
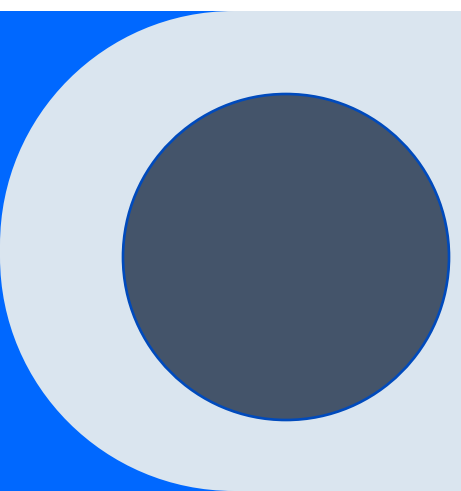


DBMS & MYSQL





Agenda

Introduction

SQL Syntax

SQL Data Types

Aggregate Functions

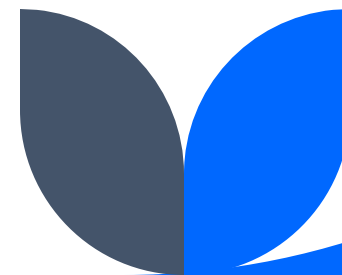
JOINS & IT'S TYPES



Creating Tables

The data in the database (RDBMS) is stored in the database objects called tables. It is a collection of related entries, and it consists of columns and rows.

Table is the most common and simplest form of data storage in a relational database.



Field Record or Row

Every table is broken up into smaller entries called fields. The fields in the CUSTOMERS table consist of ID , NAME , AGE, ADDRESS and SALARY.

A record also called a row of data, is each individual entry that exists in a table. Eg: there are 7 records in the above table mentioned previously.



Column

A column is a vertical entity in a table that contains all information associated with a specific field in a table.

NULL Value

A NULL value in a table is a value in a field that appears to be blank , which means a field with a NULL value is a field with no value.

Getting Started with SQL



SQL

SQL is a standard language for accessing databases.

SQL stands for Structured Query Language and lets you access and manipulate databases.

SQL is an ANSI (American National Standards Institute) standard

Benefits

SQL can execute queries against a database.

SQL can retrieve , insert , update and delete records from a database.

SQL Data Types

INTEGER, SMALLINT, FLOAT, DOUBLE

CHAR , VARCHAR

Date , Time Type & Timestamp

TEXT, BLOBS(Binary Large Objects)

Important SQL Commands

SELECT – extracts data from a database

UPDATE – updates data from a database

DELETE – deletes from a database

INSERT INTO – inserts new data into a database

CREATE DATABASE – creates a new database

Important SQL Commands

ALTER DATABASE– modifies a database

CREATE TABLE – creates a new table

ALTER DATABASE– modifies a table

DROP TABLE – deletes a table

SQL Syntax

CREATE DATABASE

- CREATE DATABASE dbname;

CREATE TABLE

- CREATE TABLE Persons(PersonID, LastName varchar(255), FirstName varchar(255), Address varchar(255), City varchar(255));

SQL Syntax

SQL SELECT Statement

- SELECT * FROM table_name;

SELECT Column Example

- SELECT CustomerName, City FROM Customers;

SELECT DISTINCT Statement

- SELECT DISTINCT column_name,column_name FROM table_name;

SQL WHERE Clause

- SELECT * FROM Customers WHERE Country = 'Mexico';

SQL Syntax

Operators in the where clause:

=, >, <, >=, <=, BETWEEN, LIKE, IN

SQL AND Operators

- SELECT * FROM Customers

WHERE Country = 'Germany' AND City = 'Berlin';

OR Operators Example

SELECT * FROM Customers

WHERE City = 'Berlin'

OR City = 'Munchen';



SQL Syntax

Combining AND & OR:

```
SELECT * FROM Customers
```

```
WHERE Country='Germany'
```

```
AND (City='Berlin' OR City='München');
```


Aggregate Functions

SQL Aggregate functions return a single value , calculated from values in a column.

Aggregate Functions

SUM() - Returns the sum

AVG() - Returns the average value

COUNT() - Returns the number of rows

MAX() - Returns the Largest Value

MIN() - Returns the smallest value

FIRST() - Returns the first value

LAST() - Returns the last value

SUM()

Syntax:

```
SELECT SUM(Expression) FROM tables WHERE conditions;
```

Examples:

```
SELECT SUM(Salary) AS "Total Salary" FROM Employee  
WHERE salary > 20000;
```

AVG()

Syntax:

```
SELECT AVG(Expression) FROM table WHERE  
conditions;
```

Examples:

```
SELECT AVG(Salary) AS "Average Salary" FROM Employee  
WHERE department = 'Sales';
```

COUNT()

Syntax:

```
SELECT COUNT(Expression) FROM table WHERE conditions;
```

Examples:

```
SELECT COUNT(*) AS "Total Employees" FROM Employee  
WHERE department = 'Sales';
```

MAX()

Syntax:

```
SELECT MAX(Expression) FROM table WHERE conditions;
```

Examples:

```
SELECT MAX(Salary) AS "Highest Salary" FROM Employee  
WHERE department = 'Sales';
```

MIN()

Syntax:

```
SELECT MIN(Expression) FROM table WHERE conditions;
```

Examples:

```
SELECT MIN(Salary) AS "Lowest Salary" FROM Employee  
WHERE department = 'Sales';
```

FIRST()

Syntax:

```
SELECT FIRST(Expression) FROM table WHERE conditions;
```

Examples:

```
SELECT FIRST(Salary) AS "First Salary" FROM Employee  
ORDER BY join_date;
```


LAST()

Syntax:

```
SELECT LAST(Expression) FROM table WHERE  
conditions;
```

Examples:

```
SELECT LAST(Salary) AS "Last Salary" FROM Employee ORDER  
BY join_date DESC;
```

GROUP BY

The GROUP BY clause is a SQL command that is used to group rows that have the same values.

It is used in conjunction with aggregate functions to produce summary reports from the database.

GROUP BY clause are called grouped queries and only return a single row for every grouped item.



GROUP BY

Syntax:

```
SELECT column_name, aggregate_function(column_name ) FROM  
table_name WHERE column_name operator value  
GROUP BY column name;
```

Example:

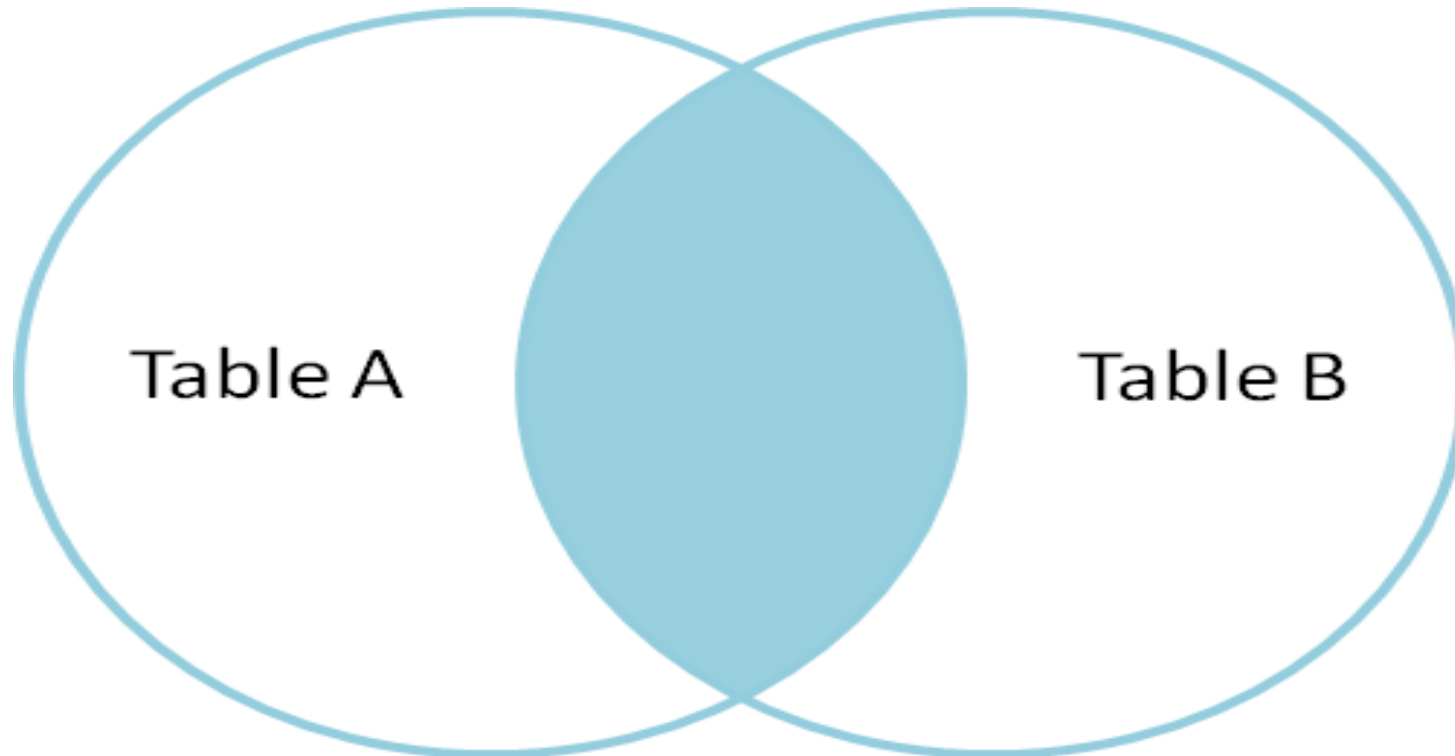
```
SELECT Department, SUM(Salary) AS "Total Salary " FROM  
Employee  
GROUP BY Department;
```

SQL JOINS

An SQL JOIN clause is used to combine rows from two or more tables, based on a common field between them.

Applications : E-Commerce , HR & Payroll , Healthcare , Banking , Education , Social Media.

SQL JOINS



SQL JOINS

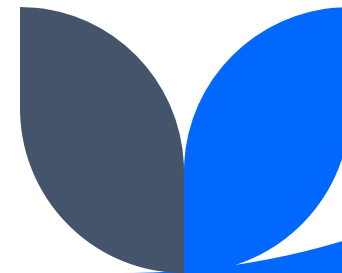
The different SQL JOINS you can use:

INNER JOIN: Returns all rows when there is at least one match in BOTH tables

LEFT JOIN: Return all rows from the left table, and the matched rows from the right table. The result is null in the right side if there is no match.

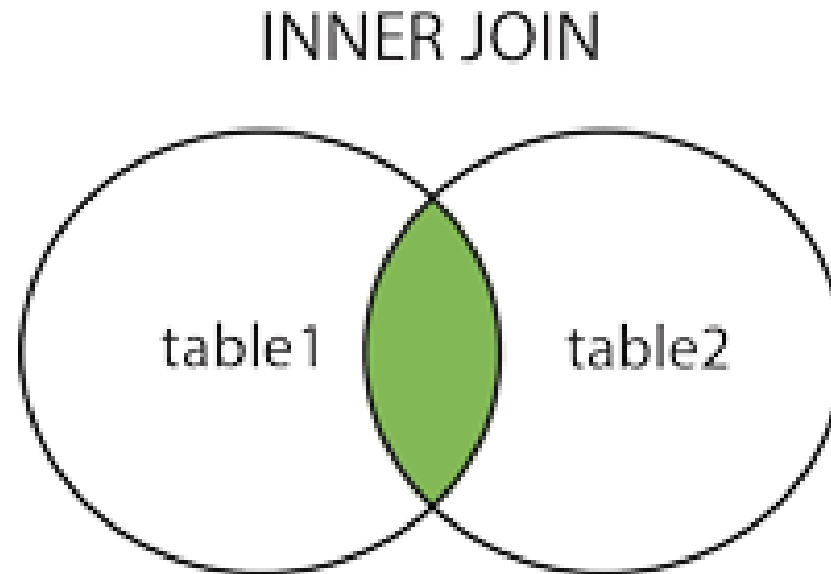
RIGHT JOIN: Return all rows from the right table, and the matched rows from the left table. The result is null in the right side if there is no match.

FULL JOIN: Return all rows from the left table and right table. It combines the result of both LEFT and RIGHT join.



SQL Inner JOINS

The most common type of join is SQL INNER JOIN (simple join). An SQL INNER JOIN returns all rows from multiple tables where the join condition is met.



SQL Inner JOINS

OrderID	CustomerID	OrderDate
10308	2	1996-09-18
10309	37	1996-09-19
10310	77	1996-09-20

SQL Inner JOINS

CustomerID	CustomerName	ContactName	Country
1	Robert Brown	Maria Anders	Germany
2	Jack Carls	Trujilo	Mexico
3	Antonio Moreno	Mathews	Mexico

Notice that "CustomerID" column in the "Orders" Table refers to the customer in the "customers" table. The relationship b/w the two tables above is the "CustomerID" column.

SQL Inner JOINS

Example:

```
SELECT Orders.OrderId, Customers.CustomerName,  
Orders.OrderDate
```

```
FROM Orders
```

```
INNER JOIN Customers
```

```
ON Orders.CustomerID=Customers.CustomerID;
```

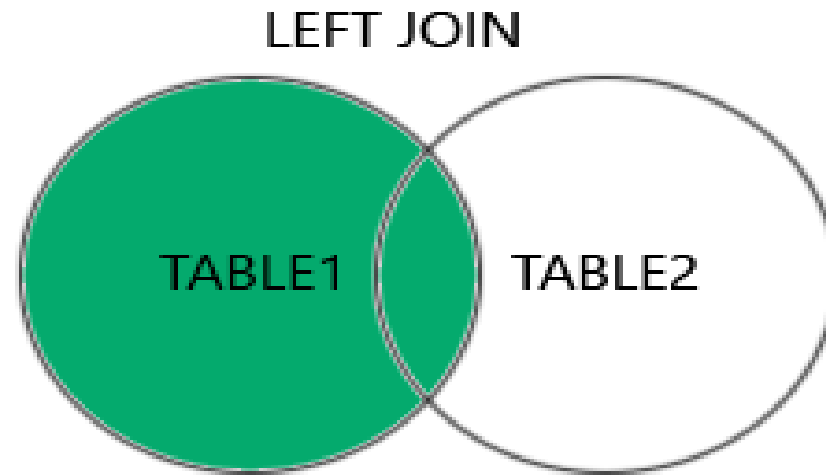
Output:

OrderID	CustomerName	OrderDate
10308	Trujilo	1996-09-18

SQL Left JOIN

Syntax :

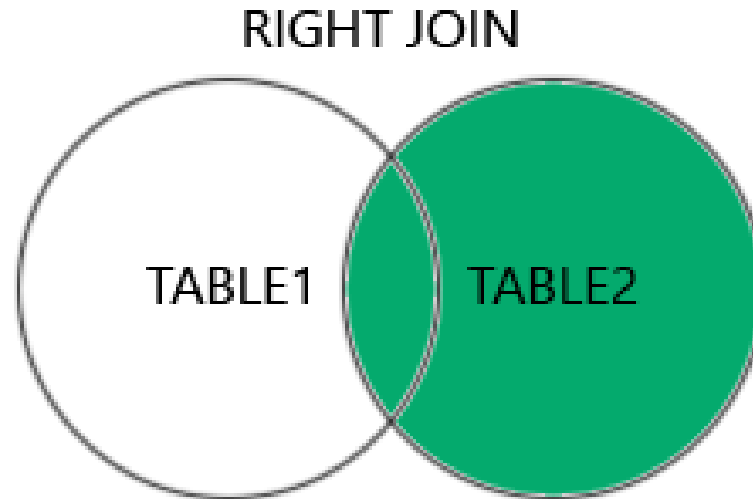
```
SELECT column_name(s) FROM table1 LEFT JOIN table2 ON  
table1.column_name = table2.column_name;
```



SQL Right JOIN

Syntax :

```
SELECT column_name(s) FROM table1 RIGHT JOIN table2 ON  
table1.column_name = table2.column_name;
```

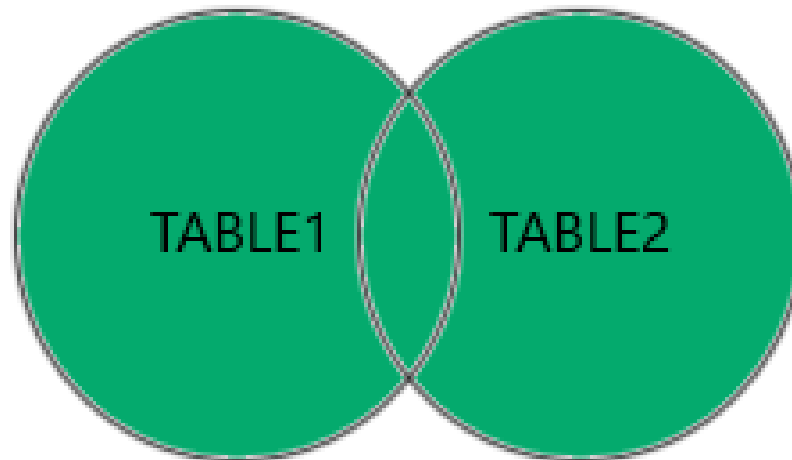


SQL Right JOIN

Syntax :

```
SELECT column_name(s) FROM table1 FULL JOIN table2 ON  
table1.column_name = table2.column_name;
```

FULL OUTER JOIN





Thank you !

Kevin Thakkar

Neel Amrutia