Tishk International University
Department of Information Technology
Database Systems 1
Week 8
Fall 2023-24
November 18, 2023

Union and Joins

Wisam Abdulaziz Qadir
Wisam.abdulaziz@tiu.edu.iq

Outline

- Union
- Basic **Joins** in MySQL
  - Cross Join
  - Inner Join
  - Outer Join

Union

- The **UNION** operator lets you combine two or more SELECT statements.

- The SELECT statements that you combine must:
  - have the same number of fields
  - have the same or compatible data types
  - be in the same order

Union (cont.)

If you want to show duplicate values which exist in the tables.

Syntax:

```
SELECT column_name(s)
FROM table_name_1
UNION ALL
SELECT column_name(s)
FROM table_name_2;
```

If you want to ignore duplicate values which exist in the tables.

Syntax:

```
SELECT column_name(s)
FROM table_name_1
UNION
SELECT column_name(s)
FROM table_name_2;
```

Union (cont.)

- E.g.: Show list of deposit and received amounts from Transaction of all users according to the given tables.

DID	F_name	Amount
1	Dara	500 $
2	Zara	400 $
3	Ali	300 $
4	Dara	600 $

Deposit

TID	F_name_sender	Amount	F_name_receiver
1	Azad	5000 $	Ali
2	Nawzad	4000 $	Dara
3	Omer	6000 $	Zara
4	Dana	3000 $	Ali

Transaction

Union (cont.)

```
SELECT F_name, Amount
FROM Deposit
UNION ALL
SELECT F_name_receiver, Amount
FROM Transaction;
```

F_name	Amount
Dara	500 $
Zara	400 $
Ali	300 $
Dara	600 $
Ali	5000 $
Dara	4000 $
Zara	6000 $
Ali	3000 $

Query_output

Union (cont.)

With ordering

```
SELECT F_name, Amount
FROM Deposit
UNION ALL
SELECT F_name_receiver, Amount
FROM Transaction
ORDER BY f_name;
```

F_name	Amount
Ali	300 $
Ali	300 $
Ali	3000 $
Dara	500 $
Dara	600 $
Dara	4000 $
Zara	400 $
Zara	6000 $

Query_output

Union (cont.)

Only for Ali

```
SELECT F_name, Amount
FROM Deposit
WHERE F_name='Ali'
UNION ALL
SELECT F_name_receiver, Amount
FROM Transaction
WHERE F_name='Ali';
```

F_name	Amount
Ali	300 $
Ali	300 $
Ali	3000 $

Query_output

Joins

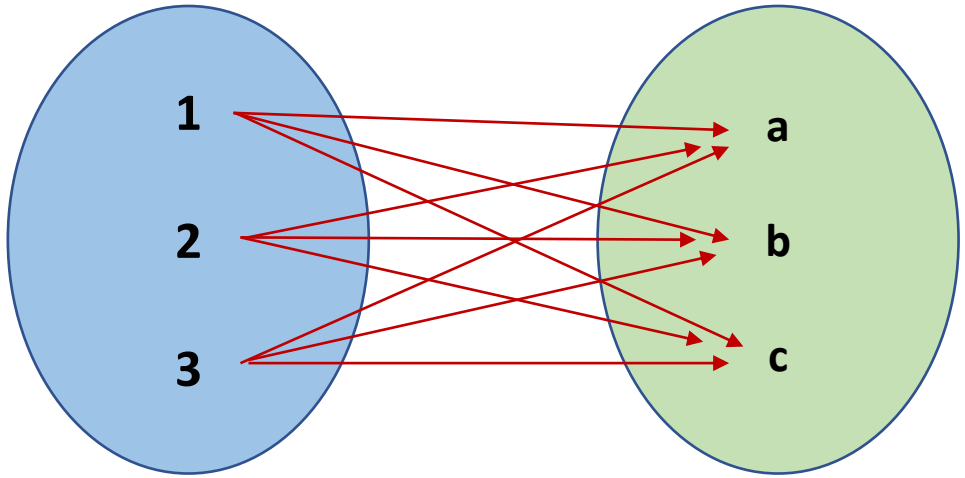
- JOIN clause is used to combine records from two or more tables.

- In MySQL, there are basically 3 different types of Joins:

  - **Cross** Join
  - **Inner** Join
  - **Outer** Join
    - Left Join
    - Right Join

Cross Join

- **CROSS JOIN** produces a result set which is the number of rows in the first table multiplied by the number of rows in the second table.



Cross Join (cont.)

- Syntax (Cross Join **without** WHERE clause)

```
SELECT  A1, A2, .. An
FROM    r1, r2;
```

- If no WHERE clause is used along with CROSS JOIN, then this kind of result is called **Cartesian Product**.

Cross Join (cont.)

- E.g.: If all the students have taken all the courses, so retrieve full name and course name of all of them.

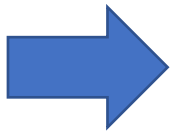
```
SELECT  F_name,L_name, Course
FROM    Student, Course;
```

Student

SID	F_name	L_name
1	Dara	Kawa
2	Zara	Nawzad
3	Ali	Omer

Course

CID	Course	Grade
IT215	Database	2
IT235	Multimedia	3



F_name	L_name	Course
Dara	Kawa	Database
Dara	Kawa	Multimedia
Zara	Nawzad	Database
Zara	Nawzad	Multimedia
Ali	Omer	Database
Ali	Omer	Multimedia

Cross Join (cont.)

- E.g.: If both tea and coffee are served to the visitors with each of the meals, so, retrieve the list.

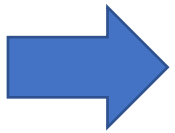
```
SELECT Meal_name, Beverage_name
FROM   Meal, Beverage;
```

Meal

MID	Meal_name
1	Fried Egg
2	Yoghurt
3	Omelet

Beverage

BID	Beverage_name
1	Tea
2	Coffee



Meal_name	Beverage_name
Fried Egg	Tea
Fried Egg	Coffee
Yoghurt	Tea
Yoghurt	Coffee
Omelet	Tea
Omelet	Coffee

Cross Join (cont.)

- Syntax (Cross Join **with** WHERE clause)

```
SELECT  A1, A2, .. An
FROM    r1, r2
WHERE r1.A1 = r2.A2;
```

- If WHERE clause is used with CROSS JOIN, it functions like an **INNER JOIN**.

Cross Join (cont.)

- E.g.: If only tea or coffee is served to the visitors with the meals, so, retrieve the list.

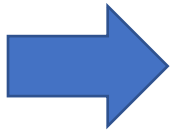
```
SELECT Meal_name, Beverage_name
FROM   Meal, Beverage
WHERE Meal.Beverage_ID = Beverage.BID;
```

Meal

MID	Meal_name	Beverage_ID
1	Fried Egg	1
2	Yoghurt	1
3	Omelet	2

Beverage

BID	Beverage_name
1	Tea
2	Coffee



Meal_name	Beverage_name
Fried Egg	Tea
Yoghurt	Tea
Omelet	Coffee

Inner Join

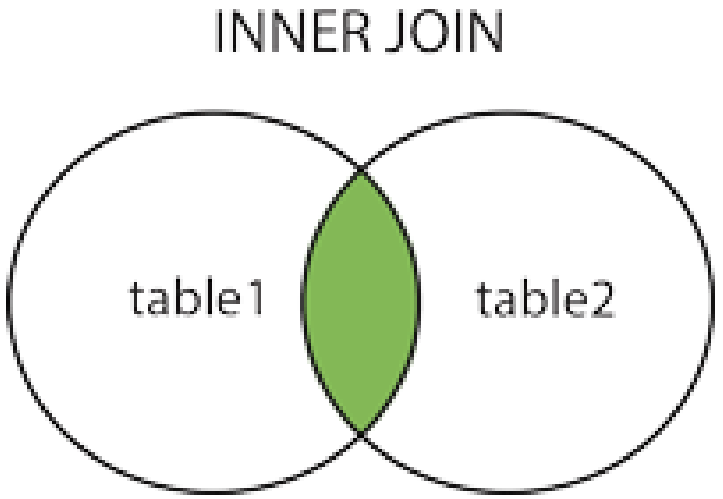
- Returns records that have matching values in both tables.

Syntax:

SELECT *A1,A2, .. An*

FROM *r1* **INNER JOIN** *r2*

ON *r1.Foreign_key=r2.Primary_key*



Inner Join (cont.)

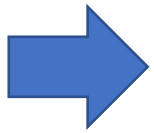
SELECT *F_name, Dept, No_of_std, Faculty*
FROM *Student* **INNER JOIN** *Department*
ON *Student.Dept = Department.Dept;*

SID	F_name	L_name	Dept
1	Dara	Azad	IT
2	Zara	Nawzad	Biology

Student

Dept	No_of_std	Faculty	Building
Biology	300	Education	Education
IT	400	Science	Main

Department



F_name	Dept	No_of_std	Faculty
Dara	IT	400	Science
Zara	Biology	300	Education

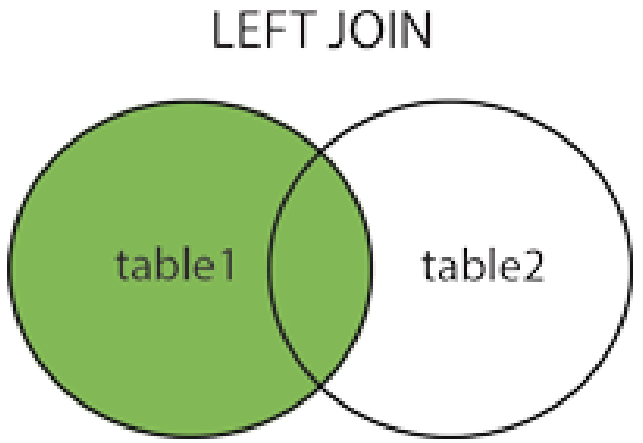
Left Join

- Returns all records from the left table (table1), and only the matched records from the right table (table2).

- The result is NULL from the right side, if there is no match.

Syntax:

SELECT *A1,A2, .. An*
FROM *r1* **LEFT JOIN** *r2*
ON *r1.Foreign_key=r2.Primary_key*



Left Join (cont.)

- For example, let's say we have a table of customers and a table of orders.
- We want to show a list of all customers, along with any orders they have placed, even if they haven't placed any orders yet.
- In this case, we need to use LEFT JOIN to ensure that all the customers are included in the result set, along with any orders they may have placed.

Left Join (cont.)

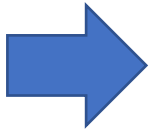
CID	F_name	L_name
1	Dara	Azad
2	Zara	Nawzad
3	Kawa	Omer

Customer

```
SELECT Cid, F_name, L_name, Datee
FROM customer LEFT JOIN order
ON customer.cid = order.customer_id;
```

OID	Customer_id	Datee
1	2	2022-4-12
2	3	2022-4-13
3	2	2022-4-14
4	2	2022-4-15

Order



CID	F_name	L_name	Datee
1	Dara	Azad	
2	Zara	Nawzad	2022-4-12
3	Zara	Nawzad	2022-4-14
4	Zara	Nawzad	2022-4-15
5	Kawa	Omer	2022-4-13

Left Join (cont.)

```
SELECT *
FROM customers LEFT JOIN orders
ON customers.customer_id = orders.customer_id;
```

```
SELECT *
FROM products LEFT JOIN categories
ON products.category_id = categories.category_id;
```

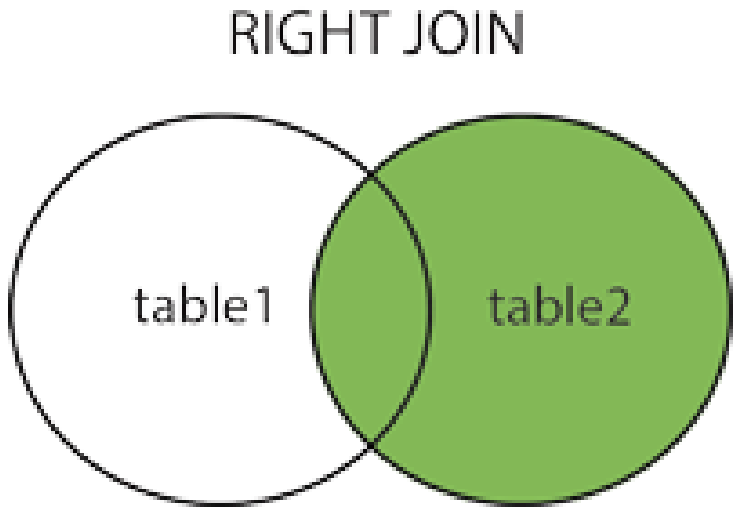
```
SELECT *
FROM departments LEFT JOIN employees
ON departments.department_id = employees.department_id;
```

Right Join

- Returns all records from the right table (table1), and only the matched records from the left table (table2).

Syntax:

SELECT *A1,A2, .. An*
FROM *r1* **RIGHT JOIN** *r2*
ON *r1.Foreign_key=r2.Primary_key*



Right Join (cont.)

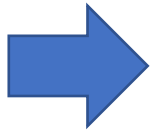
CID	F_name	L_name
1	Dara	Azad
2	Zara	Nawzad
3	Kawa	Omer

Customer

```
SELECT Cid, Datee, F_name, L_name
FROM customer RIGHT JOIN order
ON customer.cid = order.customer_id;
```

OID	Customer_id	Datee
1	2	2022-4-12
2	3	2022-4-13
3	2	2022-4-14
4	2	2022-4-15

Order



CID	datee	F_name	L_name
2	2022-4-12	Zara	Nawzad
3	2022-4-13	Kawa	Omer
2	2022-4-14	Zara	Nawzad
2	2022-4-15	Zara	Nawzad

Right Join (cont.)

CID	F_name	L_name
1	Dara	Azad
2	Zara	Nawzad
3	Kawa	Omer

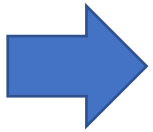
Customer

```
SELECT Cid, Datee, F_name, L_name
FROM customer RIGHT JOIN order
ON customer.cid = order.customer_id;
```

```
SELECT Cid, Datee, F_name, L_name
FROM order LEFT JOIN customer
ON customer.cid = order.customer_id;
```

OID	Customer_id	Datee
1	2	2022-4-12
2	3	2022-4-13
3	2	2022-4-14
4	2	2022-4-15

Order



CID	Datee	F_name	L_name
2	2022-4-12	Zara	Nawzad
3	2022-4-13	Kawa	Omer
2	2022-4-14	Zara	Nawzad
2	2022-4-15	Zara	Nawzad

Right Join (cont.)

```
SELECT *
FROM orders RIGHT JOIN customers
ON orders.customer_id = customers.customer_id;
```

```
SELECT *
FROM products RIGHT JOIN suppliers
ON products.supplier_id = suppliers.supplier_id;
```

```
SELECT *
FROM departments RIGHT JOIN employees
ON employees.department_id = departments.department_id;
```

Thank You