

## Contents 🔄 ⚙️

### ▼ Lesson Goals

What is a `CASE` Statement

Why Use `CASE` Statements

#### ▼ What Now?

Bucket Data

Zoom Out

Zoom In

Bucket Data

Zoom Out

Reference Mult

What is the `IF()` Function

Why Use the `IF()` Function

What Now?

#### ▼ Bonus!

Pivot Tables w

`COUNT` & `CASE`

# CASE Statement & IF() function

## Lesson Goals

- Understand how to use a `CASE` statement and why you might want to use it
  - bucket or bin data to Zoom out or reduce noise in your data by viewing it at a higher level
- Understand which logical operators you can use in your `CASE` statements
  - the same set of comparison operators you can use in a `WHERE` clause can be used in a `CASE` statement

Operator	What it Means
=	Equals
<>	Does NOT Equal
>	Greater Than
<	Less Than
>=	Greater Than Or Equal To
<=	Less Than Or Equal To
BETWEEN	A Range Between Two Values
LIKE	Matching a Pattern Like This
IN()	Equals One of These Values

- Understand how to use the `IF()` function and when you might opt for this option
- Understand the difference between the two above options

## What is a CASE Statement

A **CASE Statement** allows you to process a series of IF/THEN logical operators in a specific order. *They execute in the order they appear, so if a record satisfies more than one logical condition, the record will be assigned by the first true THEN statement.*

**Tip:** You might decide to use the ELSE condition as a catch all or error message to alert you to values you were not expecting in your data or faulty logic in your IF/THEN statements.

Check out more explanation and examples of `CASE` Statements [here \(https://ds-review-hub.github.io/sql\\_extras\)](https://ds-review-hub.github.io/sql_extras) in my review notebook.

## Contents 🔄 ⚙️

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
- ▼ What Now?
  - Bucket Data
  - Zoom Out
  - Zoom In
  - Bucket Data
  - Zoom Out
  - Reference Mult
- What is the IF() Function
  - Why Use the IF() Function
- What Now?
- ▼ Bonus!
  - Pivot Tables w COUNT & CASE

*-- If I'm only referencing one column and only testing for equality.*

```
SELECT
    CASE column_name
        WHEN condition_a THEN value_1
        WHEN condition_b THEN value_2
        ELSE value_3
    END AS new_column_name
FROM table_name;
```

*/\*  
CASE statement syntax. This allows me to reference different columns in my logic as well as use all of the conditional operators available to me in a WHERE Clause.  
\*/*

```
SELECT
    column_name,
    CASE
        WHEN column_name logic_1 THEN value1
        WHEN column_name logic_2 THEN value2
        WHEN column_name logic_3 THEN value3
        ELSE catch_all_value
    END AS new_column_name
FROM table_name;
```

---

## Why Use CASE Statements

## Contents ↻ ⚙

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
  - ▼ What Now?
    - Bucket Data
    - Zoom Out
    - Zoom In
    - Bucket Data
    - Zoom Out
    - Reference Mult
  - What is the IF() Function
  - Why Use the IF() Function
  - What Now?
  - ▼ Bonus!
    - Pivot Tables w COUNT & CASE

Fatih Kaya  
2021

# WHY DO WE USE CASE STATEMENTS?



WHEN I WANT TO ZOOM OUT OR  
REDUCE THE NOISE IN MY DATA



WHEN I WANT TO BUCKET OR BIN  
MY VALUES

## What Now?

Let's look at some examples.

```
-- Choose the chipotle database
USE chipotle;

-- Check out my orders table.
SELECT *
FROM orders;
```

## Contents 🔄 ⚙️

### ▼ Lesson Goals

What is a `CASE` Statement

Why Use `CASE` Statements

### ▼ What Now?

Bucket Data

Zoom Out

Zoom In

Bucket Data

Zoom Out

Reference Mult

What is the

`IF()` Function

Why Use the

`IF()` Function

What Now?

### ▼ Bonus!

Pivot Tables w

`COUNT` & `CASE`

## Bucket Data

- Use a `CASE` Statement to create categories called `item_type`.



*-- Use a `CASE` statement to create bins called `item_type` using `item_name`.*

```
SELECT
    item_name,
    CASE
        WHEN item_name LIKE '%chicken%' THEN 'Chicken Item'
        WHEN item_name LIKE '%veggie%' THEN 'Veggie Item'
        WHEN item_name LIKE '%beef%' THEN 'Beef Item'
        WHEN item_name LIKE '%barbacoa%'
            OR item_name LIKE '%carnitas%'
            OR item_name LIKE '%steak%' THEN 'Specialty Item'
        WHEN item_name LIKE '%chips%' THEN 'Side'
        ELSE 'Other'
    END AS item_type
FROM orders;
```

item_name	item_type
Chips and Fresh Tomato Salsa	Side
Izze	Other
Nantucket Nectar	Other
Chips and Tomatillo-Green Chili Salsa	Side
Chicken Bowl	Chicken Item

...

## Zoom Out

Add a `GROUP BY` Clause and use a `COUNT()` function to look at the popularity of item types.



## Contents 🔄 ⚙️

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
- ▼ What Now?
  - Bucket Data
  - Zoom Out
  - Zoom In
  - Bucket Data
  - Zoom Out
  - Reference Mult
- What is the IF() Function
- Why Use the IF() Function
- What Now?
- ▼ Bonus!
  - Pivot Tables w COUNT & CASE

-- How many different items do I have for each item type b  
in or category?

```
SELECT
  CASE
    WHEN item_name LIKE '%chicken%' THEN 'Chicken Item'
    WHEN item_name LIKE '%veggie%' THEN 'Veggie Item'
    WHEN item_name LIKE '%beef%' THEN 'Beef Item'
    WHEN item_name LIKE '%barbacoa%'
      OR item_name LIKE '%carnitas%'
      OR item_name LIKE '%steak%' THEN 'Specialty Item'
    WHEN item_name LIKE '%chips%' THEN 'Side'
    ELSE 'Other'
  END AS item_type,
  COUNT(*) count_of_records
FROM orders
GROUP BY item_type
ORDER BY count_of_records DESC;
```

item_type	count_of_records
Chicken Item	1560
Specialty Item	1086
Side	1084
Other	680
Veggie Item	212

### Zoom In

Add a sub-dimension to my GROUP BY Clause and a HAVING Clause to filter for Specialty Items. Adding a COUNT() Clause allows me to examine which speciality items are the most popular.



## Contents 🔄 ⚙️

### ▼ Lesson Goals

What is a CASE  
Statement

Why Use CASE  
Statements

### ▼ What Now?

Bucket Data

Zoom Out

Zoom In

Bucket Data

Zoom Out

Reference Mult

What is the

IF() Function

Why Use the

IF() Function

What Now?

### ▼ Bonus!

Pivot Tables w

COUNT & CASE

-- Filter my return set to Specialty Items item types only  
and see which item in this category is most popular.

```
SELECT
    item_name,
    CASE
        WHEN item_name LIKE '%chicken%' THEN 'Chicken Item'
        WHEN item_name LIKE '%veggie%' THEN 'Veggie Item'
        WHEN item_name LIKE '%beef%' THEN 'Beef Item'
        WHEN item_name LIKE '%barbacoa%'
            OR item_name LIKE '%carnitas%'
            OR item_name LIKE '%steak%' THEN 'Specialty Item'
        WHEN item_name LIKE '%chips%' THEN 'Side'
        ELSE 'Other'
    END AS item_type,
    COUNT(*) AS count_of_records
FROM orders
GROUP BY item_type, item_name
HAVING item_type = 'Specialty Item'
ORDER BY count_of_records DESC;
```

item_name	item_type	count_of_records
Steak Burrito	Specialty Item	368
Steak Bowl	Specialty Item	211
Barbacoa Burrito	Specialty Item	91
Carnitas Bowl	Specialty Item	68
Barbacoa Bowl	Specialty Item	66
Carnitas Burrito	Specialty Item	59
Steak Soft Tacos	Specialty Item	55
Carnitas Soft Tacos	Specialty Item	40
Steak Crispy Tacos	Specialty Item	35
Steak Salad Bowl	Specialty Item	29
Barbacoa Soft Tacos	Specialty Item	25
Barbacoa Crispy Tacos	Specialty Item	11
Barbacoa Salad Bowl	Specialty Item	10
Carnitas Crispy Tacos	Specialty Item	7
Carnitas Salad Bowl	Specialty Item	6
Steak Salad	Specialty Item	4
Carnitas Salad	Specialty Item	1

Contents ↻ ⚙

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
  - ▼ What Now?
    - Bucket Data
    - Zoom Out
    - Zoom In
    - Bucket Data
    - Zoom Out
    - Reference Mult
  - What is the IF() Function
  - Why Use the IF() Function
  - What Now?
  - ▼ Bonus!
    - Pivot Tables w COUNT & CASE

Bucket Data

Create a Categorical Variable from Numeric Variable



```
-- Create buckets for quantity to create a new categorical variable.
SELECT
    item_name,
    CASE
        WHEN quantity = 1 THEN 'single_item'
        WHEN quantity BETWEEN 2 AND 5 THEN 'family_and_friends'
    ends'
        WHEN quantity BETWEEN 6 AND 9 THEN 'small_gathering'
        WHEN quantity > 9 THEN 'party'
        ELSE 'other'
    END AS quant_cats
FROM orders;
```

item_name	quant_cats
Chips and Fresh Tomato Salsa	single_item
Izze	single_item
Nantucket Nectar	single_item
Chips and Tomatillo-Green Chili Salsa	single_item
Chicken Bowl	family_and_friends

...

Zoom Out

Add a GROUP BY Clause and use the COUNT() function to look at most common quantity category.



## Contents 🔄 ⚙️

- ▼ Lesson Goals
  - What is a `CASE` Statement
  - Why Use `CASE` Statements
- ▼ What Now?
  - Bucket Data
  - Zoom Out
  - Zoom In
  - Bucket Data
  - Zoom Out
  - Reference Mult
- What is the `IF()` Function
- Why Use the `IF()` Function
- What Now?
- ▼ Bonus!
  - Pivot Tables w
  - `COUNT` & `CASE`

*-- Add a `GROUP BY` Clause to Zoom Out and take a look at my new categorical variables `quant_cats`*

```
SELECT
  COUNT(*) AS count_of_records,
  CASE
    WHEN quantity = 1 THEN 'single_item'
    WHEN quantity BETWEEN 2 AND 5 THEN 'family_and_fri
ends'
    WHEN quantity BETWEEN 6 AND 9 THEN 'small_gatherin
g'
    WHEN quantity > 9 THEN 'party'
    ELSE 'other'
  END AS quant_cats
FROM orders
GROUP BY quant_cats
ORDER BY count_of_records DESC;
```

count_of_records	quant_cats
4355	single_item
263	family_and_friends
2	party
2	small_gathering

## Reference Multiple Columns

Let's look at an example that references different columns in our `CASE` statement logic.

```
-- Use mall_customers database.
USE mall_customers;

-- Check out the customers table.
SELECT *
FROM customers;
```





Contents ⌕ ⚙

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
  - ▼ What Now?
    - Bucket Data
    - Zoom Out
    - Zoom In
    - Bucket Data
    - Zoom Out
    - Reference Mult
  - What is the IF() Function
  - Why Use the IF() Function
  - What Now?
  - ▼ Bonus!
    - Pivot Tables w COUNT & CASE

```
-- Reference more than one column in CASE Statement logic.
SELECT
  gender,
  age,
  CASE
    WHEN gender = 'Male' AND age < 20 THEN 'Teen Male'
    WHEN gender = 'Male' AND age < 30 THEN 'Twenties Male'
    WHEN gender = 'Male' AND age < 40 THEN 'Thirties Male'
    WHEN gender = 'Male' AND age < 50 THEN 'Forties Male'
    WHEN gender = 'Male' AND age < 60 THEN 'Fifties Male'
    WHEN gender = 'Male' AND age < 70 THEN 'Sixties Male'
    WHEN gender = 'Male' AND age >= 70 THEN 'Older Male'
    WHEN gender = 'Female' AND age < 20 THEN 'Teen Female'
    WHEN gender = 'Female' AND age < 30 THEN 'Twenties Female'
    WHEN gender = 'Female' AND age < 40 THEN 'Thirties Female'
    WHEN gender = 'Female' AND age < 50 THEN 'Forties Female'
    WHEN gender = 'Female' AND age < 60 THEN 'Fifties Female'
    WHEN gender = 'Female' AND age < 70 THEN 'Sixties Female'
    WHEN gender = 'Female' AND age >= 70 THEN 'Older Female'
    ELSE 'Other'
  END AS gen_age_cat
FROM customers;
```

gender	age	gen_age_cat
Male	19	Teen Male
Male	21	Twenties Male
Female	20	Twenties Female
Female	23	Twenties Female
Female	31	Thirties Female

...



# Contents 🔄 ⚙️

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
- ▼ What Now?
  - Bucket Data
  - Zoom Out
  - Zoom In
  - Bucket Data
  - Zoom Out
  - Reference Mult
- What is the IF() Function
- Why Use the IF() Function
- What Now?
- ▼ Bonus!
  - Pivot Tables w COUNT & CASE

-- Zoom Out by adding a Group By Clause and a COUNT() function.

```
SELECT
CASE
    WHEN gender = 'Male' AND age < 20 THEN 'Teen Male'
    WHEN gender = 'Male' AND age < 30 THEN 'Twenties Male'
    WHEN gender = 'Male' AND age < 40 THEN 'Thirties Male'
    WHEN gender = 'Male' AND age < 50 THEN 'Forties Male'
    WHEN gender = 'Male' AND age < 60 THEN 'Fifties Male'
    WHEN gender = 'Male' AND age < 70 THEN 'Sixties Male'
    WHEN gender = 'Male' AND age >= 70 THEN 'Older Male'
    WHEN gender = 'Female' AND age < 20 THEN 'Teen Female'
    WHEN gender = 'Female' AND age < 30 THEN 'Twenties Female'
    WHEN gender = 'Female' AND age < 40 THEN 'Thirties Female'
    WHEN gender = 'Female' AND age < 50 THEN 'Forties Female'
    WHEN gender = 'Female' AND age < 60 THEN 'Fifties Female'
    WHEN gender = 'Female' AND age < 70 THEN 'Sixties Female'
    WHEN gender = 'Female' AND age >= 70 THEN 'Older Female'
    ELSE 'Other'
END AS gen_age_cat,
COUNT(*) AS count_of_customers
FROM customers
GROUP BY gen_age_cat
ORDER BY count_of_customers DESC;
```

gen_age_cat	count_of_customers
Thirties Female	37
Twenties Female	26
Thirties Male	24
Forties Female	24
Twenties Male	17
Forties Male	15
Fifties Female	14
Fifties Male	11
Sixties Male	10
Teen Male	9
Sixties Female	8

Contents ⌂ ⚙

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
  - ▼ What Now?
    - Bucket Data
    - Zoom Out
    - Zoom In
    - Bucket Data
    - Zoom Out
    - Reference Mult
  - What is the IF() Function
  - Why Use the IF() Function
  - What Now?
  - ▼ Bonus!
    - Pivot Tables w COUNT & CASE

gen_age_cat	count_of_customers
Teen Female	3
Older Male	2

What is the IF() Function

Faith Kane  
2021

**WHERE DOES THE IF FUNCTION FIT INTO OUR BIG 6 ELEMENTS?**

Necessary

Optional

1

SELECT

COLUMN\_NAME(S),  
IF (CONDITION, VAL\_IF\_TRUE, VAL\_IF\_FALSE) AS NEW\_COLUMN

2

FROM

TABLE\_NAME

3

WHERE

LOGICAL\_CONDITION(S)

4

GROUP BY

COLUMN\_NAME(S)

5

HAVING

LOGICAL\_CONDITION(S)

6

ORDER BY

COLUMN\_NAME(S)

Why Use the IF() Function

## Contents ↻ ⚙

### ▼ Lesson Goals

What is a CASE  
Statement

Why Use CASE  
Statements

### ▼ What Now?

Bucket Data

Zoom Out

Zoom In

Bucket Data

Zoom Out

Reference Mult

What is the

IF() Function

Why Use the

IF() Function

What Now?

### ▼ Bonus!

Pivot Tables w

COUNT & CASE

Faith Kane  
2021

# WHY DO WE USE THE IF FUNCTION?



1

WHEN I AM EVALUATING A  
CONDITION TO TRUE OR FALSE

2

IF I WANT A COLUMN OF BOOLEAN  
VALUES (DUMMY VARIABLE)

## What Now?

Let's look at some examples.

## Contents 🔄 ⚙️

### ▼ Lesson Goals

What is a `CASE` Statement

Why Use `CASE` Statements

▼ What Now?

Bucket Data

Zoom Out

Zoom In

Bucket Data

Zoom Out

Reference Mult

What is the

`IF()` Function

Why Use the

`IF()` Function

What Now?

### ▼ Bonus!

Pivot Tables w

`COUNT` & `CASE`

*-- Use the mall\_customers database.*

```
USE mall_customers;
```

*-- Check out the customers table.*

```
SELECT *  
FROM customers;
```

*-- Use an IF Function to create a dummy variable for gender.*

```
SELECT  
    gender,  
    IF(gender = 'Female', True, False) AS is_female  
FROM customers;
```

gender	is_female
--------	-----------

Male	0
------	---

Male	0
------	---

Female	1
--------	---

Female	1
--------	---

...

*-- I can create this new boolean column in another simple way, just evaluate the equality statement to True or False.*

```
SELECT  
    gender,  
    gender = 'Female' AS is_female  
FROM customers;
```

gender	is_female
--------	-----------

Male	0
------	---

Male	0
------	---

Female	1
--------	---

Female	1
--------	---

...

## Bonus!

### Pivot Tables with `COUNT` & `CASE`

I can pass a `CASE` statement to the `COUNT` function to create a pivot table or frequency table.

# Contents 🔄 ⚙️

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
- ▼ What Now?
  - Bucket Data
  - Zoom Out
  - Zoom In
  - Bucket Data
  - Zoom Out
  - Reference Mult
- What is the IF() Function
- Why Use the IF() Function
- What Now?
- ▼ Bonus!
  - Pivot Tables w COUNT & CASE

-- Create a pivot table using COUNT and CASE to view the number of titles by department. This is a count of employee s who have ever held each title by department.

```
SELECT
    dept_name,
    COUNT(CASE WHEN title = 'Senior Engineer' THEN title ELSE NULL END) AS 'Senior Engineer',
    COUNT(CASE WHEN title = 'Staff' THEN title ELSE NULL END) AS 'Staff',
    COUNT(CASE WHEN title = 'Engineer' THEN title ELSE NULL END) AS 'Engineer',
    COUNT(CASE WHEN title = 'Senior Staff' THEN title ELSE NULL END) AS 'Senior Staff',
    COUNT(CASE WHEN title = 'Assistant Engineer' THEN title ELSE NULL END) AS 'Assistant Engineer',
    COUNT(CASE WHEN title = 'Technique Leader' THEN title ELSE NULL END) AS 'Technique Leader',
    COUNT(CASE WHEN title = 'Manager' THEN title ELSE NULL END) AS 'Manager'
FROM departments
JOIN dept_emp USING(dept_no)
JOIN titles USING(emp_no)
GROUP BY dept_name
ORDER BY dept_name;
```

dept_name	Senior Engineer	Staff	Engineer	Senior Staff	Assistant Engineer	Technique Leader	Manager
Customer Service	2027	16150	2362	13925	298	309	4
Development	49326	1424	58135	1247	7769	7683	2
Finance	0	13929	0	12139	0	0	2
Human Resources	0	14342	0	12274	0	0	2
Marketing	0	16196	0	13940	0	0	2
Production	42205	1478	49649	1270	6445	6557	4
Quality Management	11864	0	13852	0	1831	1795	4
Research	2570	13495	2986	11637	378	393	2
Sales	0	41808	0	36191	0	0	2

Contents

- ▼ Lesson Goals
  - What is a CASE Statement
  - Why Use CASE Statements
- ▼ What Now?
  - Bucket Data
  - Zoom Out
  - Zoom In
  - Bucket Data
  - Zoom Out
  - Reference Mult
- What is the IF() Function
- Why Use the IF() Function
- What Now?
- ▼ Bonus!
  - Pivot Tables w COUNT & CASE

-- In this query, I filter for current employees who currently hold each title.

```
SELECT
    dept_name,
    COUNT(CASE WHEN title = 'Senior Engineer' THEN title ELSE NULL END) AS 'Senior Engineer',
    COUNT(CASE WHEN title = 'Staff' THEN title ELSE NULL END) AS 'Staff',
    COUNT(CASE WHEN title = 'Engineer' THEN title ELSE NULL END) AS 'Engineer',
    COUNT(CASE WHEN title = 'Senior Staff' THEN title ELSE NULL END) AS 'Senior Staff',
    COUNT(CASE WHEN title = 'Assistant Engineer' THEN title ELSE NULL END) AS 'Assistant Engineer',
    COUNT(CASE WHEN title = 'Technique Leader' THEN title ELSE NULL END) AS 'Technique Leader',
    COUNT(CASE WHEN title = 'Manager' THEN title ELSE NULL END) AS 'Manager'
FROM departments
JOIN dept_emp ON departments.dept_no = dept_emp.dept_no
AND dept_emp.to_date > CURDATE()
JOIN titles ON dept_emp.emp_no = titles.emp_no
AND titles.to_date > CURDATE()
GROUP BY dept_name
ORDER BY dept_name;
```

dept_name	Senior Engineer	Staff	Engineer	Senior Staff	Assistant Engineer	Technique Leader	Manager
Customer Service	1790	3574	627	11268	68	241	1
Development	38816	315	14040	1085	1652	5477	1
Finance	0	2891	0	9545	0	0	1
Human Resources	0	3073	0	9824	0	0	1
Marketing	0	3551	0	11290	0	0	1
Production	33625	349	12081	1123	1402	4723	1
Quality Management	9458	0	3405	0	389	1293	1
Research	2250	2870	830	9092	77	321	1
Sales	0	8903	0	28797	0	0	1

In [ ]: