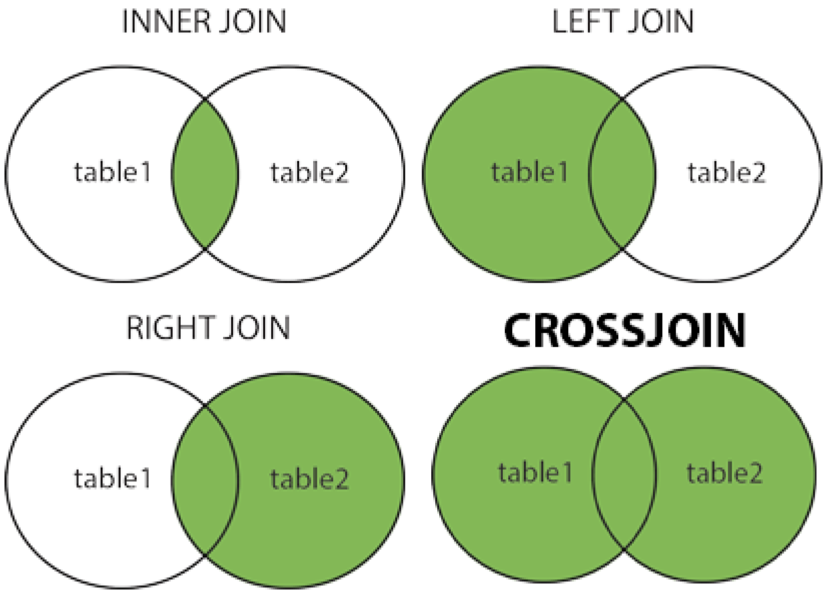
❖MySQL Joining Tables

```
SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate
FROM Orders
INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;
```

- INNER JOIN: Returns records that have matching values in both tables
- LEFT JOIN: Returns all records from the left table, and the matched records from the right table
- RIGHT JOIN: Returns all records from the right table, and the matched records from the left table
- CROSS JOIN: Returns all records from both tables



➔INNER JOIN

```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

```
SELECT Orders.OrderID, Customers.CustomerName, Shippers.ShipperName
FROM ((Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID)
INNER JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID);
```

➔LEFT JOIN

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.CustomerName;
```

➔RIGHT JOIN

```
SELECT Orders.OrderID, Employees.LastName, Employees.FirstName
FROM Orders
RIGHT JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID
ORDER BY Orders.OrderID;
```

➔CROSS JOIN

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
CROSS JOIN Orders;
```

➔ SELF JOIN

```
SELECT A.CustomerName AS CustomerName1,
B.CustomerName AS CustomerName2, A.City
FROM Customers A, Customers B
WHERE A.CustomerID <> B.CustomerID
AND A.City = B.City
ORDER BY A.City;
```

➔ MySQL NULL Values

```
SELECT CustomerName, ContactName, Address
FROM Customers
WHERE Address IS NULL;
```

```
SELECT CustomerName, ContactName, Address
FROM Customers
WHERE Address IS NOT NULL;
```

➔ UNION and UNION ALL Operator

```
SELECT City FROM Customers
UNION
SELECT City FROM Suppliers
ORDER BY City;
```

```
SELECT City FROM Customers
UNION ALL
SELECT City FROM Suppliers
ORDER BY City;
```

➔ GROUP BY CLAUSE

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country;
```

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
ORDER BY COUNT(CustomerID) DESC;
```

```
SELECT Shippers.ShipperName, COUNT(Orders.OrderID) AS NumberOfOrders
FROM Orders
LEFT JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID
GROUP BY ShipperName;
```

➔ HAVING CLAUSE

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
HAVING COUNT(CustomerID) > 5;
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
HAVING COUNT(CustomerID) > 5
ORDER BY COUNT(CustomerID) DESC;
```

```
SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOfOrders
FROM (Orders
INNER JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID)
```

```
GROUP BY LastName
HAVING COUNT(Orders.OrderID) > 10;

SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOfOrders
FROM Orders
INNER JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID
WHERE LastName = 'Davolio' OR LastName = 'Fuller'
GROUP BY LastName
HAVING COUNT(Orders.OrderID) > 25;
```

EXISTS OPERATOR

```
SELECT SupplierName
FROM Suppliers
WHERE EXISTS (SELECT ProductName FROM Products WHERE Products.
SupplierID = Suppliers.supplierID AND Price < 20);

SELECT SupplierName
FROM Suppliers
WHERE EXISTS (SELECT ProductName FROM Products WHERE Products.
SupplierID = Suppliers.supplierID AND Price = 22);
```

MySQL ANY and ALL Operators

```
SELECT ProductName
FROM Products
WHERE ProductID = ANY
  (SELECT ProductID
  FROM OrderDetails
  WHERE Quantity = 10);

SELECT ProductName
FROM Products
WHERE ProductID = ANY
```

```
  (SELECT ProductID
  FROM OrderDetails
  WHERE Quantity > 99);
```

```
SELECT ProductName
FROM Products
WHERE ProductID = ANY
  (SELECT ProductID
  FROM OrderDetails
  WHERE Quantity > 1000);
```

```
SELECT ProductName
FROM Products
WHERE ProductID = ANY
  (SELECT ProductID
  FROM OrderDetails
  WHERE Quantity > 1000);
```

```
SELECT ALL ProductName
FROM Products
WHERE TRUE;
```

```
SELECT ProductName
FROM Products
WHERE ProductID = ALL
  (SELECT ProductID
  FROM OrderDetails
  WHERE Quantity = 10);
```

❖ INSERT INTO - inserts new data into a database

```
INSERT INTO Customers (CustomerName, ContactName, Address, City,
PostalCode, Country) VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen
21', 'Stavanger', '4006', 'Norway');
```

```
INSERT INTO Customers (CustomerName, City, Country)
VALUES ('Cardinal', 'Stavanger', 'Norway');
```

➔ INSERT INTO SELECT Statement

```
INSERT INTO Customers (CustomerName, City, Country)
SELECT SupplierName, City, Country FROM Suppliers;

INSERT INTO Customers (CustomerName, ContactName, Address, City,
PostalCode, Country)
SELECT SupplierName, ContactName, Address, City,
PostalCode, Country FROM Suppliers;

INSERT INTO Customers (CustomerName, City, Country)
SELECT SupplierName, City, Country FROM Suppliers
WHERE Country='Germany';
```

➔ CASE STATEMENT

```
SELECT OrderID, Quantity,
CASE
   WHEN Quantity > 30 THEN 'The quantity is greater than 30'
   WHEN Quantity = 30 THEN 'The quantity is 30'
   ELSE 'The quantity is under 30'
END AS QuantityText
FROM OrderDetails;

SELECT CustomerName, City, Country
FROM Customers
ORDER BY
(CASE
    WHEN City IS NULL THEN Country
    ELSE City
END);
```

➔ IFNULL() and COALESCE() Functions

```
SELECT ProductName, UnitPrice * (UnitsInStock + UnitsOnOrder)
FROM Products;
```

```
SELECT ProductName, UnitPrice * (UnitsInStock + IFNULL(UnitsOnOrder, 0))
FROM Products;
```

```
SELECT ProductName, UnitPrice * (UnitsInStock
+ COALESCE(UnitsOnOrder, 0))
FROM Products;
```

➔ Comments

```
-- Select all:
SELECT * FROM Customers;
```

```
SELECT * FROM Customers -- WHERE City='Berlin';
```

```
-- SELECT * FROM Customers;
SELECT * FROM Products;
```

➔ Multi-line Comments

```
/*Select all the columns
of all the records
in the Customers table:*/
SELECT * FROM Customers;
```

```
/*SELECT * FROM Customers;
SELECT * FROM Products;
SELECT * FROM Orders;
SELECT * FROM Categories;*/
SELECT * FROM Suppliers;


SELECT CustomerName, /*City,*/ Country FROM Customers;


SELECT * FROM Customers WHERE (CustomerName LIKE 'L%'
OR CustomerName LIKE 'R%' /*OR CustomerName LIKE 'S%'
OR CustomerName LIKE 'T%'*/ OR CustomerName LIKE 'W%')
AND Country='USA'
ORDER BY CustomerName;
```

➔ MySQL Operators

- ➢ Arithmetic Operators

  + -   *   /   %

- ➢ Bitwise Operators

  & | ^

- ➢ Comparison Operators

  =<   >   >=   <=   <>

- ➢ Compound Operators

  +=   -=   *=   /=   %=   &=   ^-= |*=

- ➢ Logical Operators

  ALL   AND   ANY   BETWEEN   EXISTS   IN   LIKE   NOT   OR   SOME