Operators:

1. Arithmetic Operator: +,-,*,/

Ex. To get the annual salary of employees

```
SELECT ename,SAL*12 AS AnnualSalary from emp;
```

```
+--------+--------------+
| ename  | AnnualSalary |
+--------+--------------+
| SMITH  |     9600.00 |
| ALLEN  |    19200.00 |
| WARD   |    15000.00 |
| JONES  |    35700.00 |
| MARTIN |    15000.00 |
| BLAKE  |    34200.00 |
| CLARK  |    29400.00 |
| SCOTT  |    36000.00 |
| KING   |    60000.00 |
| TURNER |    18000.00 |
| ADAMS  |    13200.00 |
| JAMES  |    11400.00 |
| FORD   |    36000.00 |
| MILLER |    15600.00 |
+--------+--------------+
```

2. Relational Operators: =,!=,>,<

Ex.To get a list of employee whose salary is greater than 2000.

SELECT * from emp where sal > 2000;

```
+-------+-------+-----------+------+------------+---------+------+--------+
| EMPNO | ENAME | JOB       | MGR  | HIREDATE   | SAL     | COMM | DEPTNO |
+-------+-------+-----------+------+------------+---------+------+--------+
|  7566 | JONES | MANAGER   | 7839 | 1981-04-02 | 2975.00 | NULL |     20 |
|  7698 | BLAKE | MANAGER   | 7839 | 1981-05-01 | 2850.00 | NULL |     30 |
|  7782 | CLARK | MANAGER   | 7839 | 1981-06-09 | 2450.00 | NULL |     10 |
|  7788 | SCOTT | ANALYST   | 7566 | 1982-12-09 | 3000.00 | NULL |     20 |
|  7839 | KING  | PRESIDENT | NULL | 1981-11-17 | 5000.00 | NULL |     10 |
|  7902 | FORD  | ANALYST   | 7566 | 1981-12-03 | 3000.00 | NULL |     20 |
+-------+-------+-----------+------+------------+---------+------+--------+
```

3.Logical Operators: AND, OR, NOT

Ex. SELECT * from emp where age > 25 AND SAL > 2000;

Where Clause:

Filters records based on the condition

Examples:

1. Retrieve employees with job title as ‘Manager’.
   - ➔ SELECT ename,job
   - ➔ From emp;
   - ➔ Where job=’ Manager’ ;

2. Select employee names,job titles and salary for those who are working in department no 30.

```
mysql> select ename,job,sal
    -> from emp
    -> where deptno=30;
+--------+----------+---------+
| ename  | job      | sal     |
+--------+----------+---------+
| ALLEN  | SALESMAN | 1600.00 |
| WARD   | SALESMAN | 1250.00 |
| MARTIN | SALESMAN | 1250.00 |
| BLAKE  | MANAGER  | 2850.00 |
| TURNER | SALESMAN | 1500.00 |
| JAMES  | CLERK    |  950.00 |
+--------+----------+---------+
```

ASSIGNMENT QUESTIONS:

BETWEEN.....AND Operator

- This operator is used to filter records based on given range of values.
- It includes both that start and the end values of the range.

SYNTAX:

```
SELECT column_Name(s)
From table_name
Where column_name BETWEEN value1 AND value2;
```

Note:

Works with numeric,text and date values.

Example:

Selecting employees hired between two dates.

```
mysql> SELECT ename,hiredate
    -> FROM emp
    -> Where hiredate between '1981-04-01' AND
'1981-05-01';
```

ename	hiredate
JONES	1981-04-02
BLAKE	1981-05-01

Ex. Finding products priced between 10 and 50.

Table: products

Columns: Price, Product_name

➔ Select Product_name, price
From products
Where price between 10 and 50;

Ex: Retrieving students with grades between 80 to 90.

IN Operator

It allows us to specify multiple values in where clause.

It filters rows that match any value given in list.

SYNTAX:

SELECT column_name(s)

From table_name

Where column_name IN (value1,value2,……);

Advantage:

Reduces the need of typing multiple OR conditions.

Ex:Selecting employees with specific job titles(Manager & Clerk)

```
Select ename, job
From emp
Where job=’ Manager’  OR job =  ‘Clerk’ ;
```

```
Select ename, job
From emp
Where job IN ( ‘Manager’ ,’ clerk’ );
```

Resultset will be same in both queries but IN is faster one.

Ex. Finding orders with specific status

Table: order

Columns: OrderID, Status(Shipped, pending)

```
Select ordered, status
From order
Where status in ( ‘Shipped’ ,’ pending’ );
```

Ex. Retrieving students enrolled in specific courses.

IS NULL Operator

- This operator tests for NULL values in a column
- NULL represents missing or empty data

SYNTAX:

```
SELECT column_name(s)
From table_name
Where column_name is null;
```

Ex. Selecting employees without any commission.

```
Select ename, comm
From emp
Where comm is null;
```

Ex. Select employee names and salaries where commission is not null.(i.e., employees who receive a commission)

```
Select ename, sal
From emp
Where comm is not null;
```

Ex. Finding orders without a ship date.

```
Select order_id, ship_date
From orders
Where ship_date is null;
```

Ex. Finding products without any description.

LIKE Operator

- Used to search for patterns in column
- It works with two wildcards
  - % - represents zero or more characters
  - _ - represents a single character

Ex. Selecting employees whose name starts with ‘S’.

```
mysql> Select ename
    -> From emp
    -> Where ename like ’S%’;
+-------+
| ename |
+-------+
| SMITH |
| SCOTT |
+-------+
```

Ex. Retrieve names ending with ‘S’

```
mysql> Select ename
    -> From emp
    -> Where ename like '%S';
+-------+
| ename |
+-------+
| JONES |
| ADAMS |
| JAMES |
+-------+
```

Ex. Names containing ‘E’.

```
mysql> select ename
    -> from emp
    -> where ename like '%E%';
+--------+
| ename  |
+--------+
| ALLEN  |
| JONES  |
| BLAKE  |
```

```
| TURNER |
| JAMES  |
| MILLER |
+--------+
```

Ex. Names with ‘L’ at 3rd position.

```
mysql> select ename
    -> from emp
    -> where ename like '__L%';
+--------+
| ename  |
+--------+
| ALLEN  |
| MILLER |
+--------+
```

Ex. 5 character name;

```
SELECT ename
From emp
Where ename like ‘_____’ ; (5 underscores)
```

```
mysql> SELECT ename
    -> From emp
    -> Where ename like '_____';
+-------+
| ename |
+-------+
| SMITH |
| ALLEN |
| JONES |
| BLAKE |
| CLARK |
| SCOTT |
| ADAMS |
| JAMES |
+-------+
```

ORDER BY clause

- The order by clause sorts the resultset in ascending or descending order.

SYNTAX:

```
Select column_name(s)
From table_name
ORDER BY column_name[ASC|DESC];
```

Ex. Sort employees by salary in ascending order.

```
mysql> SELECT ename, sal
    -> From emp
    -> Order by sal asc;
+--------+---------+
| ename  | sal     |
+--------+---------+
| SMITH  |  800.00 |
| JAMES  |  950.00 |
| ADAMS  | 1100.00 |
| WARD   | 1250.00 |
| MARTIN | 1250.00 |
| MILLER | 1300.00 |
```

```
| TURNER | 1500.00 |
| ALLEN  | 1600.00 |
| CLARK  | 2450.00 |
| BLAKE  | 2850.00 |
| JONES  | 2975.00 |
| SCOTT  | 3000.00 |
| FORD   | 3000.00 |
| KING   | 5000.00 |
+--------+---------+
```

Ex. Sorting employees by salary in descending order.

```
Select ename,sal
From emp
Order by sal desc;
```

Ex. Sorting employee names alphabetically

```
mysql> Select ename, sal
    -> From emp
    -> Order by ename asc;
+--------+---------+
```

ename	sal
ADAMS	1100.00
ALLEN	1600.00
BLAKE	2850.00
CLARK	2450.00
FORD	3000.00
JAMES	950.00
JONES	2975.00
KING	5000.00
MARTIN	1250.00
MILLER	1300.00
SCOTT	3000.00
SMITH	800.00
TURNER	1500.00
WARD	1250.00

Ex. Sorting the products by name alphabetically

```
SELECT product_name, price
From products
Order by product_name asc;
```

LIMIT Clause:

It specifies the number of rows to return in the resultset.

SYNTAX:

```
SELECT column_name(s)
From table_name
Limit number_of_records;
```

Ex. To retrieve top 5 highest paid employees.

```
mysql> select ename, sal
    -> from emp
    -> order by sal desc
    -> limit 5;
+-------+---------+
| ename | sal     |
```

```
+-------+---------+
| KING  | 5000.00 |
| SCOTT | 3000.00 |
| FORD  | 3000.00 |
| JONES | 2975.00 |
| BLAKE | 2850.00 |
+-------+---------+
```

Ex. Selecting the first 10 students

```
SELECT student_name, grade
From students
Limit 10;
```