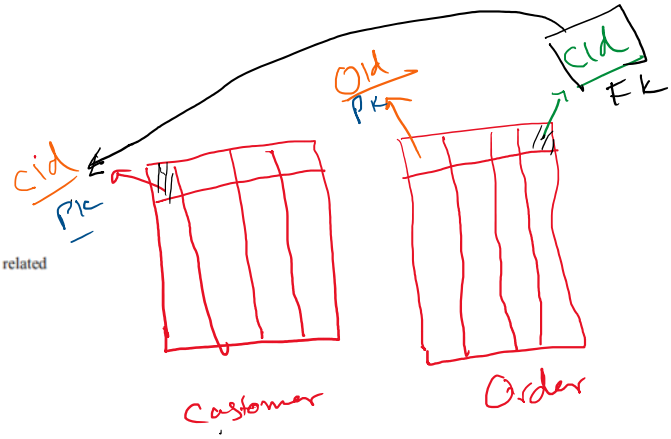
Session- 21 Database

22 August 2024 11:53 AM

Agenda

- **Keys, Constraints and Clauses**
- **Operators and AggregateFunctions**
- **Basic SQL Queries**
- **Data Filtering and Sorting Data**
- **Joining Tables**
- **Joins -Inner Join -Left Join -Right Join -Full Outer Join -Self**
- **Join Subqueries Nested Queries**



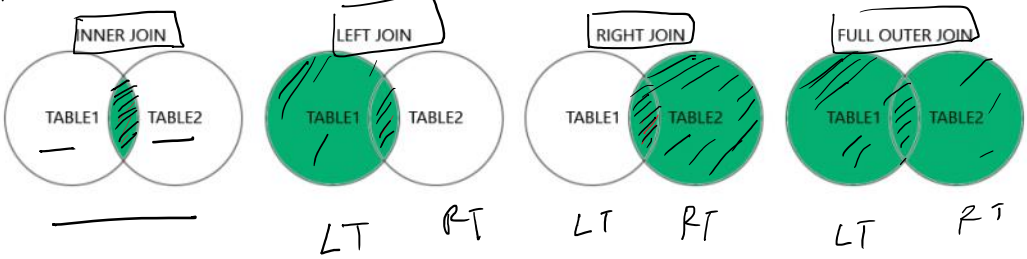
SQL JOIN

The SQL Join help in retrieving data from two or more database tables. The tables are mutually related using primary keys and foreign keys.

Different Types of SQL JOINs

Here are the different types of the JOINs in SQL:

1. `(INNER) JOIN` : Returns records that have matching values in both tables
2. `LEFT (OUTER) JOIN` : Returns all records from the left table, and the matched records from the right table
3. `RIGHT (OUTER) JOIN` : Returns all records from the right table, and the matched records from the left table
4. `FULL (OUTER) JOIN` : Returns all records when there is a match in either left or right table



1 INNER JOIN

The `INNER JOIN` keyword selects records that have matching values in both tables.

Let's look at a selection of the **Products** table:

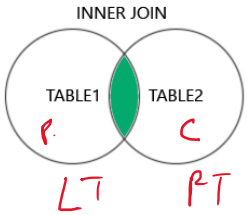
ProductID	ProductName	CategoryID	Price
1	Chais	1	18
2	Chang	1	19
3	Aniseed Syrup	2	10

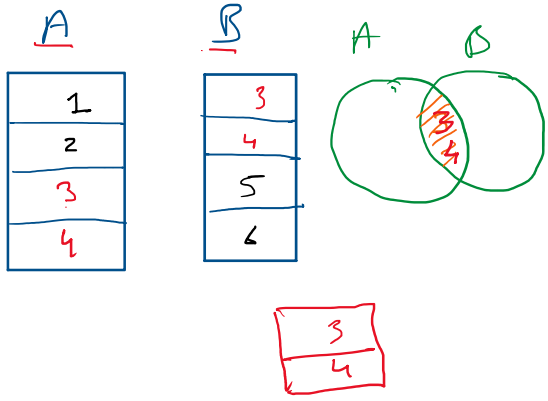
And a selection of the **Categories** table:

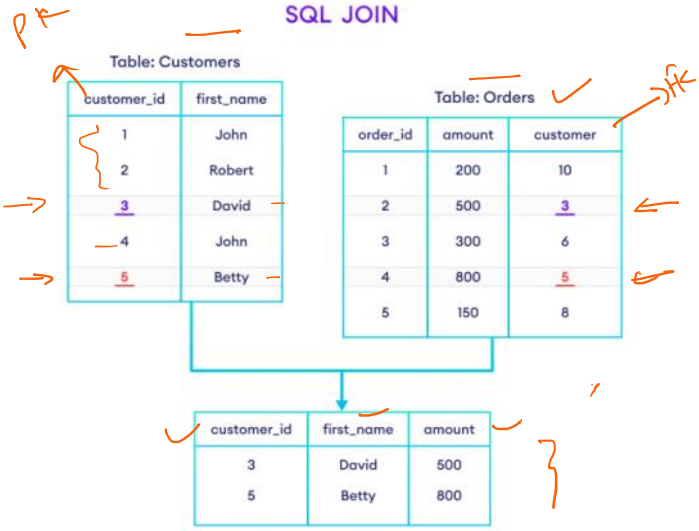
CategoryID	CategoryName	Description
1	Beverages	Soft drinks, coffees, teas, beers, and ales
2	Condiments	Sweet and savory sauces, relishes, spreads, and seasonings
3	Confections	Desserts, candies, and sweet breads

We will join the Products table with the Categories table, by using the `CategoryID` field from both tables:

```
SELECT ProductID, ProductName, CategoryName
FROM Products
INNER JOIN Categories ON Products.CategoryID = Categories.CategoryID;
```







Join Two Table Based on Common Column

Note: The `INNER JOIN` keyword returns only rows with a match in both tables. Which means that if you have a product with no CategoryID, or with a CategoryID that is not present in the Categories table, that record would not be returned in the result.

ProductID	ProductName	CategoryName
39	Chartreuse verte	Beverages
2	Chang	Beverages
24	Guaraná Fantástica	Beverages
34	Sasquatch Ale	Beverages
35	Steeleye Stout	Beverages
1	Chais	Beverages
38	Côte de Blaye	Beverages
43	Ipoh Coffee	Beverages

Syntax

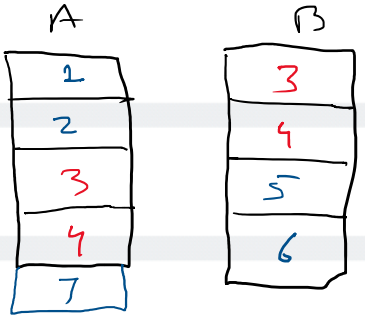
```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```

SQL LEFT JOIN Keyword

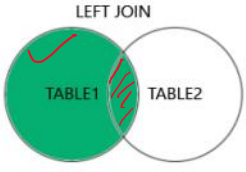
The `LEFT JOIN` keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side, if there is no match.

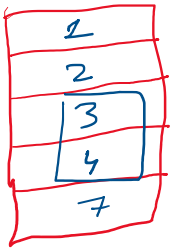
LEFT JOIN Syntax

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



Note: In some databases LEFT JOIN is called LEFT OUTER JOIN.





Below is a selection from the "Customers" table:

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico

And a selection from the "Orders" table:

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	1996-09-18	3
10309	37	3	1996-09-19	1
10310	77	8	1996-09-20	2

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

CustomerName	OrderID
Alfreds Futterkiste	
Ana Trujillo Emparedados y helados	10308
Antonio Moreno Taquería	10365
Around the Horn	10383
Around the Horn	10355
Berglunds snabbköp	10278
Berglunds snabbköp	10280
Berglunds snabbköp	10384

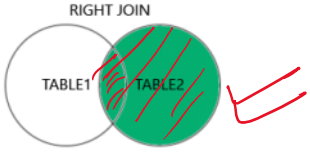
SQL RIGHT JOIN Keyword

The `RIGHT JOIN` keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match.

RIGHT JOIN Syntax

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

Note: In some databases `RIGHT JOIN` is called `RIGHT OUTER JOIN`.



Below is a selection from the "Orders" table:

Below is a selection from the "Orders" table:

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	1996-09-18	3
10309	37	3	1996-09-19	1
10310	77	8	1996-09-20	2

And a selection from the "Employees" table:

EmployeeID	LastName	FirstName	BirthDate	Photo
1	Davolio	Nancy	12/8/1968	EmpID1.pic
2	Fuller	Andrew	2/19/1952	EmpID2.pic
3	Leverling	Janet	8/30/1963	EmpID3.pic

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

OrderID	LastName	FirstName
	West	Adam
10248	Buchanan	Steven
10249	Suyama	Michael
10250	Peacock	Margaret
10251	Leverling	Janet
10252	Peacock	Margaret
10253	Leverling	Janet
10254	Buchanan	Steven

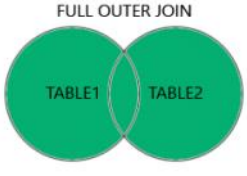
SQL FULL OUTER JOIN Keyword

The `FULL OUTER JOIN` keyword returns all records when there is a match in left (table1) or right (table2) table records.

Tip: `FULL OUTER JOIN` and `FULL JOIN` are the same.

FULL OUTER JOIN Syntax

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



Note: `FULL OUTER JOIN` can potentially return very large result-sets!

Below is a selection from the "Customers" table:

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico

And a selection from the "Orders" table:

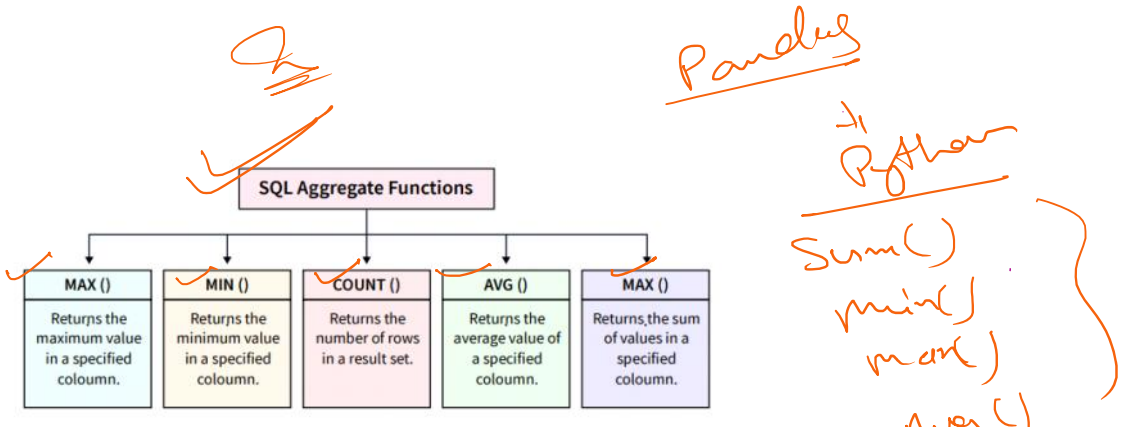
OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	1996-09-18	3
10309	37	3	1996-09-19	1
10310	77	8	1996-09-20	2

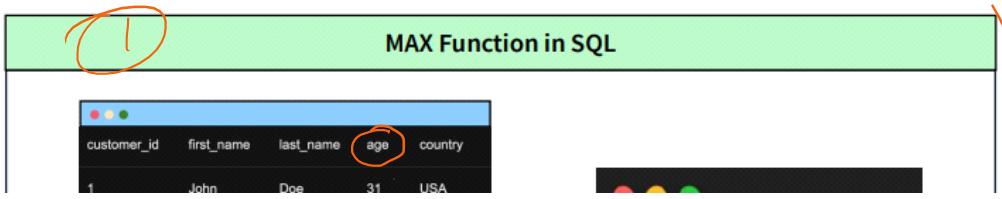
```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
FULL OUTER JOIN Orders ON Customers.CustomerID=Orders.CustomerID
ORDER BY Customers.CustomerName;
```

A selection from the result set may look like this:

CustomerName	OrderID
Null	10309
Null	10310
Alfreds Futterkiste	*Null*
Ana Trujillo Emparedados y helados	10308
Antonio Moreno Taquería	*Null*

Note: The `FULL OUTER JOIN` keyword returns all matching records from both tables whether the other table matches or not. So, if there are rows in "Customers" that do not have matches in "Orders", or if there are rows in "Orders" that do not have matches in "Customers", those rows will be listed as well.





MAX Function in SQL

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

```
SELECT MAX(age)
FROM Customers;
```

MAX
31

MIN Function in SQL

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

```
SELECT MIN(age)
FROM Customers;
```

MIN(age)
22

COUNT Function in SQL

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

```
SELECT COUNT(DISTINCT Country)
FROM Customers;
```

COUNT(DISTINCT Country)

COUNT(DISTINCT Country)
3

AVG Function SQL

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

```
SELECT AVG(age) AS average_age
FROM Customers;
```

average_age
25.6

SUM Function in SQL

order_id	item	amount	customer_id
1	Keyboard	400	4
2	Mouse	300	4
3	Mouse	12000	3
4	Keyboard	400	1
5	Mousepad	250	2

```
SELECT SUM(amount) AS total_sales
FROM Orders;
```

total_sales
133509