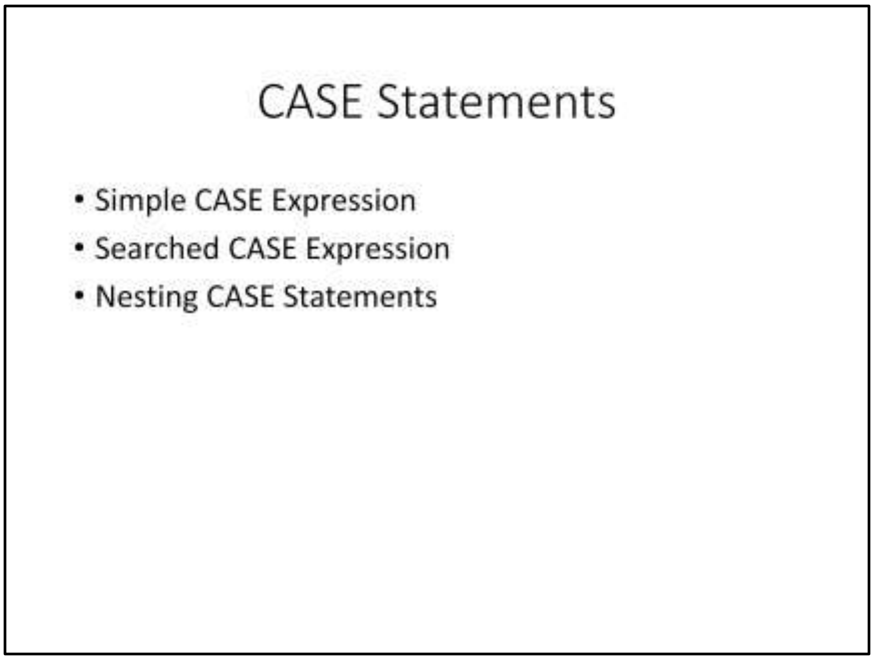
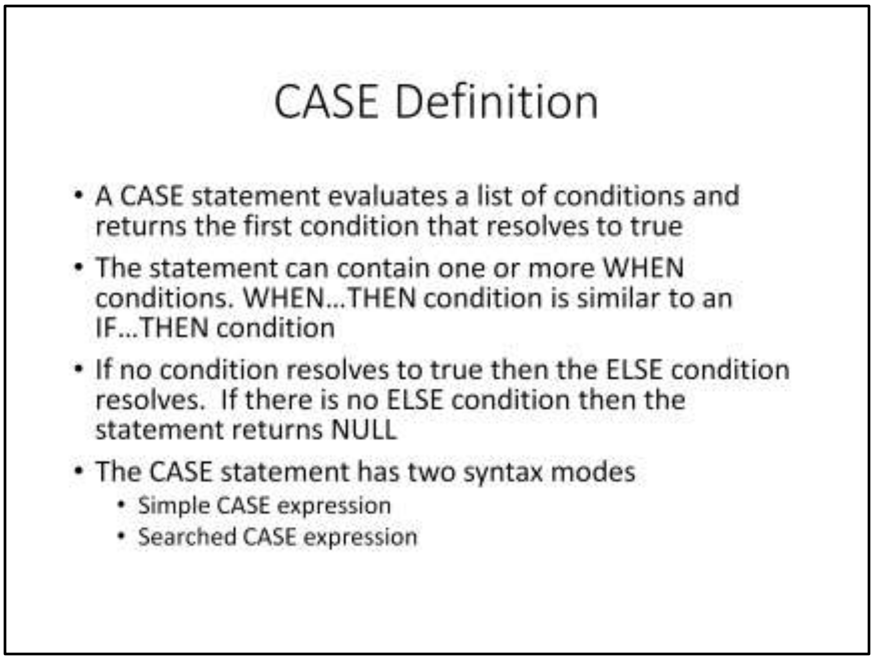


CASE Statements

- Simple CASE Expression
- Searched CASE Expression
- Nesting CASE Statements

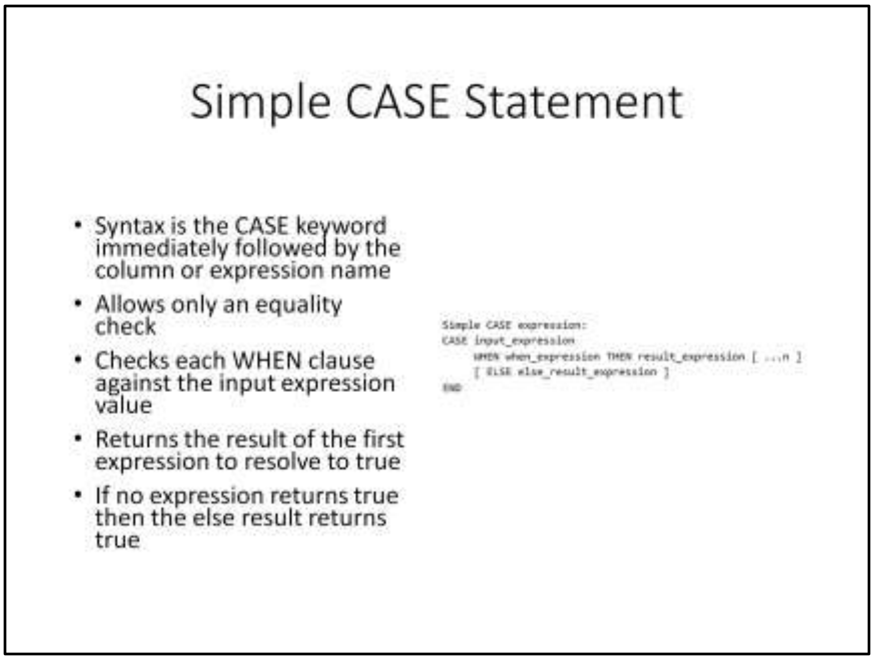
This presentation will cover CASE statements. I will talk about the two types of CASE statements as well as nesting case statements.



CASE Definition

- A CASE statement evaluates a list of conditions and returns the first condition that resolves to true
- The statement can contain one or more WHEN conditions. WHEN…THEN condition is similar to an IF…THEN condition
- If no condition resolves to true then the ELSE condition resolves. If there is no ELSE condition then the statement returns NULL
- The CASE statement has two syntax modes
  - Simple CASE expression
  - Searched CASE expression

A CASE statement is often, but not always used to create a derived column in a select statement. A CASE statement evaluates one or more conditions. It evaluates these conditions in order, and the first one to evaluate to true is the one that has its result set returned. Conditions are a series of WHEN THEN statements which perform similarly to an IF THEN statement. CASE statements also allow for an ELSE condition which is the default if none of the WHEN THEN conditions resolve to true. If no ELSE condition is provided then the default will be a NULL value. CASE statements come in two versions, the Simple CASE and the Searched CASE.

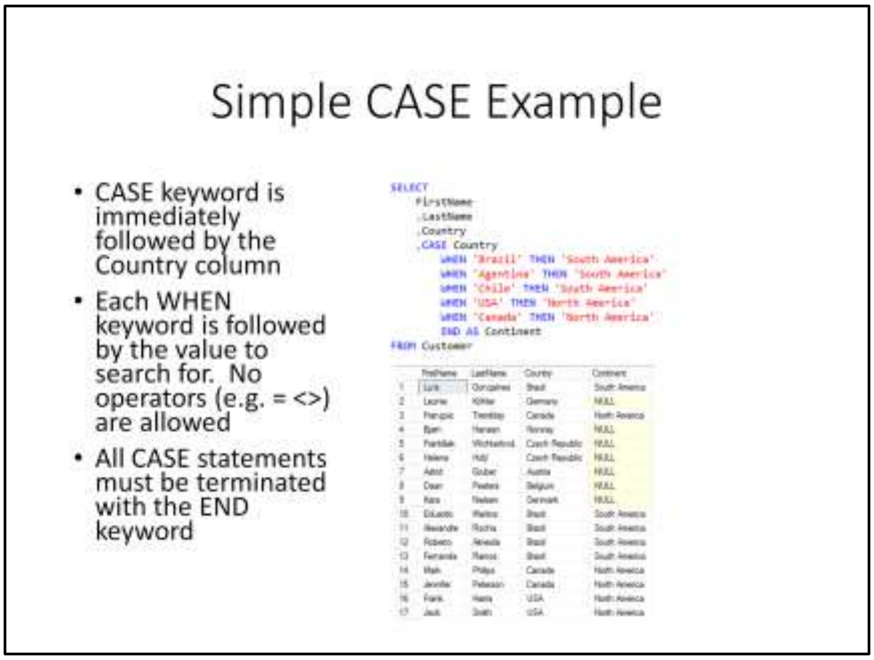


Simple CASE Statement

- Syntax is the CASE keyword immediately followed by the column or expression name
- Allows only an equality check
- Checks each WHEN clause against the input expression value
- Returns the result of the first expression to resolve to true
- If no expression returns true then the else result returns true

```
Simple CASE expression:
CASE input_expression
     WHEN when_expression THEN result_expression [ ...n ]
     [ ELSE else_result_expression ]
END
```

The Simple CASE statement checks the values of a single column. If the WHEN value is found then the statement returns the content of the THEN clause. Only equality checks are allowed in a Simple CASE statement.



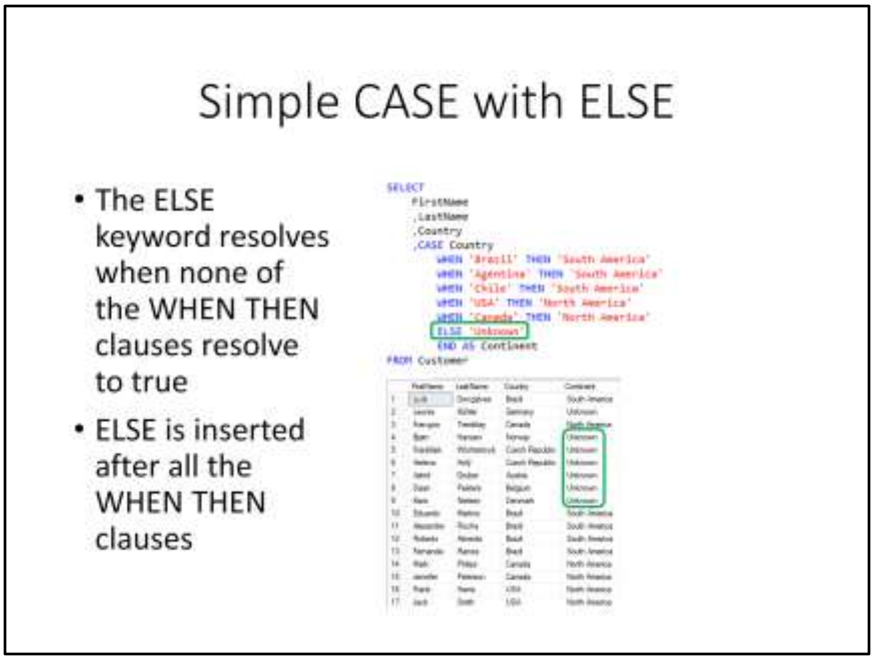
Simple CASE Example

- CASE keyword is immediately followed by the Country column
- Each WHEN keyword is followed by the value to search for. No operators (e.g. = <>) are allowed
- All CASE statements must be terminated with the END keyword

```
SELECT
    FirstName
   ,LastName
   ,Country
   ,CASE Country
        WHEN 'Brazil' THEN 'South America'
        WHEN 'Agentina' THEN 'South America'
        WHEN 'Chile' THEN 'South America'
        WHEN 'USA' THEN 'North America'
        WHEN 'Canada' THEN 'North America'
        END AS Continent
FROM Customer
```

	FirstName	LastName	Country	Continent
1	Luís	Gonçalves	Brazil	South America
2	Leonie	Köhler	Germany	NULL
3	François	Tremblay	Canada	North America
4	Bjørn	Hansen	Norway	NULL
5	František	Wichterlová	Czech Republic	NULL
6	Helena	Holý	Czech Republic	NULL
7	Astrid	Gruber	Austria	NULL
8	Daan	Peeters	Belgium	NULL
9	Kara	Nielsen	Denmark	NULL
10	Eduardo	Martins	Brazil	South America
11	Alexandre	Rocha	Brazil	South America
12	Roberto	Almeida	Brazil	South America
13	Fernanda	Ramos	Brazil	South America
14	Mark	Philips	Canada	North America
15	Jennifer	Peterson	Canada	North America
16	Frank	Harris	USA	North America
17	Jack	Smith	USA	North America

In the example I am searching the Country column for specific country names. If the country name is found then the continent name is returned. There is no ELSE clause in this CASE statement so any Country column names that aren't identified in the WHEN clauses will return as NULL.



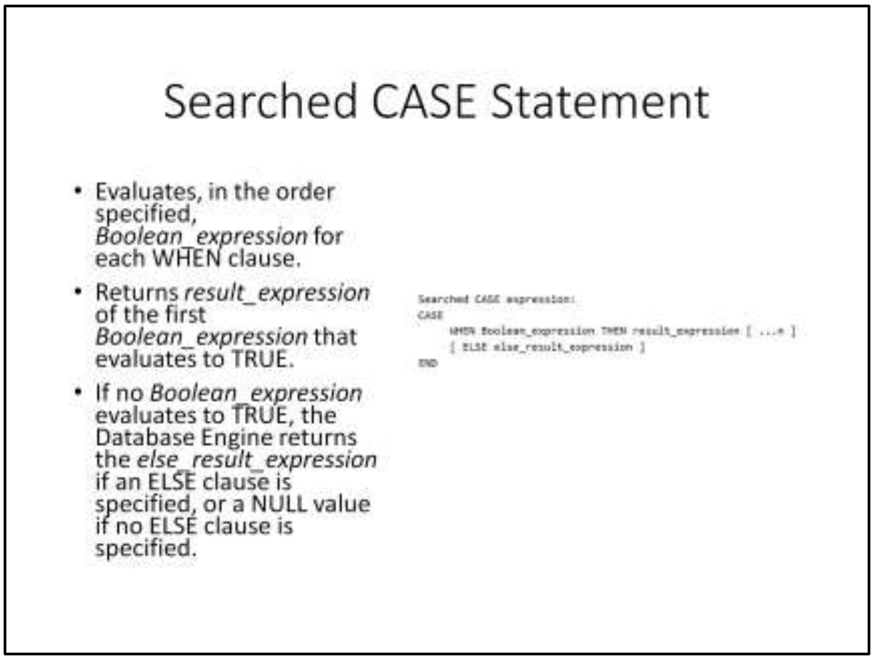
Simple CASE with ELSE

- The ELSE keyword resolves when none of the WHEN THEN clauses resolve to true
- ELSE is inserted after all the WHEN THEN clauses

```
SELECT
    FirstName
    ,LastName
    ,Country
    ,CASE Country
        WHEN 'Brazil' THEN 'South America'
        WHEN 'Agentina' THEN 'South America'
        WHEN 'Chile' THEN 'South America'
        WHEN 'USA' THEN 'North America'
        WHEN 'Canada' THEN 'North America'
        ELSE 'Unknown'
        END AS Continent
FROM Customer
```

	FirstName	LastName	Country	Continent
1	Luís	Gonçalves	Brazil	South America
2	Leonie	Köhler	Germany	Unknown
3	François	Tremblay	Canada	North America
4	Bjørn	Hansen	Norway	Unknown
5	František	Wichterlová	Czech Republic	Unknown
6	Helena	Holý	Czech Republic	Unknown
7	Astrid	Gruber	Austria	Unknown
8	Daan	Peeters	Belgium	Unknown
9	Kara	Nielsen	Denmark	Unknown
10	Eduardo	Martins	Brazil	South America
11	Alexandre	Rocha	Brazil	South America
12	Roberto	Almeida	Brazil	South America
13	Fernanda	Ramos	Brazil	South America
14	Mark	Philips	Canada	North America
15	Jennifer	Peterson	Canada	North America
16	Frank	Harris	USA	North America
17	Jack	Smith	USA	North America

This example is identical to the one on the previous slide except I have include the ELSE clause. I set the ELSE clause equal to “Unknown”. This causes all records that didn’t have a match in the WHEN THEN clauses to resolve to “Unknown” instead of NULL.

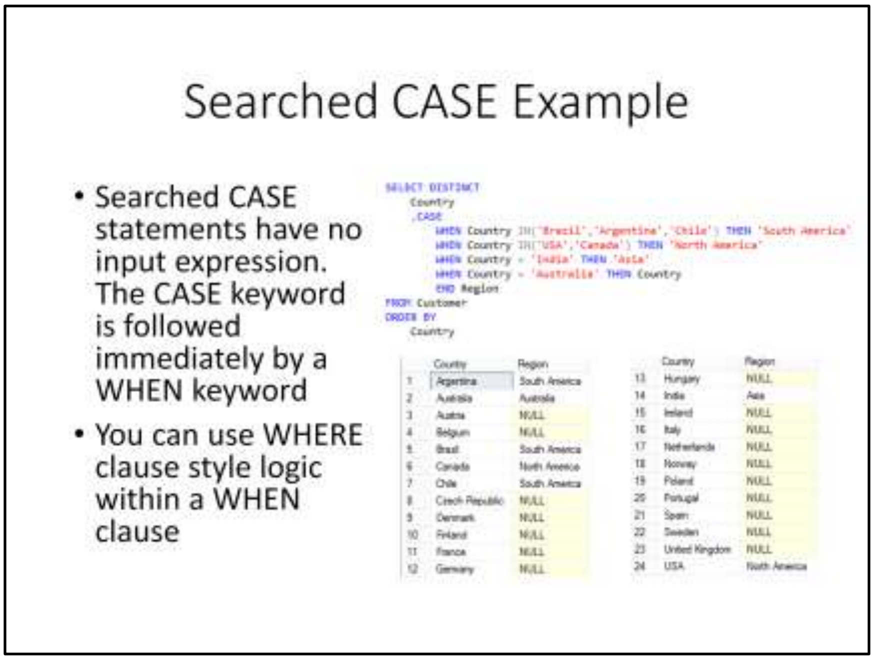


Searched CASE Statement

- Evaluates, in the order specified, *Boolean_expression* for each WHEN clause.
- Returns *result_expression* of the first *Boolean_expression* that evaluates to TRUE.
- If no *Boolean_expression* evaluates to TRUE, the Database Engine returns the *else_result_expression* if an ELSE clause is specified, or a NULL value if no ELSE clause is specified.

```
Searched CASE expression:
CASE
     WHEN Boolean_expression THEN result_expression [ ...n ]
     [ ELSE else_result_expression ]
END
```

A searched CASE statement allows more flexibility in the WHEN clause. You can use operators and other WHERE clause style logic within each WHEN clause. Also you are not limited to evaluating against a single column. You can evaluate multiple conditions within a single WHEN clause. Syntactically a searched CASE is identical to a simple CASE except there is no input expression after the CASE keyword, also you have more options with the search conditions.



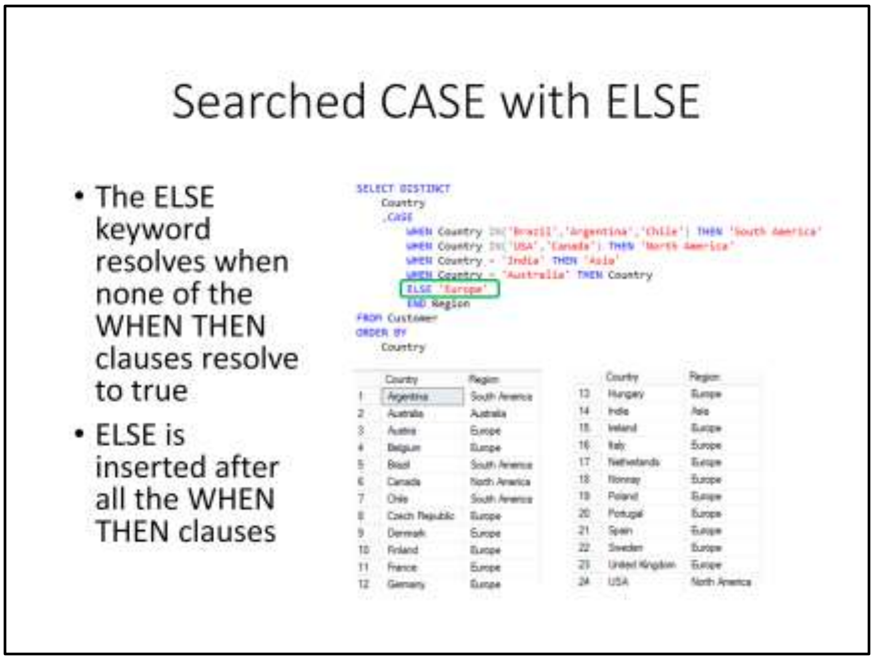
Searched CASE Example

- Searched CASE statements have no input expression. The CASE keyword is followed immediately by a WHEN keyword
- You can use WHERE clause style logic within a WHEN clause

```sql
SELECT DISTINCT
    Country
    ,CASE
        WHEN Country IN('Brazil','Argentina','Chile') THEN 'South America'
        WHEN Country IN('USA','Canada') THEN 'North America'
        WHEN Country = 'India' THEN 'Asia'
        WHEN Country = 'Australia' THEN Country
        END Region
FROM Customer
ORDER BY
    Country
```

	Country	Region
1	Argentina	South America
2	Australia	Australia
3	Austria	NULL
4	Belgium	NULL
5	Brazil	South America
6	Canada	North America
7	Chile	South America
8	Czech Republic	NULL
9	Denmark	NULL
10	Finland	NULL
11	France	NULL
12	Germany	NULL
13	Hungary	NULL
14	India	Asia
15	Ireland	NULL
16	Italy	NULL
17	Netherlands	NULL
18	Norway	NULL
19	Poland	NULL
20	Portugal	NULL
21	Spain	NULL
22	Sweden	NULL
23	United Kingdom	NULL
24	USA	North America

In the example I am still checking for a country name, but I am able to use WHERE style logic in the WHEN clauses. In this instance I am using the IN clause in addition to some equal operators. I do not have to limit myself to using the same column either. I can have one WHEN clause referencing country, and the next referencing the email column if the logic made sense to do so.



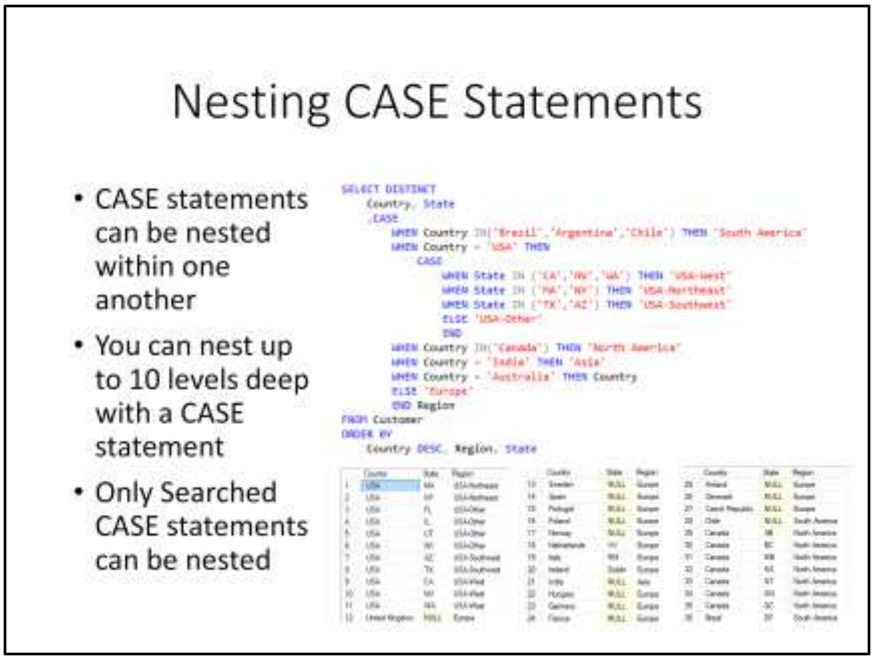
Searched CASE with ELSE

- The ELSE keyword resolves when none of the WHEN THEN clauses resolve to true
- ELSE is inserted after all the WHEN THEN clauses

```
SELECT DISTINCT
    Country
    ,CASE
        WHEN Country IN('Brazil','Argentina','Chile') THEN 'South America'
        WHEN Country IN('USA','Canada') THEN 'North America'
        WHEN Country = 'India' THEN 'Asia'
        WHEN Country = 'Australia' THEN Country
        ELSE 'Europe'
        END Region
FROM Customer
ORDER BY
    Country
```

	Country	Region
1	Argentina	South America
2	Australia	Australia
3	Austria	Europe
4	Belgium	Europe
5	Brazil	South America
6	Canada	North America
7	Chile	South America
8	Czech Republic	Europe
9	Denmark	Europe
10	Finland	Europe
11	France	Europe
12	Germany	Europe
13	Hungary	Europe
14	India	Asia
15	Ireland	Europe
16	Italy	Europe
17	Netherlands	Europe
18	Norway	Europe
19	Poland	Europe
20	Portugal	Europe
21	Spain	Europe
22	Sweden	Europe
23	United Kingdom	Europe
24	USA	North America

This example is identical to the previous slide except I added the ELSE clause. This is the catch all when none of the WHEN THEN clauses evaluate to true. The ELSE syntax is identical to that in the simple CASE statement.



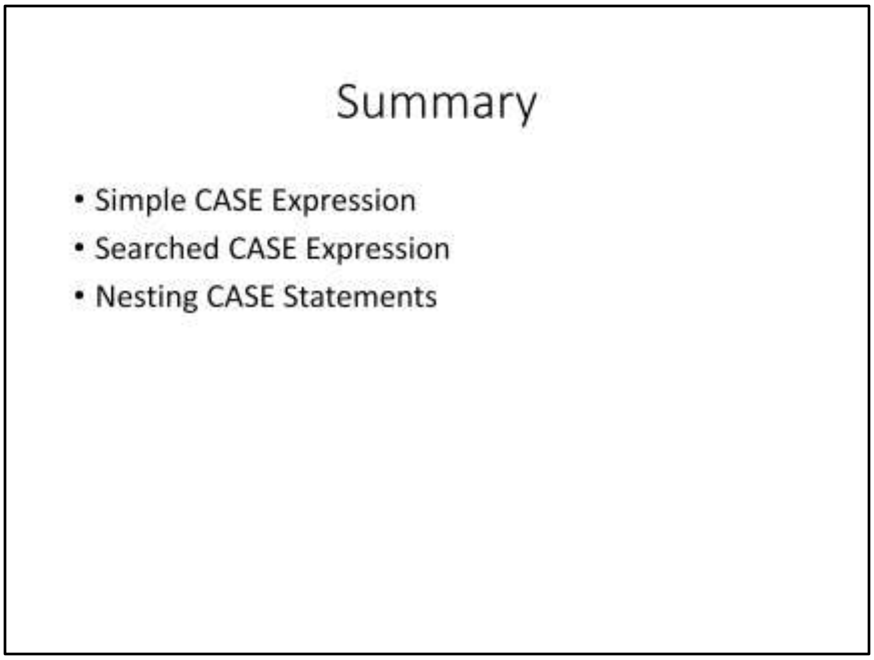
Nesting CASE Statements

- CASE statements can be nested within one another
- You can nest up to 10 levels deep with a CASE statement
- Only Searched CASE statements can be nested

```
SELECT DISTINCT
    Country, State
    ,CASE
        WHEN Country IN('Brazil','Argentina','Chile') THEN 'South America'
        WHEN Country = 'USA' THEN
            CASE
                WHEN State IN ('CA','NV','WA') THEN 'USA-West'
                WHEN State IN ('MA','NY') THEN 'USA-Northeast'
                WHEN State IN ('TX','AZ') THEN 'USA-Southwest'
                ELSE 'USA-Other'
                END
        WHEN Country IN('Canada') THEN 'North America'
        WHEN Country = 'India' THEN 'Asia'
        WHEN Country = 'Australia' THEN Country
        ELSE 'Europe'
        END Region
FROM Customer
ORDER BY
    Country DESC, Region, State
```

	Country	State	Region
1	USA	MA	USA-Northeast
2	USA	NY	USA-Northeast
3	USA	FL	USA-Other
4	USA	IL	USA-Other
5	USA	UT	USA-Other
6	USA	WI	USA-Other
7	USA	AZ	USA-Southwest
8	USA	TX	USA-Southwest
9	USA	CA	USA-West
10	USA	NV	USA-West
11	USA	WA	USA-West
12	United Kingdom	NULL	Europe
13	Sweden	NULL	Europe
14	Spain	NULL	Europe
15	Portugal	NULL	Europe
16	Poland	NULL	Europe
17	Norway	NULL	Europe
18	Netherlands	[illegible]	Europe
19	Italy	[illegible]	Europe
20	Ireland	Dublin	Europe
21	India	NULL	Asia
22	Hungary	NULL	Europe
23	Germany	NULL	Europe
24	France	NULL	Europe
25	Finland	NULL	Europe
26	Denmark	NULL	Europe
27	Czech Republic	NULL	Europe
28	Chile	NULL	South America
29	Canada	AB	North America
30	Canada	BC	North America
31	Canada	MB	North America
32	Canada	NS	North America
33	Canada	NT	North America
34	Canada	ON	North America
35	Canada	QC	North America
36	Brazil	SP	South America

Nesting CASE expressions is when you put one CASE statement inside of another. This allows you to create some very complex logic conditions within your CASE statement. In the example I am searching for countries and outputting the country's continent when I find a match. However in the case of the USA I added a nested CASE statement that further checks for the state. The output is a region of the united states based on the state location instead of the continent name.



Summary

- Simple CASE Expression
- Searched CASE Expression
- Nesting CASE Statements

This concludes the presentation on CASE statements.