Control Flow Function

The control flow functions allow you to add if-then-else logic to SQL queries without using the procedural code. The following show the most commonly used MySQL control flows functions:

- **CASE** – return the corresponding result in THEN branch if the condition in the WHEN branch is satisfied, otherwise, return the result in the ELSE branch.

  Syntax:

```
CASE value
   WHEN [compare_value] THEN result
   [WHEN [compare_value] THEN result ...]
   [ELSE result]
END CASE
```

  Or

```
CASE
   WHEN expression1 THEN result
   [WHEN expression2 THEN result ...]
   [ELSE result]
END CASE
```

- **IF** – return a value based on a given condition.

  Syntax:

  IF(condition, value_if_true, value_if_false)

- **IFNULL**– return the first argument if it is not NULL , otherwise returns the second argument

  Syntax:

  IFNULL(*expression*, *alt_value*)

- **NULLIF**– return NULL if the first argument is equal to the second argument, otherwise, returns the first argument.

  Syntax:

  NULLIF(expr1, expr2)

1. Write a query to display first name and departments of all employees

```
SELECT first_name,
CASE dept_id
WHEN 20 THEN 'SALES'
WHEN 30 THEN 'EXE'
WHEN 40 THEN 'SHIP'
WHEN 50 THEN 'MKT'
WHEN 70 THEN 'ACC'
ELSE 'I_T'
END as DEPT
FROM employee;
```

Output:

first_name	DEPT
kelly	I_T
tom	ACC
mike	MKT
andy	I_T
anjel	SHIP
ram	SHIP
rohan	SALES

john	SALES

Here you have fetched the values by specifying the value, let's see the same example using expression

```
SELECT first_name,
CASE
WHEN dept_id = 20 THEN 'SALES'
WHEN dept_id IN (30,40) THEN 'E/S'
WHEN dept_id> 40 THEN 'M/A/I'
ELSE 'N/A'
END as DEPT
FROM employee;
```

Output:

first_name	**DEPT**
kelly	M/A/I
tom	M/A/I
mike	M/A/I
andy	M/A/I
anjel	E/S
ram	E/S
rohan	SALES
john	SALES

2. Write a query to display first name and salary and salary as high salary if

the salary is greater than 80000

```
SELECT first_name,
CASE
WHEN salary > 80000 THEN 'HIGH_SALARY'
ELSE 'LOW_SALARY'
END as SAL
FROM sql_notes.employee;
```

Let's see the same example IF()

```
SELECT
  first_name, IF(salary > 80000, 'HIGH_SALARY',
'LOW_SALARY') as SAL
FROM
  employee;
```

Output:

first_name	**SAL**
kelly	LOW_SALARY
tom	HIGH_SALARY
mike	HIGH_SALARY
andy	LOW_SALARY
anjel	LOW_SALARY
ram	LOW_SALARY
rohan	HIGH_SALARY
john	HIGH_SALARY

2. Write a query to display as new employee for those who hired in the year 2021 else old employee

```
SELECT
  first_name, IF(YEAR(hire_date) = '2021',
'NEW_EMPLOYEE', 'OLD_EMPLOYEE') as emp
FROM
  employee;
```

Output:

first_name	emp
kelly	NEW_EMPLOYEE
tom	OLD_EMPLOYEE
mike	NEW_EMPLOYEE
andy	NEW_EMPLOYEE
anjel	OLD_EMPLOYEE
ram	OLD_EMPLOYEE
rohan	NEW_EMPLOYEE
john	NEW_EMPLOYEE

3. Write a query to display null values in email id column as ‘no value entered’

```
SELECT
  IF(EMAIl IS NOT NULL, email, 'No Values Entered') as
email
```

```
FROM
  employee;
```

Output:

email
davis@gmail.com
tom@gmail.com
mike@gmail.com
andy@gmail.com
anj@gmail.com
ram@gmail.com
ro@gmail.com
No Values Entered

Now let's use IFNULL() to achieve the same

```
SELECT
  IFNULL(email, 'NO Values Entered') as email
FROM
  employee;
```

4. Query the email id's of all the employees with john's email id as null

```
SELECT NULLIF(email, 'ro@gmail.com') as email FROM
employee;
```

Output:

email

davis@gmail.com
tom@gmail.com
mike@gmail.com
andy@gmail.com
anj@gmail.com
ram@gmail.com
null
null

Here if you observe from the output now mail of ro@gmail.com is updated with null.