The most fundamental SQL command for data retrieval is the SELECT statement. It is used to fetch data from one or more tables in a database. The syntax for a simple SELECT statement is as follows:

SELECT column1, column2, ...

FROM table_name;

1. Basic Data Retrieval

Suppose we have a table named **employees** with the following structure:

employee_id	first_name	Age	department
1	John Doe	28	HR
2	Jane Smith	34	Finance
3	Jim Brown	25	IT
4	Jake White	30	Marketing
5	Jill Black	29	IT

To retrieve all records from the employees table, you would use:

SELECT * FROM employees;

This query will return **all columns** for all rows in the employees table.

2. Selecting Specific Columns

To retrieve specific columns, you simply list the column names separated by commas.

Example

To retrieve only the name and department columns from the employees table:

SELECT name, department

FROM employees;

This query returns only the name and department columns for all rows in the employees table.

name	department
John Doe	HR
Jane Smith	Finance
Jim Brown	IT
Jake White	Marketing
Jill Black	IT

3. Using Aliases

Aliases can be used to give a table or a column a temporary name that is easier to work with or more descriptive.

Example

To retrieve the name column and give it an alias employee_name, and the department column with an alias dept:

SELECT name AS employee_name, department AS dept

FROM employees;

This query returns the name column as employee_name and the department column as dept.

employee_name	dept
John Doe	HR
Jane Smith	Finance
Jim Brown	IT
Jake White	Marketing
Jill Black	IT

4. Filtering Data with WHERE Clause

The WHERE clause allows you to specify conditions that the data must meet to be included in the results.

Example

To retrieve all employees who are older than 30 years:

SELECT * FROM employees

WHERE age > 30;

This query will return only the employees whose age is greater than 30.

employee_id	name	age	department
2	Jane Smith	34	Finance

5. Filtering Data with IN and NOT IN Command

You can use **IN** and **NOT IN** to filter data based on a list of values.

1. To retrieve employees in the **Sales** or **Marketing** department:

SELECT * FROM employees

WHERE department IN ('Sales', 'Marketing');

employee_id	name	age	department
4	Jake White	30	Marketing

2. To retrieve employees who are **not** in the **Sales** or **Marketing** department:

SELECT * FROM employees

WHERE department NOT IN ('Sales', 'Marketing');

employee_id	first_name	last_name	age	department
1	John	Doe	28	HR
2	Jane	Smith	34	Finance
3	Jim	Brown	25	IT
5	Jill	Black	29	IT

6. Using Multiple Conditions with AND and OR

You can combine multiple conditions using AND and OR.

Example

To retrieve all employees who are older than 25 and work in the IT department:

SELECT * FROM employees

WHERE age > 25 AND department = 'IT';

To retrieve all employees who are either older than 30 or work in the HR department:

SELECT * FROM employees

WHERE age > 30 OR department = 'HR';

employee_id	name	age	department
1	John Doe	28	HR
2	Jane Smith	34	Finance

7. Using Comparison Operators

SQL supports various comparison operators for filtering data:

>: Greater than

<: Less than

>=: Greater than or equal to

<=: Less than or equal to

BETWEEN: Within a range

LIKE: Pattern matching

IS NULL: Is null

IS NOT NULL: Is not null

Example

1. To retrieve all employees whose age is between 25 and 30:

SELECT * FROM employees

WHERE age BETWEEN 25 AND 30;

employee_id	name	age	department
1	John Doe	28	HR
3	Jim Brown	25	IT
4	Jake White	30	Marketing
5	Jill Black	29	IT

2. To retrieve all employees whose name starts with 'J':

SELECT * FROM employees

WHERE name LIKE 'J%';

employee_id	name	age	department
1	John Doe	28	HR
2	Jane Smith	34	Finance
3	Jim Brown	25	IT
4	Jake White	30	Marketing
5	Jill Black	29	IT

Query 3: To Retrieve All Employees Who Do Not Have a Department Assigned

SELECT * FROM employees

WHERE department IS NULL;

Query 4: To Retrieve All Employees Who Do Have a Department Assigned

SELECT * FROM employees

WHERE department IS NOT NULL;

employee_id	first_name	last_name	age	department
1	John	Doe	28	HR
2	Jane	Smith	34	Finance
3	Jim	Brown	25	IT
4	Jake	White	30	Marketing
5	Jill	Black	29	IT

8. Sorting Data with ORDER BY

The ORDER BY clause is used to sort the result set by one or more columns. By default, the sorting is in ascending order. To sort in descending order, you can use the DESC keyword.

Example

To retrieve all employees and sort them by their age in ascending order:

SELECT * FROM employees

ORDER BY age ASC;

To retrieve all employees and sort them by their age in descending order:

SELECT * FROM employees

ORDER BY age DESC;

employee_id	name	age	department
2	Jane Smith	34	Finance
4	Jake White	30	Marketing
5	Jill Black	29	IT
1	John Doe	28	HR
3	Jim Brown	25	IT

9. Multiple Column Sorting

You can sort by multiple columns by listing them separated by commas.

Example

To sort employees by department in ascending order and within each department by age in descending order:

SELECT * FROM employees

ORDER BY department ASC, age DESC;

employee_id	name	age	department
2	Jane Smith	34	Finance
1	John Doe	28	HR
5	Jill Black	29	IT
3	Jim Brown	25	IT
4	Jake White	30	Marketing

10. Using Offsets and Limits

The LIMIT clause is used to specify the number of rows to return, and the OFFSET clause is used to specify the starting point within the result set.

Example

To retrieve the first 3 employees:

SELECT * FROM employees

LIMIT 3;

employee_id	name	age	department
1	John Doe	28	HR
2	Jane Smith	34	Finance
3	Jim Brown	25	IT

To retrieve 3 employees starting from the second row:

SELECT * FROM employees

LIMIT 3 OFFSET 1;

employee_id	name	age	department
2	Jane Smith	34	Finance
3	Jim Brown	25	IT
4	Jake White	30	Marketing

11. Saving SQL Queries in MySQL Workbench

MySQL Workbench is a popular tool for managing MySQL databases. To save an SQL query in MySQL Workbench:

1. Write your SQL query in the query editor.
2. Click on the "File" menu and select "Save Script" or "Save Script As..."
3. Choose a location and provide a name for your SQL file. The default extension for SQL scripts is .sql.
4. Click "Save".

Your SQL query is now saved and can be executed later by opening the saved file in MySQL Workbench.