



Conditional Expressions and Operators

By
Vijaya Nandini M



SQL

- Section Overview
 - CASE
 - COALESCE
 - NULLIF
 - CAST
 - Views
 - Import and Export Functionality



SQL

- These keywords and functions will allow us to add logic to our commands and workflows in SQL.
- Let's get started!



CASE



SQL

- We can use the **CASE** statement to only execute SQL code when certain conditions are met.
- This is very similar to **IF/ELSE** statements in other programming languages.



SQL

- There are two main ways to use a CASE statement, either a general CASE or a CASE expression.
- Both methods can lead to the same results.
- Let's first show the syntax for a “general” CASE.



SQL

- General Syntax
 - CASE

WHEN condition1 THEN result1

WHEN condition2 THEN result2

ELSE some_other_result

END



SQL

- Simple Example
 - `SELECT a,`
`CASE WHEN a = 1 THEN 'one'`
`WHEN a = 2 THEN 'two'`
`ELSE 'other'`

a
1
2



SQL

- Simple Example
 - `SELECT a,`
`CASE WHEN a = 1 THEN 'one'`
`WHEN a = 2 THEN 'two'`
`ELSE 'other'`
`END`
`FROM test;`

a	Case
1	one
2	two



SQL

- Simple Example
 - `SELECT a,
CASE WHEN a = 1 THEN 'one'
 WHEN a = 2 THEN 'two'
 ELSE 'other' AS label
END
FROM test;`

a	label
1	one
2	two



SQL

- The CASE expression syntax first evaluates an expression then compares the result with each value in the WHEN clauses sequentially.



SQL

- *CASE Expression Syntax*
 - CASE expression

WHEN value1 THEN result1

WHEN value2 THEN result2

ELSE some_other_result

END



SQL

- Rewriting our previous example:
 - `SELECT a,`
 `CASE a WHEN 1 THEN 'one'`
 `WHEN 2 THEN 'two'`
 `ELSE 'other'`
 `END`
 `FROM test;`

a	label
1	one
2	two



CASE

Challenge Task



SQL

- Challenge
 - We want to know and compare the various amounts of films we have per movie rating.
 - Use CASE and the dvdrental database to re-create this table:

r	pg	pg13
bigint	bigint	bigint
195	194	223



SQL

- Hints
 - Review our CASE expression example that used SUM in the previous lecture



COALESCE



SQL

- The COALESCE function accepts an unlimited number of arguments. It returns the first argument that is not null. If all arguments are null, the COALESCE function will return null.
 - COALESCE (arg_1,arg_2,...,arg_n)



SQL

- Example
 - `SELECT COALESCE (1, 2)`
 - 1
 - `SELECT COALESCE(NULL, 2, 3)`
 - 2



SQL

- The COALESCE function becomes useful when querying a table that contains null values and substituting it with another value. Let's see a simple example



SQL

- Table of Products
 - Price and Discount in Dollars

Item	Price	Discount
A	100	20
B	300	null
C	200	10



SQL

- Table of Products
 - What is the final price?

Item	Price	Discount
A	100	20
B	300	null
C	200	10



SQL

- `SELECT item,(price - discount) AS final
FROM table`

Item	final
A	80
B	null
C	190



SQL

- `SELECT item,(price - discount) AS final`
`FROM table`
- Doesn't work for item B, should be 300.

Item	final
A	80
B	null
C	190



SQL

```
SELECT item,(price - COALESCE(discount,0))  
AS final FROM table
```

Item	final
A	80
B	300
C	190



SQL

- Keep the COALESCE function in mind in case you encounter a table with null values that you want to perform operations on!



CAST



SQL

- The CAST operator lets you convert from one data type into another.
- Keep in mind not every instance of a data type can be CAST to another data type, it must be reasonable to convert the data, for example '5' to an integer will work, 'five' to an integer will not.



SQL

- Syntax for CAST function
 - `SELECT CAST('5' AS INTEGER)`
- PostgreSQL CAST operator
 - `SELECT '5'::INTEGER`



SQL

- Keep in mind you can then use this in a SELECT query with a column name instead of a single instance.
 - **SELECT CAST(date AS TIMESTAMP)
FROM table**



NULLIF



SQL

- The NULLIF function takes in 2 inputs and returns NULL if both are equal, otherwise it returns the first argument passed.
 - NULLIF(arg1,arg2)
- Example
 - NULLIF(10,10)
 - Returns NULL



SQL

- The NULLIF function takes in 2 inputs and returns NULL if both are equal, otherwise it returns the first argument passed.
 - NULLIF(arg1,arg2)
- Example
 - NULLIF(10,12)
 - Returns 10



SQL

- This becomes very useful in cases where a NULL value would cause an error or unwanted result.
- Let's see an example.



SQL

- Given this table calculate the ratio of Department A to Department B

Name	Department
Lauren	A
Vinton	A
Claire	B



SQL

- We can see easily the ratio of A to B is 2:1 or 200%
- Let's use SQL to solve for this with CASE

Name	Department
Lauren	A
Vinton	A
Claire	B



SQL

- Let's jump to pgAdmin, quickly create this table and walk through solving the RATIO and why we may need NULLIF

Name	Department
Lauren	A
Vinton	A
Claire	B



VIEWS

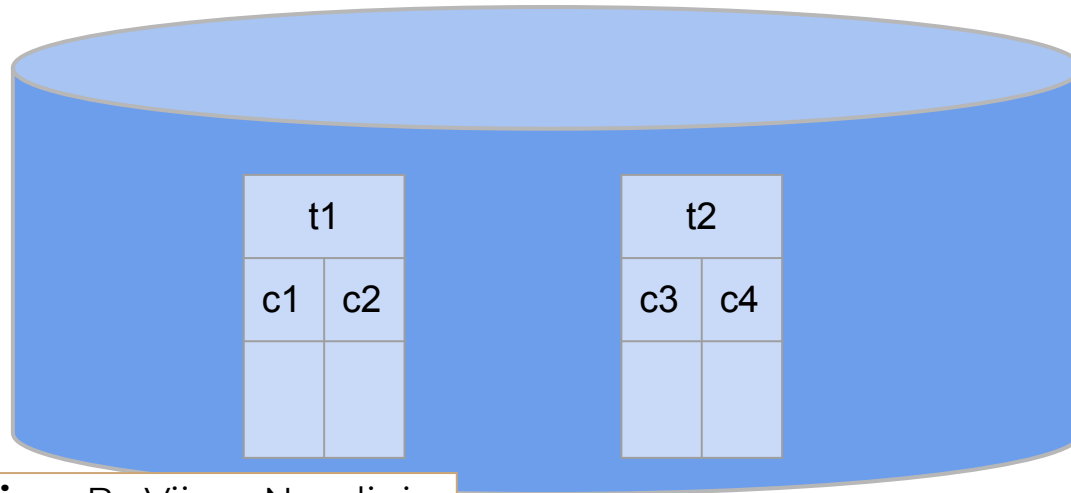


SQL

- Often there are specific combinations of tables and conditions that you find yourself using quite often for a project.
- Instead of having to perform the same query over and over again as a starting point, you can create a VIEW to quickly see this query with a simple call.



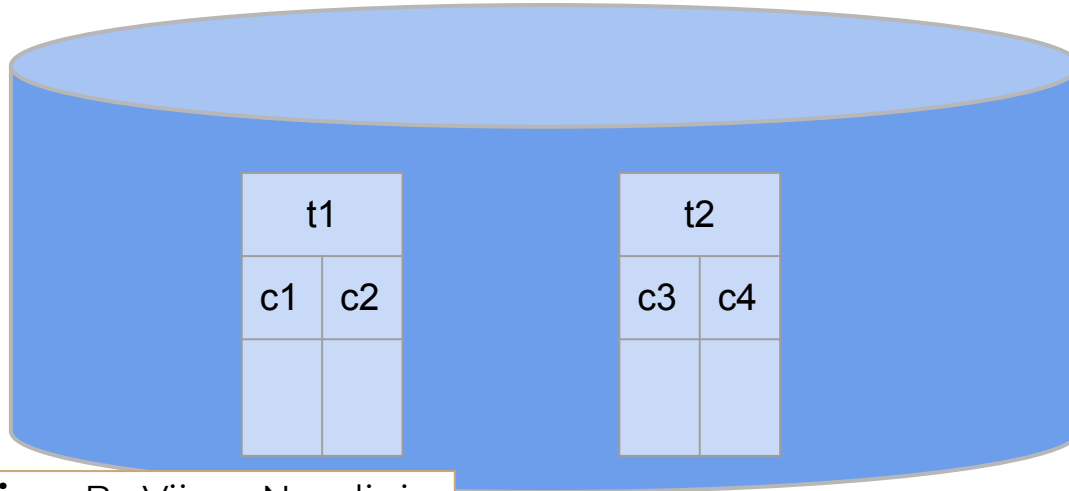
SQL





SQL

c1	c2	c3	c3

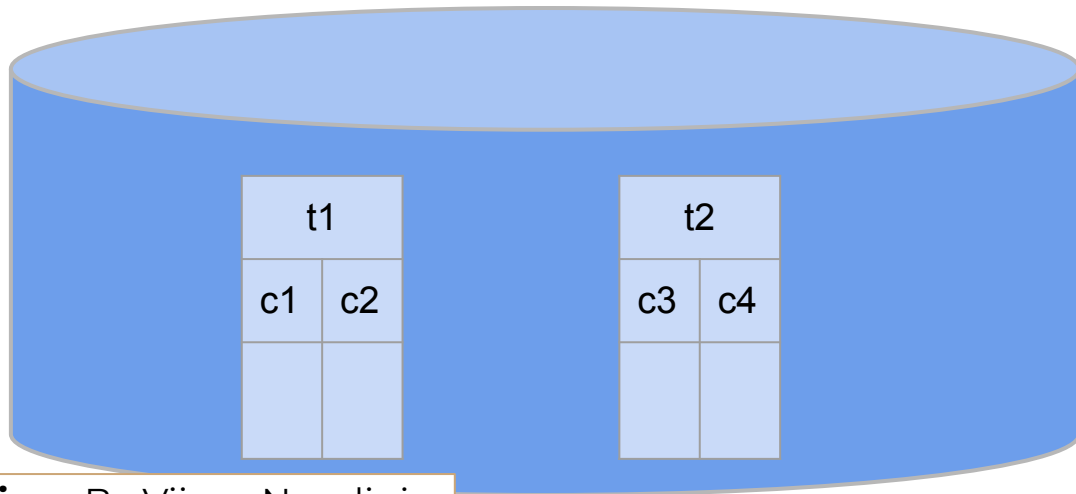


```
SELECT c1,c2,c3,c4  
FROM t1  
INNER JOIN t2  
ON t1.c1 = t2.c3
```



SQL

c1	c2	c3	c3



```
SELECT * FROM view
```



```
SELECT c1,c2,c3,c4  
FROM t1  
INNER JOIN t1  
ON t1.c1 = t2.c3
```



SQL

- A view is a database object that is of a stored query.
- A view can be accessed as a virtual table in PostgreSQL.
- Notice that a view does not store data physically, it simply stores the query.
- You can also update and alter existing views.



Importing and Exporting Data



SQL

- Let's explore the Import/Export functionality of PgAdmin, which allows us to import data from a .csv file to an already existing table.
- There are some important notes to keep in mind when using Import/Export



SQL

- Important Note!
 - Not every outside data file will work, variations in formatting, macros, data types, etc. may prevent the Import command from reading the file, at which point, you must edit your file to be compatible with SQL.



SQL

- Details of compatible file types and examples are available in the online documentation:
- **postgresql.org/docs/12/sql-copy.html**



SQL

- Important Note!
 - You **MUST** provide the 100% correct file path to your outside file, otherwise the Import command will fail to find the file.
 - The most common mistake is failing to provide the correct file path, confirm the file's location under its properties.



SQL

- **VERY** Important Note!
 - The Import command **DOES NOT** create a table for you.
 - It assumes a table is already created.
 - Currently there is no automated way within pgAdmin to create a table directly from a .csv file.