**1. Comparison Functions in MySQL**

These functions are used to compare values and return results based on those comparisons. Let’s dive into the common comparison functions:

**A. COALESCE Function**

* **What it does:** Returns the **first non-NULL value** from a list of arguments.
* **Syntax:**

COALESCE (value1, value2, ..., value\_n)

* **Why use it:** To handle NULL values and replace them with a default or fallback value.

**Example:**

Suppose we have an employees table with columns: name, salary, and commission.

| **name** | **salary** | **commission** |
| --- | --- | --- |
| Alice | 50000 | NULL |
| Bob | 45000 | 5000 |

**Query:** Calculate total compensation (salary + commission) and treat NULL commission as 0.

SELECT name, salary + COALESCE(commission, 0) AS total\_compensation

FROM employees;

**Result:**

| **name** | **total\_compensation** |
| --- | --- |
| Alice | 50000 |
| Bob | 50000 |

**Explanation:**

* For Alice, commission is NULL. COALESCE replaces it with 0.
* For Bob, commission is 5000. COALESCE uses the actual value.

**B. GREATEST Function**

* **What it does:** Returns the **largest value** from a list of arguments.
* **Syntax:**

GREATEST (value1, value2, ..., value\_n)

**Example:**

Suppose we have a products table with columns: id, price1, price2, and price3.

| **id** | **price1** | **price2** | **price3** |
| --- | --- | --- | --- |
| 101 | 10 | 15 | 12 |
| 102 | 8 | 20 | 5 |

**Query:** Find the highest price for each product.

SELECT id, GREATEST (price1, price2, price3) AS highest\_price

FROM products ;

**Result:**

| **id** | **highest\_price** |
| --- | --- |
| 101 | 15 |
| 102 | 20 |

**C. LEAST Function**

* **What it does:** Returns the **smallest value** from a list of arguments.
* **Syntax:**

sql

Copy code

LEAST(value1, value2, ..., value\_n)

**Example:**

Using the same products table, find the lowest price for each product.

**Query:**

sql

Copy code

SELECT id, LEAST(price1, price2, price3) AS lowest\_price

FROM products;

**Result:**

| **id** | **lowest\_price** |
| --- | --- |
| 101 | 10 |
| 102 | 5 |

**D. ISNULL Function**

* **What it does:** Checks if a value is NULL and returns 1 if true, otherwise 0.
* **Syntax:**

ISNULL(expression)

**Example:**

Check if the commission column in the employees table is NULL.

**Query:**

sql

Copy code

SELECT name, ISNULL(commission) AS commission\_is\_null

FROM employees;

**Result:**

| **name** | **commission\_is\_null** |
| --- | --- |
| Alice | 1 |
| Bob | 0 |

**2. Control Flow Functions in MySQL**

Control flow functions let you add **if-then-else** logic to SQL queries.

**A. CASE Function**

* **What it does:** Evaluates conditions and returns specific values based on the conditions.
* **Syntax:**

CASE

WHEN condition1 THEN result1

WHEN condition2 THEN result2

...

ELSE default\_result

END

**Example:**

Suppose we want to categorize employees’ salaries into grades:

| **name** | **salary** |
| --- | --- |
| Alice | 25000 |
| Bob | 50000 |

**Query:** Classify salaries into 'Low', 'Medium', or 'High'.

SELECT name,

CASE

WHEN salary < 30000 THEN 'Low'

WHEN salary BETWEEN 30000 AND 60000 THEN 'Medium'

ELSE 'High'

END AS salary\_grade

FROM employees;

**Result:**

| **name** | **salary\_grade** |
| --- | --- |
| Alice | Low |
| Bob | Medium |

**B. IF Function**

* **What it does:** Returns a value based on a single condition.
* **Syntax:**

IF(condition, value\_if\_true, value\_if\_false)

**Example:**

Check if employees’ salaries are above 45000.

**Query:**

SELECT name,

IF(salary > 45000, 'Above 45000', '45000 or Below') AS salary\_flag

FROM employees;

**Result:**

| **name** | **salary\_flag** |
| --- | --- |
| Alice | 45000 or Below |
| Bob | Above 45000 |

**C. IFNULL Function**

* **What it does:** Returns the first argument if it is **not NULL**, otherwise returns the second argument.
* **Syntax:**

IFNULL(expression1, expression2)

**Example:**

Calculate total compensation like with COALESCE.

**Query:**

SELECT name, salary + IFNULL(commission, 0) AS total\_compensation

FROM employees;

**D. NULLIF Function**

* **What it does:** Returns NULL if two expressions are equal; otherwise, returns the first expression.
* **Syntax:**

NULLIF (expression1, expression2)

**Example:**

Nullify bonus if it is 0.

| **name** | **bonus** |
| --- | --- |
| Alice | 1000 |
| Bob | 0 |

**Query:**

SELECT name, NULLIF(bonus, 0) AS adjusted\_bonus

FROM employees;

**Result:**

| **name** | **adjusted\_bonus** |
| --- | --- |
| Alice | 1000 |
| Bob | NULL |

**Additional Examples**

**Find employees without managers:**

SELECT name

FROM employees

WHERE manager\_id IS NULL;

**Group employees by designation and display in JSON format:**

SELECT designation,

JSON\_OBJECTAGG(employee\_id, name) AS employees

FROM employees

GROUP BY designation;