECT Name, Marks, Marks + 5 AS Updated_Marks
FROM Student;
```

2. Relational (Comparison) Operators

Used in WHERE clause to compare values.

Operator	Description	Example Query
=	Equal to	SELECT * FROM Student WHERE Marks = 90;
!=	Not equal to	SELECT * FROM Student WHERE Marks != 50;
>	Greater than	SELECT * FROM Student WHERE Marks > 70;
<	Less than	SELECT * FROM Student WHERE Marks < 60;
>=	Greater than or equal	SELECT * FROM Student WHERE Marks >= 85;
<=	Less than or equal	SELECT * FROM Student WHERE Marks <= 40;

Example:
Show students who scored more than 80

```
SELECT Name, Marks
FROM Student
WHERE Marks > 80;
```

3. Logical Operators

Used to combine multiple conditions in a WHERE clause.

Operator	Description	Example Query
AND	True if **both** conditions are true	SELECT * FROM Student WHERE Class = 12 AND Marks > 80;
OR	True if **any one** condition is true	SELECT * FROM Student WHERE Class = 11 OR Class = 12;
NOT	Reverses the condition	SELECT * FROM Student WHERE NOT Marks > 75;

Example:
Students of class 12 who scored more than 85

```
SELECT Name, Class, Marks
FROM Student
WHERE Class = 12 AND Marks > 85;
```

Students who are NOT in class 10

```
SELECT Name, Class
FROM Student
WHERE NOT Class = 10;
```

Other Useful Clauses and Concepts

Clause / Concept	Example	Description
Aliasing	SELECT Name AS StudentName FROM Student;	Gives a temporary name to a column
DISTINCT	SELECT DISTINCT Class FROM Student;	Removes duplicate values
WHERE	SELECT * FROM Student WHERE Marks > 80;	Filters rows based on condition
IN	WHERE Name IN ('Ravi', 'Neha')	Matches any value in a list
BETWEEN	WHERE Marks BETWEEN 50 AND 90	Range filter
ORDER BY	ORDER BY Name ASC	Sorts the result set
NULL	Represents missing or undefined value	
IS NULL	WHERE Marks IS NULL	Checks for null values
IS NOT NULL	WHERE Marks IS NOT NULL	Checks for non-null values
LIKE	WHERE Name LIKE 'A%'	Pattern matching using % and _

Aggregate Functions

Function	Description
MAX()	Returns highest value
MIN()	Returns lowest value
SUM()	Adds all values
AVG()	Calculates average
COUNT()	Counts the number of rows

Example:

```
SELECT MAX(Marks), AVG(Marks) FROM Student;
```

GROUP BY and HAVING

Group records by class and show average marks

```
SELECT Class, AVG(Marks)
FROM Student
GROUP BY Class
HAVING AVG(Marks) > 70;
```