SQL Introduction with Examples

What is SQL?

SQL (Structured Query Language) is the standard language used to communicate with and manage data in a **Relational Database Management System (RDBMS)** like MySQL, PostgreSQL, SQL Server, or Oracle.

Basic SQL Operations

Operation	Purpose
SELECT	Retrieve data from a table
INSERT	Add new data into a table
UPDATE	Modify existing data
DELETE	Remove data from a table
CREATE TABLE	Create a new table
DROP TABLE	Delete an entire table

Example Table: students

```
CREATE TABLE students (
  id INT PRIMARY KEY,
  name VARCHAR(100),
  age INT,
  grade VARCHAR(10)
);
```

1. Insert Data

```
INSERT INTO students (id, name, age, grade)
VALUES (1, 'Rahul Sharma', 20, 'A');
```

2. Select Data

```
SELECT * FROM students;
```

3. Update Data

```
UPDATE students
SET grade = 'B',name="raj" WHERE id = 1;
```

4. Delete Data

```
DELETE FROM students
WHERE id = 1;
```

5. Filter with WHERE

```
SELECT name, grade
FROM students
WHERE age > 18;
```

6. Sort Results

```
SELECT * FROM students
ORDER BY age DESC;
```

7. Count Records

```
sql
CopyEdit
SELECT COUNT(*) FROM students;
```

Types of SQL Functions

1. Aggregate Functions

Used to perform calculations on a set of values and return a single value.

Function	Description	Example
COUNT()	Counts rows	SELECT COUNT(*) FROM Students;
SUM()	Total of values	SELECT SUM(salary) FROM Employees;
AVG()	Average	SELECT AVG(age) FROM Users;
MAX()	Highest value	SELECT MAX(score) FROM Results;
MIN()	Lowest value	SELECT MIN(score) FROM Results;

2. String Functions

Function	Description	Example
UPPER()	Converts to uppercase	SELECT UPPER(name) FROM Users;
LOWER()	Converts to lowercase	SELECT LOWER(name) FROM Users;
LENGTH()	Length of string	SELECT LENGTH(name) FROM Users;
CONCAT()	Combine strings	SELECT CONCAT(first_name, ' ', last_name) FROM Employees;
SUBSTRING()	Extract part of string	SELECT SUBSTRING(name, 1, 3) FROM Users;

3. Date Functions

Function	Description	Example
NOW()	Current date and time	SELECT NOW();
CURDATE()	Current date	SELECT CURDATE();
YEAR()	Extract year	SELECT YEAR(birthdate) FROM Students;
MONTH()	Extract month	SELECT MONTH(birthdate) FROM Students;
DATEDIFF()	Difference between dates	SELECT DATEDIFF(NOW(), '2024-01-01');

4. Mathematical Functions

Function	Description	Example
ROUND()	Round a number	SELECT ROUND(89.567, 1); → 89.6
CEIL() or CEILING()	Round up	SELECT CEIL(12.3); → 13
FLOOR()	Round down	SELECT FLOOR(12.9); → 12

Function	Description	Example
ABS()	Absolute value	SELECT ABS(-20); → 20
MOD()	Modulus (remainder)	SELECT MOD(10, 3); → 1

💡 Example Query Using Multiple Functions

```
SELECT
  UPPER(name) AS Name_Uppercase,
  ROUND(salary, 2) AS Rounded_Salary,
  YEAR(joining_date) AS Joining_Year
FROM Employees
WHERE salary > 30000;
```

SQL JOINs – Introduction

SQL JOIN is used to **combine rows** from two or more tables, based on a **related column** between them.

Assume Two Tables:

Customers

customer_id	name	city
1	Raj	Delhi
2	Meena	Mumbai
3	Ankit	Jaipur

Orders

order_id	customer_id	product
101	1	Laptop
102	2	Mobile
103	4	Headphones

1. **INNER JOIN**

Returns only the rows where there is a match in **both** tables.

```
SELECT Customers.name, Orders.product
FROM Customers
INNER JOIN Orders ON Customers.customer_id = Orders.customer_id;
```

Result:

name	product
Raj	Laptop
Meena	Mobile

2. **LEFT JOIN (or LEFT OUTER JOIN)**

Returns all records from the **left table (Customers)** and matched records from the right table.

If no match, returns NULL on the right side.

```
SELECT Customers.name, Orders.product
FROM Customers
LEFT JOIN Orders ON Customers.customer_id = Orders.customer_id;
```

Result:

name	product
Raj	Laptop
Meena	Mobile
Ankit	NULL

3. RIGHT JOIN (or RIGHT OUTER JOIN)

Returns all records from the **right table (Orders)** and matched records from the left table.

If no match, returns NULL on the left side.

```
SELECT Customers.name, Orders.product
FROM Customers
RIGHT JOIN Orders ON Customers.customer_id = Orders.customer_id;
```

Result:

name	product
Raj	Laptop
Meena	Mobile
NULL	Headphones

FULL JOIN (or FULL OUTER JOIN)

Returns all records when there is a match in **either left or right** table.

(Note: Not supported in MySQL, can simulate using UNION)

```
-- Not directly supported in MySQL
SELECT Customers.name, Orders.product
FROM Customers
LEFT JOIN Orders ON Customers.customer_id = Orders.customer_id
UNION
SELECT Customers.name, Orders.product
FROM Customers
RIGHT JOIN Orders ON Customers.customer_id = Orders.customer_id;
```

Result:

name	product
Raj	Laptop
Meena	Mobile
Ankit	NULL
NULL	Headphones

Use Case Summary:

JOIN Type	Use Case
INNER	Only matched data needed
LEFT	All left + matched right
RIGHT	All right + matched left
FULL	All data from both sides

Would you like this guide exported to **PDF**, or should I continue with **SELF JOIN, CROSS JOIN, and advanced JOINs** next?