LET'S START WITH SQL :)

CASE AND IF IN SQL

- **CASE:** It allows you to perform conditional logic within a query. It can be used in both SELECT and UPDATE statements to evaluate conditions and return specific values based on those conditions.

QUERY:

```
CASE
  WHEN condition1 THEN result1 WHEN
condition2 THEN result2 ... ELSE resultN
END
```

LET’S START WITH SQL :)

CASE with Select statement

Q. Categorise the students on basis of their percentage to Top, Pass and fail in a new column category

QUERY:

```
SELECT sid, name, percentage,
 CASE
  WHEN percentage > 90 THEN ’Top’
  WHEN percentage BETWEEN 89 AND 34 THEN ‘Pass’
  ELSE ‘Fail’
 END AS category
FROM student;
```

LET'S START WITH SQL :)

CASE with Update statement

Q. Students have got some grace marks so update their grades. Where its A update to A+ and where its B update to A.

QUERY:

```
UPDATE student
SET grade = CASE
       WHEN grade= = ‘B’ THEN ‘A’
       WHEN grade = ‘A’ THEN ‘A+’
      END;
```

LET'S START WITH SQL :)

IF IN SQL

- **IF:** It is used to return one of two values depending on whether a condition is true or false. It is not supported in may DB but supported in MySQL

QUERY:

```
IF(condition, value_if_true, value_if_false)
```

LET’S START WITH SQL :)

IF with Select statement

Q. Categorise the students on basis of their percentage to Top, Pass and fail in a new column category

QUERY:

```
SELECT sid, name, percentage,
 IF(percentage > 90, ’Top’ , IF(percentage BETWEEN 89 AND 34, ‘Pass’, ‘Fail’)) AS category
FROM student;
```

LET’S START WITH SQL :)

IF with Update statement

Q. Swap all 'f' and 'm' values (i.e., change all 'f' values to 'm' and vice versa) with a single update statement and no intermediate temporary tables.

QUERY:

```
UPDATE employee
SET gender = if(gender = 'm', 'f', 'm')
```