Data manipulation in Snowflake

DATA MANIPULATION IN SNOWFLAKE

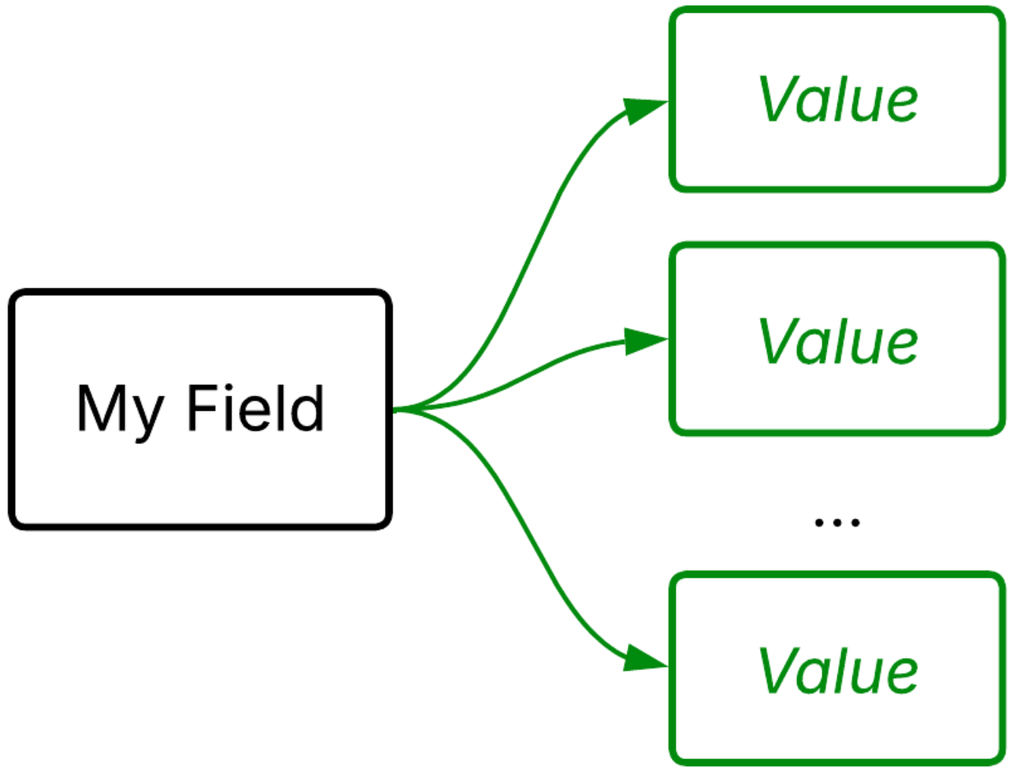
Jake Roach
Field Data Engineer

Snowflake and the modern data stack

Data manipulation in Snowflake

Categorize and analyze song metrics	**CASE Statements**
Break down top-performing products	Subqueries
Perform item-level order validation	Common Table Expressions

Conditional logic in Snowflake



> Evaluate the value of a field and *do something* based on that value.

`CASE` Statements

- Categorize/bucket data
- Transform and filter data
- Perform operations on that data

CASE statements

`CASE ... WHEN ... THEN ... END`

- Begin the evaluation with `CASE`
- Check a condition with `WHEN`
- Respond with `THEN`
- Complete the evaluation with `END`
- Alias the column

```
SELECT
    student_name,
    CASE
        WHEN grade_num = 12 THEN 'Senior'
        WHEN grade_num = 11 THEN 'Junior'
        ...
    END AS grade
FROM students;
```

```
|   student_name   |   grade   |
| ---------------- | --------- |
|     Viraj        |   Junior  |
|     Stephanie    |   Senior  |
```

Converting Grades to Grade Point Averages

```
SELECT
    student_id,
    course_name,
    CASE
        WHEN grade = 'A' THEN 4.0
        WHEN grade = 'B' THEN 3.0
        WHEN grade = 'C' THEN 2.0
        WHEN grade = 'D' THEN 1.0
        WHEN grade = 'F' THEN 0.0
    END AS gpa  -- Grade Point Average
FROM student_courses;
```

```
| student_id |  course_name  | gpa |
| ---------- | ------------- | --- |
|     001    |  Stats 101    | 4.0 |
|     001    |  Calculus     | 3.0 |
|     002    |  Biology      | 3.0 |
|     003    |  Finance      | 1.0 |
|     004    |  Engineering  | 4.0 |
|     004    |  Sales        | 2.0 |
|     004    |  Botany       | 4.0 |

...
```

Let's practice!

DATA MANIPULATION IN SNOWFLAKE

Conditional logic with CASE statements

DATA MANIPULATION IN SNOWFLAKE

Jake Roach
Field Data Engineer

datacamp

Using ELSE

> Sometimes, there might be additional values that don't fall into a `WHEN ... THEN` statement.

- Use `ELSE` to capture these additional scenarios
- Great way to catch edge cases

```
SELECT
    student,
    CASE
        WHEN grade_num = 12 THEN 'Senior'
        WHEN grade_num = 11 THEN 'Junior'
        ...
        ELSE 'Not in HS'  -- Catch others!
    END AS grade
FROM students;
```

```
|   student  |    grade   |
| ---------- | ---------- |
|  Viraj     |  Junior    |
|  Stephanie |  Senior    |
|  Lewis     |  Not in HS |
```

Evaluating a condition

Operator	Condition	Example
`=`	Value is equal to another value	`column_a = 'value_a'`
`IN (...)`	Value is in a list of values	`column_a IN ('value_a', 'value_b')`
`>` , `<`	Value is greater than or less than another value	`column_a > 0`
`>=` , `<=`	Counterparts to `>` and `<` , also includes equality	`column_a >= 0`
`BETWEEN`	Value is between two values, inclusive	`column_a BETWEEN 0 AND 10`

These comparison operators can be combined with `AND` , `OR` , or `NOT`

`CASE` statements in Snowflake are efficient due to being a column-oriented database

Categorizing temperatures

```
SELECT
    todays_date,
    temperature,
    CASE
        WHEN temperature BETWEEN 70 AND 90 THEN 'Ideal for Swimming'
        WHEN temperature >= 50 AND tempreature < 70 THEN 'Perfect for Sports'
        WHEN temperature > 32 AND temperature < 50 THEN 'Spring/Fall Temps'
        WHEN temperature > 0 AND temperature <= 32 THEN 'Winter Weather'
        ELSE 'Extreme Temperatures'
    END AS temperature_description
FROM weather;
```

Categorizing temperatures

todays_date	temperature	temperature_description
2025-01-01	17	Winter Weather
2025-02-11	34	Spring/Fall Temps
2025-05-28	68	Perfect for Sports

Combining conditional statements

```
SELECT
    todays_date,
    temperature,
    status,
    CASE
        WHEN temperature > 70 AND status NOT IN ('Rain', 'Wind') THEN 'Beach'
        WHEN temperature BETWEEN 45 AND 70 AND status = 'Sun' THEN 'Sports'
        WHEN temperature <= 32 OR status = 'Snow' THEN 'Skiing'
        ELSE 'Stay In'
    END AS activity
FROM weather;
```

> We can evaluate multiple columns in a single `CASE` statement

Combining conditional statements

todays_date	temperature	status	activity
2025-06-13	78	Overcast	Beach
2025-09-06	64	Sun	Sports
2025-10-30	41	Rain	Stay In
2025-12-19	27	Snow	Skiing

Let's practice!

DATA MANIPULATION IN SNOWFLAKE

Applying conditional logic in Snowflake

DATA MANIPULATION IN SNOWFLAKE

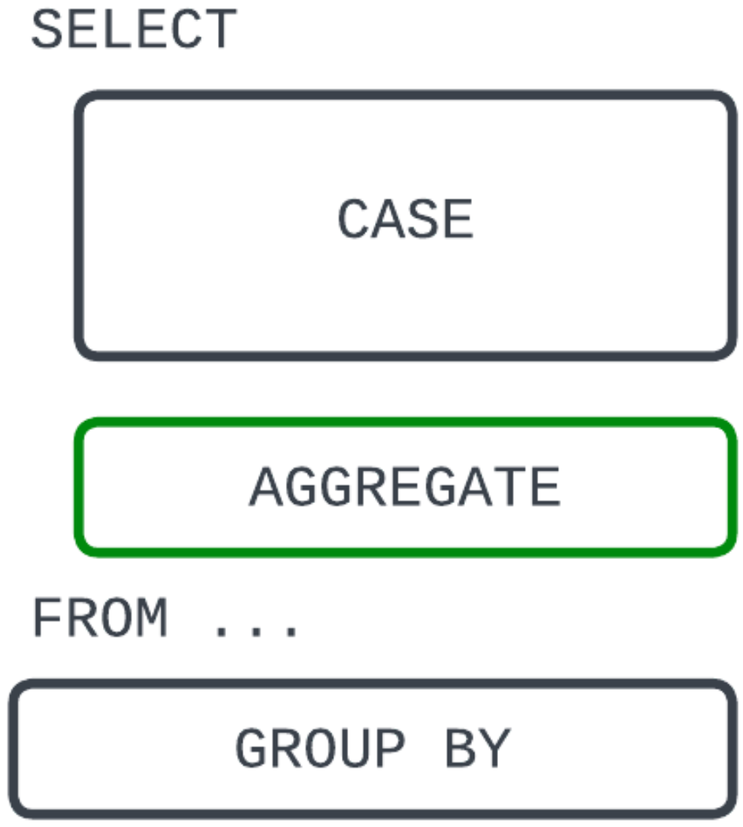
Jake Roach
Field Data Engineer

datacamp

Aggregation with CASE statements

Queries with a `CASE` statement can still use aggregation functions.

- `SUM()`
- `AVG()`
- `MIN()`
- `MAX()`
- ...
- `MODE()`



Just make sure to include a `GROUP BY` !

Aggregation with CASE statements

```
SELECT
    CASE
        WHEN month_num IN (12, 1, 2) THEN 'Winter'
        WHEN month_num IN (3, 4, 5) THEN 'Spring'
        WHEN month_num in (6, 7, 8) THEN 'Summer'
        ELSE 'Fall'
    END AS season,

    AVG(temperature) AS average_temperature,  -- Manipulate data with AVG()

    year_num
FROM weather
GROUP BY season, year_num;  -- Remember to group by non-aggregated fields
```

Aggregation with CASE statements

season	average_temperature	year_num
Summer	71.11	2024
Fall	57.38	2024
Spring	63.25	2025

Aggregating CASE'd fields

```
SELECT
    season,
    AVG(
        CASE
            WHEN season = 'Winter' THEN temperature - 30
            WHEN season IN ('Spring', 'Fall') THEN temperature - 60
            WHEN season = 'Summer' THEN temperature - 75
        END
    ) AS relation_to_average_temperature
FROM weather
GROUP BY season;
```

Aggregating CASE'd fields

season	relation_to_average_temperature
Winter	-4.278
Spring	-1.892
Summer	11.327
Fall	7.154

JOIN's

```
SELECT
    ...

    -- Define a CASE statement
    CASE ...

FROM <left_table>

-- JOIN new records to existing table
LEFT JOIN <right_table>
    ...
;
```

> `CASE` statements can be constructed using data joined from two different tables.

- Use columns from more than one table in a `CASE` statement
- `LEFT`, `RIGHT`, `INNER`, `OUTER`, etc.
- Combine with other tools, including aggregation functions

Let's check it out!

Determining college credit status

```
SELECT
    students.student_name,
    student_courses.course_name,

    -- Evaluate both the student's grade number and grade for the course
    CASE
        WHEN students.grade_num = 12 AND student_courses.grade > 90
            THEN 'College Credit Eligible'
        ELSE 'Too Early for College Credit'
    END AS college_credit_status

FROM student_courses
LEFT JOIN students ON student_courses.student_id = students.student_id;
```

Determining college credit status

student_name	course_name	college_credit_status
Haley	Calculus	College Credit Eligible
Andrew	Biology	Too Early for College Credit
Kirthi	Statistics	College Credit Eligible

Let's practice!

DATA MANIPULATION IN SNOWFLAKE