Control flow functions

Control flow functions

Data overview

We will use the following **`Households_individuals`** table which contains certain information about the individuals in all households in Kenya collected during a household survey in 2020.

ID	Hh_ID	Sex	Age	Weight	Schooling	Current_ed	Marital_status	Spouse	Hhc_rship
3901	2401	Male	9	29.01	Yes	Class 3	Single	N/A	Grandchild
3821	2789	Male	22	67.00	Yes	Year 3	Single	N/A	Other relative
3961	2233	Female	35	59.00	Yes	Masters	Married	Yes	Partner
3741	2560	Female	14	45.22	Yes	Class 8	Single	N/A	Child
3661	2934	Male	69	77.00	No	N/A	Married	Yes	NULL
3921	2006	Female	16	45.99	Yes	Form 2	Single	N/A	Child

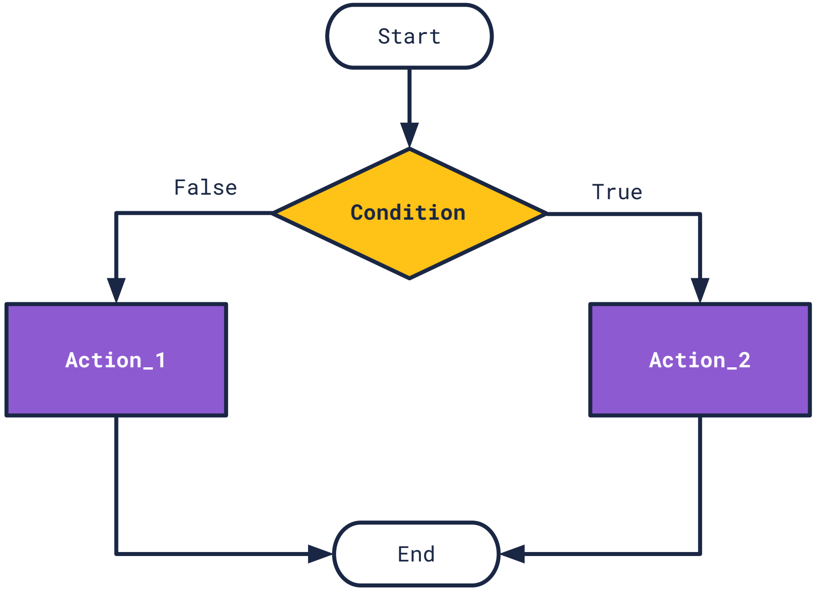
Control flow functions

> Control flow functions are used to **implement conditional logic** and **control the flow of execution** within SQL queries. They allow us to **perform different actions** or **return different values** based on specific **conditions**.

Commonly used control flow functions in SQL are the:

- IF function
- CASE statement

We can use control flow functions in various SQL statements such as the `SELECT`, `UPDATE`, `WHERE`, `ORDER BY`, and the `GROUP BY` clause.



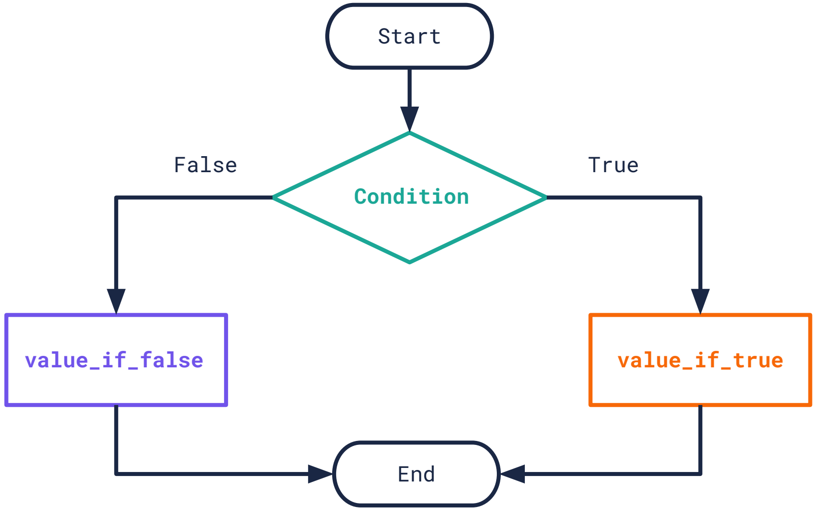
IF() function

The **`IF()`** function **evaluates a given condition** and returns a particular value if a condition is TRUE, or another value if a condition is FALSE. It can be used to conditionally retrieve or assign values in a table based on certain conditions.

Basic syntax

```
SELECT
     IF(condition,
        value_if_true,
        value_if_false
     )
FROM
     Table_name;
```

Flowchart

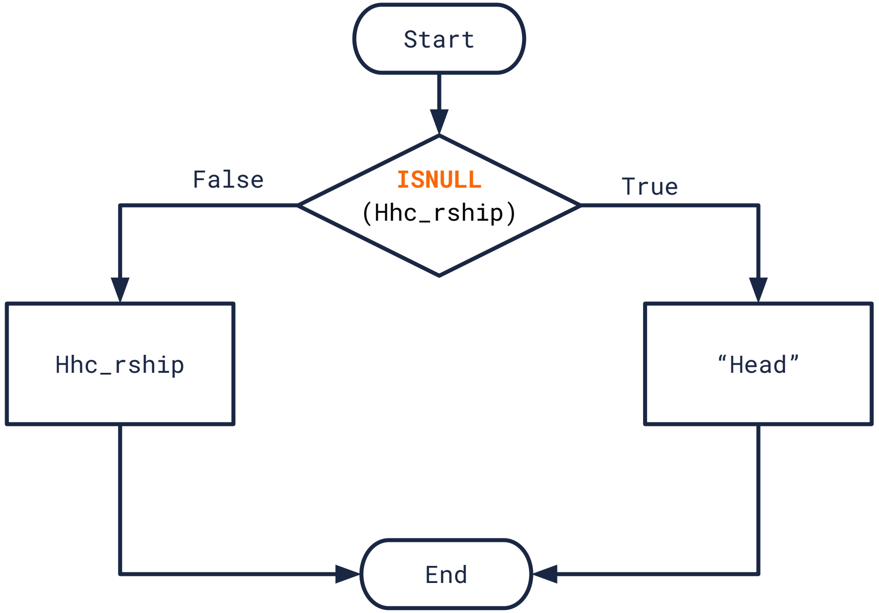


IF() function example

The **Hhc_rship** column indicates the relationship that an individual has with the household head. If an individual is the household head, their value has been set to NULL. Let's use the IF function below to **assign these NULL values the string value "Head"**.

```
…
IF(ISNULL(Hhc_rship),
   'Head',
   Hhc_rship
)
  AS New_hhc_rship
…;
```

We're saying when Hhc_rship is NULL, then we set the value to 'Head' but when it is not NULL, we set it to the original value.



Control flow functions

IF() function example

Query

```
SELECT
     ID,
     Hhc_rship,
     IF(ISNULL(Hhc_rship),
     'Head',
     Hhc_rship
     )
     AS New_hhc_rship
FROM
     Household_individuals;
```

Output

ID	Hhc_rship	New_hhc_rship
3901	Grandchild	Grandchild
3821	Other relative	Other relative
3961	Partner	Partner
3741	Child	Child
3661	NULL	Head
3921	Child	Child

CASE statement

The **CASE** statement allows us to apply **multiple conditions** that lead to **different sets of actions** within a SQL query.

There are two types of CASE statements:

1. The simple CASE statement

A list of values is compared to a given CASE expression.

2. The searched CASE statement

A list of conditions is evaluated to be either TRUE or FALSE.

For both types of CASE statements, once a match is found or a condition is true, the function stops reading and returns the **corresponding result.**

Optionally, an **ELSE clause** can be included, which specifies the value to be returned if **no match is found or no conditions are true**.

If there is **no ELSE clause** and no conditions are true, **NULL** is returned.

1. Simple CASE

In this form of the CASE function, a **CASE expression is specified**, and its **value compared to each of a set of values**. When a match is found, the corresponding result is returned. If none of the values matches the expression, the optional ELSE result is returned.

Basic syntax

```
SELECT
      CASE Case_Expression
           WHEN value_1 THEN result_1
           WHEN value_2 THEN result_2
           .
           .
           .
           WHEN value_N THEN result_N
           ELSE result
      END AS Alias_name
FROM
      Table_name;
```

Case expression: The expression to be compared to value_1, value_2,...value_N.

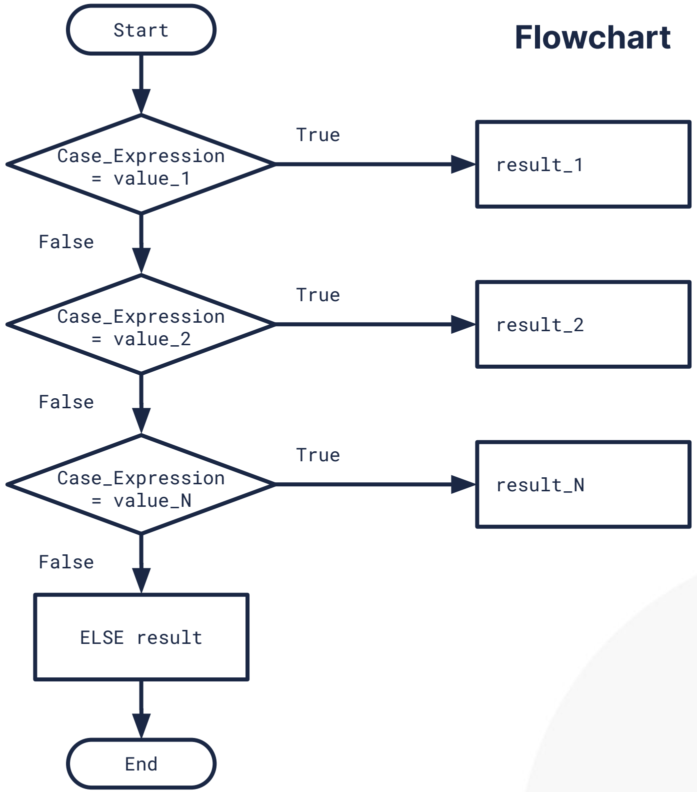
value_1, value_2, ...value_N: The values that are to be compared to the CASE expression in the same order as they are listed.

result_1, result_2, ...result_N: The result to be returned once the corresponding value matches the CASE expression.

ELSE result: The result to be returned if no values are matched.

1. Simple CASE

```
…
CASE Case_Expression
          WHEN value_1 THEN result_1
          WHEN value_2 THEN result_2
          .
          .
          .
          WHEN value_N THEN result_N
          ELSE result
END AS Alias_name
…;
```


Start
Case_Expression = value_1
True
result_1
False
Case_Expression = value_2
True
result_2
False
Case_Expression = value_N
True
result_N
False
ELSE result
End

Flowchart

2. Searched CASE

This form of the CASE function evaluates a **series of conditions** and returns a result based on the **first condition that evaluates to true**. If none of the conditions is true, the optional ELSE result is returned.

Basic syntax

```
SELECT
      CASE
            WHEN condition_1 THEN result_1
            WHEN condition_2 THEN result_2
            .
            .
            .
            WHEN condition_N THEN result_N
            ELSE result
      END AS Alias_name
FROM
      Table_name;
```

condition_1, condition_2, …condition_N: The conditions that are to be evaluated in the same order as they are listed.

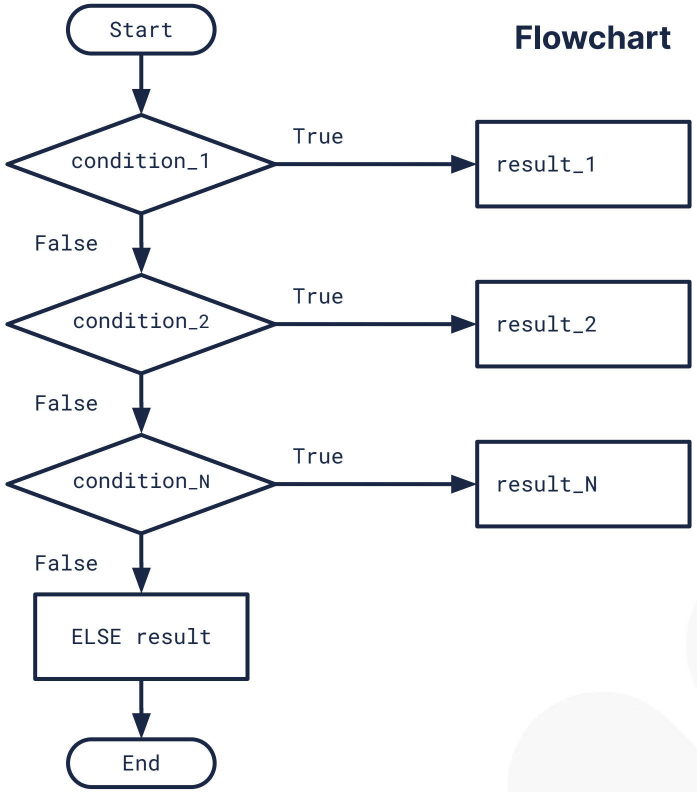
result_1, result_2, …result_N: The corresponding value to be returned for each condition if it evaluates to true.

ELSE result: The result to be returned if no condition is true.

Control flow functions

2. Searched CASE

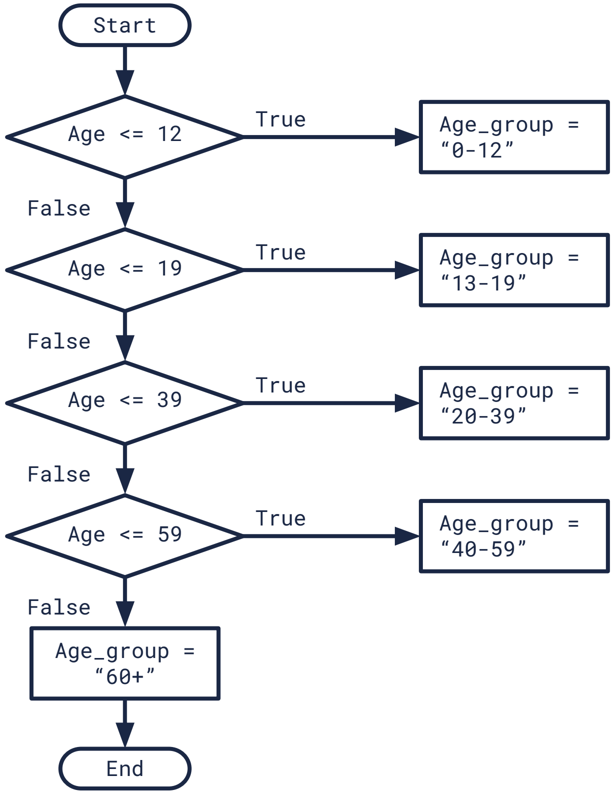
```
…
CASE
      WHEN condition_1 THEN result_1
      WHEN condition_2 THEN result_2
      .
      .
      .
      WHEN condition_N THEN result_N
      ELSE result
END AS Alias_name
…;
```



CASE statement example

Suppose we want to **categorise the individuals in our table into different age groups** representing the various stages of life. We can use the following CASE statement to perform the categorisation.

```
…
CASE
     WHEN Age <= 12 THEN “0-12”
     WHEN Age <= 19 THEN “13-19”
     WHEN Age <= 39 THEN “20-39”
     WHEN Age <= 59 THEN “40-59”
     ELSE ”60+”
END AS Age_group
…;
```



Control flow functions

CASE statement example

Query

```
SELECT
      ID,
      Age,
      CASE
            WHEN Age <= 12 THEN “0-12”
            WHEN Age <= 19 THEN “13-19”
            WHEN Age <= 39 THEN “20-39”
            WHEN Age <= 59 THEN “40-59”
            ELSE ”60+”
      END AS Age_group
FROM
      Household_individuals;
```

Output

ID	Age	Age_group
3901	9	0-12
3821	22	20-39
3961	35	20-39
3741	14	13-19
3661	69	60+
3921	16	13-19

Nested conditional statements

> This refers to the use of **one or more conditional statements**, like the `IF` and `CASE` control flow functions, **within another conditional statement.** This enables us to introduce multiple levels of conditions or to define the logic in a more granular way.

Nested conditional statements can exist in many different variations. We will look at examples of the following:

1. **Nested `IF` statement:**

The use of an `IF` function inside of another `IF` function.

Say we wish to **analyse the distribution of students at the various education levels** at that time. We can use the following nested IF condition to specify the education level for each individual that is currently enrolled in school.

2. **Nested `IF` and `CASE` statement:**

The use of an `IF` function inside of a `CASE` statement.

Suppose we want to **categorise all the individuals based on their age and gender** into the following groups: "Young female", "Young male", "Adult female", and "Adult male." Let's use the following nested `IF` and `CASE` statement to perform the categorisation.

Control flow functions

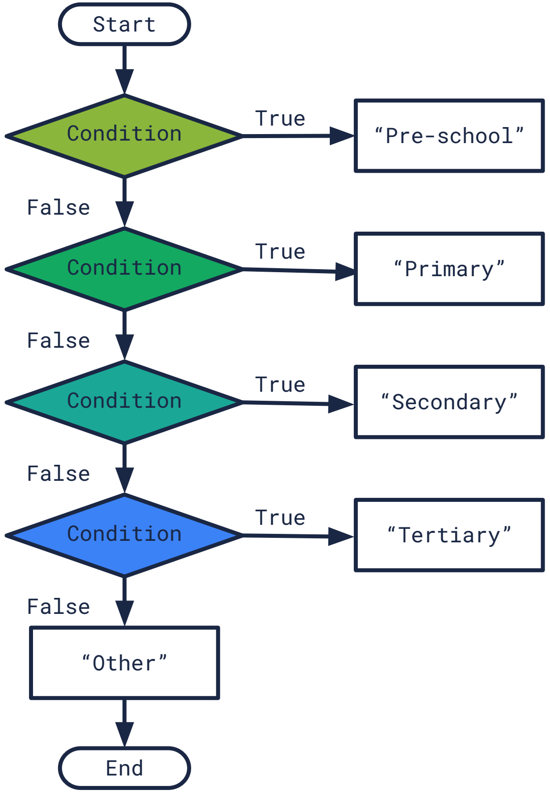
Nested IF example

```
…
IF(Current_ed IN (“PP1”, PP2”),
      “Pre-school”,
      IF(Current_ed IN (“Class 1”, “Class 2”, “Class 3”, “Class 4”, “Class 5”, “Class 6”,
                        “Class 7”, “Class 8”),
            “Primary”,
            IF(Current_ed IN (“Form 1”, “Form 2”, “Form 3”, “Form 4”),
                  “Secondary”,
                  IF(Current_ed IN (“First year”, “Second year”, “Third year”, “Fourth year”,
                                    “Fifth year”, “Sixth year”, “Masters”, “PHD”),
                        “Tertiary”,
                        “Other”
                  )
            )
      )
) AS Ed_level
…;
```

Control flow functions

Nested IF example

```
…
IF(Current_ed IN (“PP1”, PP2”),
      “Pre-school”,
      IF(Current_ed IN (“Class 1”, “Class 2”, “Class 3”, “Class 4”
      , “Class 5”, “Class 6”, “Class 7”, “Class 8”),
            “Primary”,
            IF(Current_ed IN (“Form 1”, “Form 2”, “Form 3”, “Form
            4”),
                  “Secondary”,
                  IF(Current_ed IN (“First year”, “Second year”,
                  “Third year”, “Fourth year”, “Fifth year”,
                  “Sixth year”, “Masters”, “PHD”),
                        “Tertiary”,
                        “Other”
                  )
            )
      )
) AS Ed_level
…;
```



Control flow functions

Nested IF example

Query

```
SELECT
        Schooling,
        Current_ed,
        IF(Current_ed IN (“PP1”, PP2”),
                “Pre-school”,
                IF(Current_ed IN (“Class 1”, “Class 2”, “Class 3”, “Class 4”, “Class 5”, “Class 6”, “Class 7”, “Class 8”),
                        “Primary”,
                        IF(Current_ed IN (“Form 1”, “Form 2”, “Form 3”, “Form 4”),
                                “Secondary”,
                                IF(Current_ed IN (“First year”, “Second year”, “Third year”, “Fourth year”, “Fifth year”, “Sixth year”,
                                “Masters”, “PHD”),
                                        “Tertiary”,
                                        “Other”
                                )
                        )
                )
        ) AS Ed_level

FROM
        Household_individuals
WHERE
        Schooling == “Yes”;
```

Control flow functions

Nested IF example

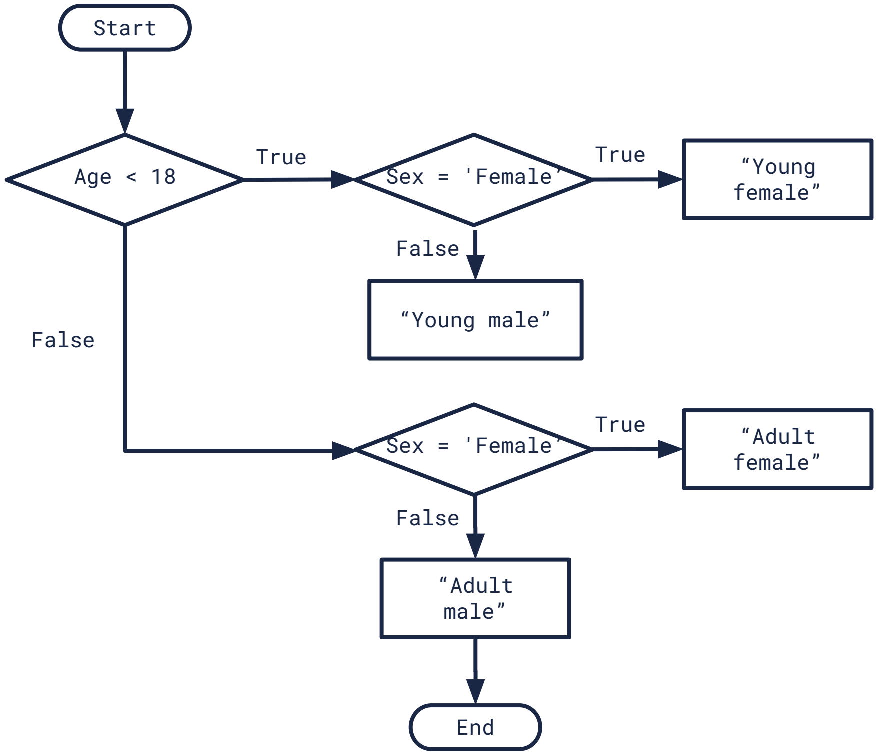
Output

Schooling	Current_ed	Ed_level
Yes	Class 3	Primary
Yes	Year 3	Tertiary
Yes	Masters	Tertiary
Yes	Class 8	Primary
Yes	Form 2	Secondary

Control flow functions

Nested IF and CASE statements example

```
…
CASE
     WHEN age < 18 THEN
          IF(Sex = “Female”,
               “Young female”,
               “Young male”)
     ELSE
          IF(Sex = “Female”,
               “Adult female”,
               “Adult male”)
END AS Age_category
…;
```



Control flow functions

Nested IF and CASE statements example

Query

```
SELECT
     Sex,
     Age,
     CASE
     WHEN age < 18 THEN
          IF(Sex = “Female”,
               “Young female”,
               “Young male”)
     ELSE
          IF(Sex = “Female”,
               “Adult female”,
               “Adult male”)
END AS Age_category
FROM
     Household_individuals;
```

Output

Sex	Age	Age_group
Male	9	Young male
Male	22	Adult male
Female	35	Adult female
Female	14	Young female
Male	69	Adult male
Female	16	Young female