Data Bhau

Class 4
Joins - II

What is Self JOIN?

A self-join is a Structured Query Language (SQL) statement that joins a table with itself. An inner join is performed on a single table, especially when comparing records from the same table to determine a relationship.

Structure of a Self Join Query

```
SELECT a.column_1 , b.column_2

FROM table_name a, table_name b

WHERE some_condition;
```

How SELF JOIN work?

A self-join joins a table with itself based on a predefined condition. Assume we have a group of students, and one of them has a best friend relationship with another student at the same table, and we want to know the names of the student and his friend.

Id	Student_name	Friend_id
1	JAY	3
2	RAY	1
3	MAY	2

How SELF JOIN work?

Now, to obtain the name of each student and their friend, we can use a self-join, which will join the table in the following manner if friend id equals student id.

id	Student_name	Friend_id
1	JAY	3
2	RAY	1
3	MAY	2

id	Student_name	Friend_id
1	JAY	3
2	RAY	1
3	MAY	2

How SELF JOIN work?

The Resultant table will look like this.

Id	Student_name	Friend
1	JAY	MAY
2	RAY	JAY
3	MAY	RAY

Understanding the use case of self join

This is a table called customers. We will try using self join here

ID	Customer	City	Country	Items_purchased	Amount_paid
1	Peter King	Manchester	England	Books	120
2	Priya Krishna	New Delhi	India	pen	50
3	Jim Halpert	Manchester	England	pencil	43
4	Michael Scott	New York	USA	Books	250
5	Harvey Spector	Birmingham	England	pencil	100

Self Join – Example 1

Find the pairs of customers who belong to the same city

```
select distinct a.customer, b.customer, a.city, b.city

from

customer as a

join

customer as b

on a.id<>b.id

where a.city = b.city
```

customer	customer	city
Jim Halpert	Peter King	Manchester

What is CARTESIAN/CROSS JOIN?

The CARTESIAN JOIN is also referred to as the CROSS JOIN. Each row of one table is joined to every row of another table in a CARTESIAN JOIN. This usually occurs when no matching column or WHERE condition is specified.

- The SQL Cross Join is useful for obtaining all the combinations of items from two different records of two other collections of tables, such as colors or sizes, etc., at the business level or where an extensive database of this type is present in the industries.
- Assume we have two tables X and Y with 'n' and ‘m’ rows respectively, and when we perform a Cross join, each row of each table is paired to each other, and the resultant set contains n*m rows

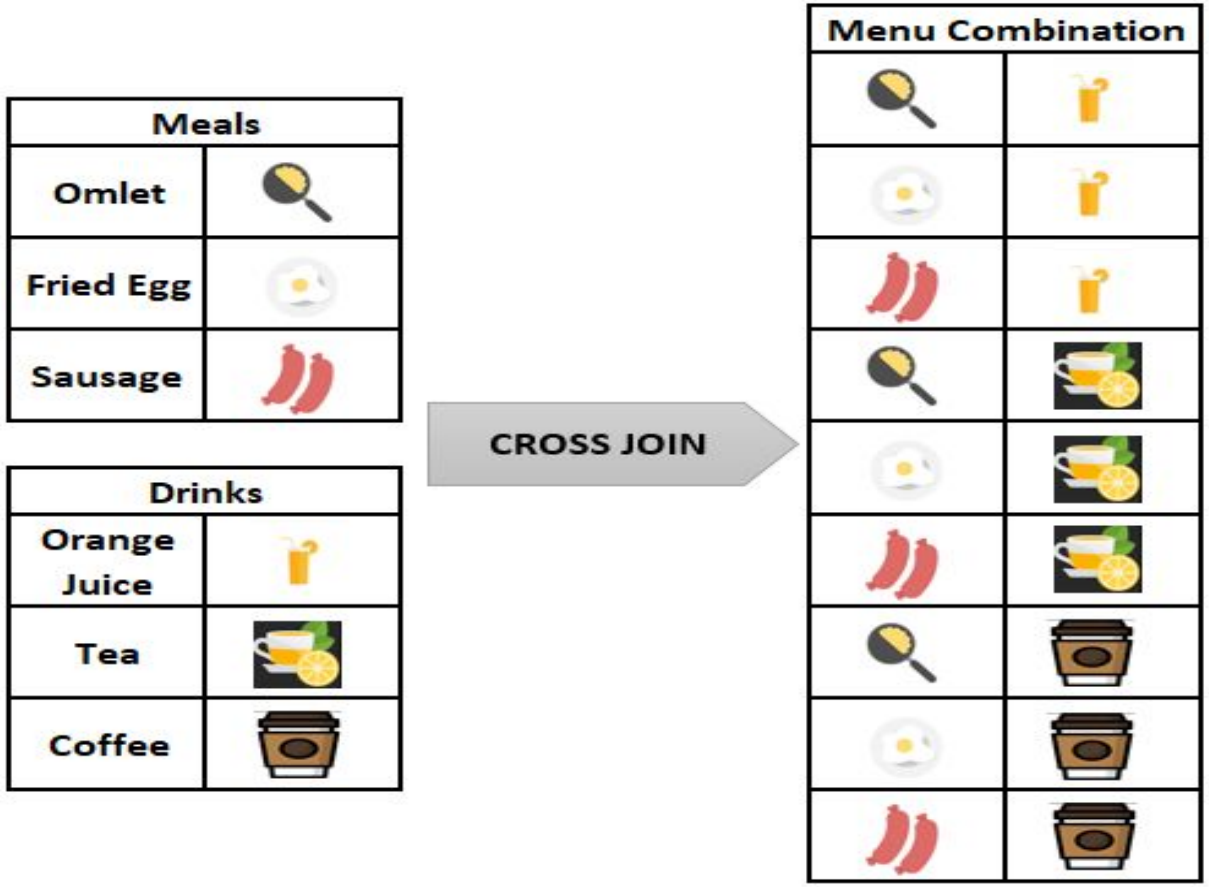
Structure of a Cross Join Query

```
SELECT table1.column1 , table1.column2, table2.column1...

FROM table1

CROSS JOIN table2;
```

What combination should you eat for Breakfast?



Cross Join – Example 1

T-shirt Table

tshirt	
id	**size**
1	S
2	M
3	L
4	XL

Colour Table

id	name
1	yellow
2	green
3	pink

Query

```
SELECT *
FROM tshirt
CROSS JOIN color;
```

Output

id	size	id	name
1	S	1	yellow
2	M	1	yellow
3	L	1	yellow
4	XL	1	yellow
1	S	2	green
2	M	2	green
3	L	2	green
4	XL	2	green
1	S	3	pink
2	M	3	pink
3	L	3	pink
4	XL	3	pink

Cross Join – Example 2

INPUT

e_name	age	salary
John	10	2000
Jane	50	1000
Johnny	20	2500

employee

dept_name	id
Sales	10
Marketing	20

department

QUERY

```
SELECT *
FROM employee
CROSS JOIN department
```

OUTPUT

e_name	age	salary	dept_name	id
John	10	2000	Sales	10
John	10	2000	Marketing	20
Jane	50	1000	Sales	10
Jane	50	1000	Marketing	20
Johnny	20	2500	Sales	10
Johnny	20	2500	Marketing	20

UNION

Union combines results of two or more select statements. Thereafter, it will return the result without any duplicate rows. To perform this operation, the tables should have the same number of columns and same data types.

Structure of a Union Query

```
SELECT * FROM table A

UNION

SELECT * FROM table B;
```

UNION

The union all is another SQL command to perform set operations. Similar to Union, this will also combine results of two or more select statements. It is also necessary to have the same number of columns and same data types to the tables which the union all operation applies to.

Structure of a Union Query

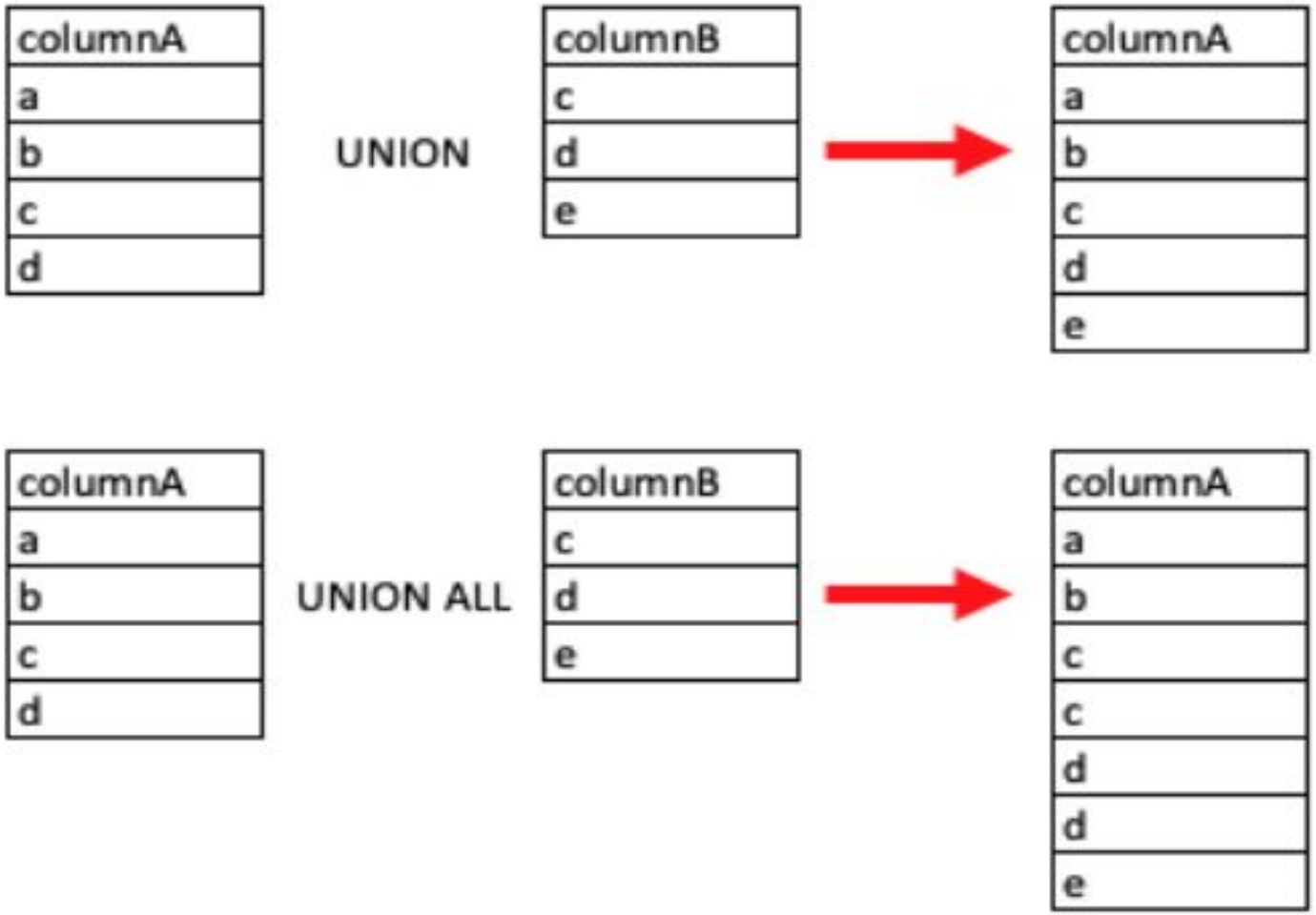
```
SELECT * FROM table A
UNION ALL
SELECT * FROM table B;
```

Conditions for performing UNION and UNION ALL

The following conditions must be met to perform the UNION and UNION ALL operations for combining the tables:

- Each SELECT statement to be combined should return the same number of columns.
- Each SELECT statements' columns must be of the same data type.
- Columns returned by each SELECT statement must be returned in the same order.

UNION vs UNION ALL



Practice Question - 1

https://platform.stratascratch.com/coding/9894-employee-and-manager-salaries?python=

QUESTION

Find employees who are earning more than their managers. Output the employee's first name along with the corresponding salary.

SOLUTION

```
SELECT a.first_name AS employee_name, a.salary AS employee_salary
FROM employee AS employee_table
JOIN employee AS manager_table
ON a.manager_id = b.id
WHERE a.salary > b.salary;
```

Practice Question - 2

Write a query to return the employee and the manager they report to.

INPUT TABLE

Employee

Id	Name	Salary	ManagerId
1	Greg	100000	1
2	George	150000	1
3	Helen	130000	1
4	Tom	120000	2
5	Kevin	110000	2
6	David	120000	3
7	Geek	110000	3
8	Lucy	150000	3
9	Tesla	120000	3

SOLUTION

```
SELECT e.Name AS Employee_name, m.Name as Manager_Name
FROM Employee e
JOIN Employee m
ON e.ManagerId = m.Id
order by 1,2
```

Practice Question - 2

Write a query to return the name of manager, number of team members, and sum of total salary of the team.

INPUT TABLE

Employee

Id	Name	Salary	ManagerId
1	Greg	100000	1
2	George	150000	1
3	Helen	130000	1
4	Tom	120000	2
5	Kevin	110000	2
6	David	120000	3
7	Geek	110000	3
8	Lucy	150000	3
9	Tesla	120000	3

SOLUTION

```
SELECT m.Name AS Manager_Name, count(m.Id) AS team_amount, sum(e.salary) AS
Total_Salary
FROM Employees e
JOIN Employees m
ON e.ManagerId = m.Id
GROUP BY m.name
```

Practice Question 2 - Output

Manager_Name	team_amount	Total_Salary
George	2	230000
Greg	3	380000
Helen	4	500000

Practice Question - 3

Write a query to combine the suppliers table (where supplier_id is greater than 2000) with companies table (where company id is greater than 1000)

INPUT TABLE

Suppliers

supplier_id	supplier_name
1000	Microsoft
2000	Oracle
3000	Apple
4000	Samsung

Companies

company_id	company_name
1000	Microsoft
3000	Apple
7000	Sony
8000	IBM

SOLUTION

```
SELECT supplier_id AS ID_Value, supplier_name AS Name_Value
FROM suppliers
WHERE supplier_id > 2000
UNION
SELECT company_id AS ID_Value, company_name AS Name_Value
FROM companies
WHERE company_id > 1000
ORDER BY 1;
```

Practice Question 3 - Output

Output

ID_Value	Name_Value
3000	Apple
4000	Samsung
7000	Sony
8000	IBM

Thanks for attending

DataBhau communication platforms:

- **Whatsapp :** https://chat.whatsapp.com/BaP2CAajm9597J8LwzYE3j
- **Linkedin Handle:** https://www.linkedin.com/company/databhau/
- **Instructors Linkedin Handles:**
  - Shrey Jain : https://www.linkedin.com/in/shrey-jain-74a90b13b/
  - Harsh Katyayan : https://www.linkedin.com/in/harsh-katyayan-a2248316b/