CASE Statements & Mastering Subqueries & CTEs

🎯 Session Objectives:

- ✅ Understand how SQL CASE statements work.
- ✅ Use CASE WHEN to create new flags/segments.
- ✅ Apply conditional logic for aggregation.
- ✅ Use CASE in SELECT, UPDATE, and JOIN.
- ✅ Understand Common Table Expressions (CTEs) and why we use them
- ✅ Apply subqueries in SELECT, FROM, WHERE, HAVING, and JOIN
- ✅ Use nested & correlated subqueries for advanced querying
- ✅ Optimize queries using subqueries

CASE STATEMENT SYNTAX:

```
SELECT column1,
       CASE
           WHEN condition1 THEN result1
           WHEN condition2 THEN result2
           ELSE default_result
       END AS alias_name
FROM table_name;
```

```
-- Create a new column 'Income_Category'-> Categorize the Customers with AnnualIncome
/*
        CASES - <50k - 'Low Income'
              - [50k - 100k] - 'Moderate Income'
              - > 100k - 'High Income'
*/
```

```
USE Weekend_analysis;

SELECT * FROM Customers;
```

```
SELECT
    CustomerKey,
    FullName,
    AnnualIncome,
    CASE
        WHEN AnnualIncome < 50000 THEN 'Low Income'
        WHEN AnnualIncome BETWEEN 50000 AND 100000 THEN 'Moderate Income'
        ELSE 'High Income'
    END AS Income_Category
FROM Customers;
```

CustomerKey	FullName	AnnualIncome	Income_Category
11000	JON YANG	90000	Moderate Income
11001	EUGENE HUANG	60000	Moderate Income
11002	RUBEN TORRES	60000	Moderate Income
11003	CHRISTY ZHU	NULL	High Income
11004	ELIZABETH JOHNSON	80000	Moderate Income
11005	JULIO RUIZ	70000	Moderate Income
11007	MARCO MEHTA	60000	Moderate Income
11008	ROBIN VERHOFF	60000	Moderate Income
11009	SHANNON CARLSON	70000	Moderate Income
11010	JACQUELYN SUAREZ	70000	Moderate Income
11011	CURTIS LU	60000	Moderate Income
11012	LAUREN WALKER	100000	Moderate Income
11013	IAN JENKINS	100000	Moderate Income
11014	SYDNEY BENNETT	100000	Moderate Income

CustomerKey	FullName	AnnualIncome	Income_Category
11075	FELICIA JIMENEZ	80000	Moderate Income
11076	BLAKE ANDERSON	80000	Moderate Income
11077	LEAH YE	80000	Moderate Income
11078	GINA MARTIN	40000	Low Income
11079	DONALD GONZALEZ	160000	High Income
11080	DAMIEN CHANDER	170000	High Income
11081	SAVANNAH BAKER	NULL	High Income
11082	ANGELA BUTLER	130000	High Income
11083	ALYSSA COX	130000	High Income
11084	LUCAS PHILLIPS	80000	Moderate Income
11085	EMILY JOHNSON	60000	Moderate Income
11086	RYAN BROWN	70000	Moderate Income
11087	TAMARA LIANG	70000	Moderate Income
11089	ABIGAIL PRICE	80000	Moderate Income

```sql
-- Handling Null Cases
SELECT
    CustomerKey,
    FullName,
    AnnualIncome,
    CASE
        WHEN AnnualIncome IS NULL THEN 'No Income Defined'
        WHEN AnnualIncome < 50000 THEN 'Low Income'
        WHEN AnnualIncome BETWEEN 50000 AND 100000 THEN 'Moderate Income'
        ELSE 'High Income'
    END AS Income_Category
FROM Customers;
```

CustomerKey	FullName	AnnualIncome	Income_Category
11000	JON YANG	90000	Moderate Income
11001	EUGENE HUANG	60000	Moderate Income
11002	RUBEN TORRES	60000	Moderate Income
11003	CHRISTY ZHU	NULL	No Income Defined
11004	ELIZABETH JOHNSON	80000	Moderate Income
11005	JULIO RUIZ	70000	Moderate Income
11007	MARCO MEHTA	60000	Moderate Income
11008	ROBIN VERHOFF	60000	Moderate Income
11009	SHANNON CARLSON	70000	Moderate Income
11010	JACQUELYN SUAREZ	70000	Moderate Income
11011	CURTIS LU	60000	Moderate Income
11012	LAUREN WALKER	100000	Moderate Income
11013	IAN JENKINS	100000	Moderate Income
11014	SYDNEY BENNETT	100000	Moderate Income
11015	CHLOE YOUNG	NULL	No Income Defined
11016	WYATT HILL	30000	Low Income

Create IncomeCategory using Alter Command and update the new column

```
857 •  DESC Customers;
858
859 •  ALTER TABLE Customers
860    ADD COLUMN IncomeCategory VARCHAR(100)
861    AFTER AnnualIncome;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
CustomerKey	int	NO	PRI	NULL	
Prefix	text	YES		NULL	
FirstName	varchar(50)	YES		NULL	
LastName	varchar(50)	YES		NULL	
FullName	varchar(100)	YES		NULL	
DateOfBirth	text	YES		NULL	
MaritalStatus	text	YES		NULL	
EmailAddress	varchar(100)	YES		NULL	
Gender	text	YES		NULL	
Regions	varchar(50)	YES		NULL	
AnnualIncome	int	YES		NULL	
IncomeCategory	varchar(100)	YES		NULL	
TotalChildren	int	YES		NULL	
EducationLevel	text	YES		NULL	
Occupation	text	YES		NULL	
HomeOwner	text	YES		NULL	

```
DESC Customers;

ALTER TABLE Customers
ADD COLUMN IncomeCategory VARCHAR(100)
AFTER AnnualIncome;
```

```
864 •  SELECT
865        CustomerKey,
866        FullName,
867        AnnualIncome,
868        IncomeCategory
869    FROM Customers;
```

Result Grid | Filter Rows: | Edit:

CustomerKey	FullName	AnnualIncome	IncomeCategory
11000	JON YANG	90000	NULL
11001	EUGENE HUANG	60000	NULL
11002	RUBEN TORRES	60000	NULL
11003	CHRISTY ZHU	NULL	NULL
11004	ELIZABETH JOHNSON	80000	NULL
11005	JULIO RUIZ	70000	NULL
11007	MARCO MEHTA	60000	NULL
11008	ROBIN VERHOFF	60000	NULL
11009	SHANNON CARLSON	70000	NULL
11010	JACQUELYN SUAREZ	70000	NULL
11011	CURTIS LU	60000	NULL
11012	LAUREN WALKER	100000	NULL
11013	IAN JENKINS	100000	NULL
11014	SYDNEY BENNETT	100000	NULL
11015	CHLOE YOUNG	NULL	NULL
11016	WYATT HILL	30000	NULL

```
-- UPDATE <IncomeCateogry> From Customers Table.
```

```
SET SQL_SAFE_UPDATES = 0;

UPDATE Customers
SET IncomeCategory =
        CASE
            WHEN AnnualIncome IS NULL THEN 'No Income Defined'
            WHEN AnnualIncome < 50000 THEN 'Low Income'
            WHEN AnnualIncome BETWEEN 50000 AND 100000 THEN 'Moderate Income'
            ELSE 'High Income'
        END;
```

CustomerKey	FullName	AnnualIncome	IncomeCategory
11000	JON YANG	90000	Moderate Income
11001	EUGENE HUANG	60000	Moderate Income
11002	RUBEN TORRES	60000	Moderate Income
11003	CHRISTY ZHU	NULL	No Income Defined
11004	ELIZABETH JOHNSON	80000	Moderate Income
11005	JULIO RUIZ	70000	Moderate Income
11007	MARCO MEHTA	60000	Moderate Income
11008	ROBIN VERHOFF	60000	Moderate Income
11009	SHANNON CARLSON	70000	Moderate Income
11010	JACQUELYN SUAREZ	70000	Moderate Income
11011	CURTIS LU	60000	Moderate Income
11012	LAUREN WALKER	100000	Moderate Income
11013	IAN JENKINS	100000	Moderate Income
11014	SYDNEY BENNETT	100000	Moderate Income
11015	CHLOE YOUNG	NULL	No Income Defined
11016	WYATT HILL	30000	Low Income

CONDITIONAL AGGREGATION WITH CASE SYNTAX:

```
SELECT column,
       SUM(CASE WHEN condition THEN value ELSE 0 END) AS label
FROM table
GROUP BY column;
```

```
893 •  SELECT
894        TerritoryKey,
895        OrderQuantity
896    FROM Sales2017;
```

Result Grid | Filter Rows:

TerritoryKey	OrderQuantity
1	2
1	1
1	1
1	2
1	1
1	1
4	1
4	1

	High	Moderate	Low
10 - 2	0	2	0
10 - 2	0	2	0
10 - 1	0	0	1
10 - 1	0	0	1
10 - 3	3	0	0
10 - 2	0	2	0
10 - 1	0	0	1
10 - 3	3	0	0
10 - 2	0	2	0
10 - 1	0	0	1
10 - 1	0	0	1
10 - 2	0	2	0
10 t.k	6	10	5

```
SELECT
    TerritoryKey,
    SUM(CASE WHEN OrderQuantity = 3 THEN OrderQuantity ELSE 0 END) AS High_Sales,
    SUM(CASE WHEN OrderQuantity = 2 THEN OrderQuantity ELSE 0 END) AS Moderate_Sales,
    SUM(CASE WHEN OrderQuantity = 1 THEN OrderQuantity ELSE 0 END) AS Low_Sales
FROM Sales2017
GROUP BY TerritoryKey;
```

TerritoryKey	High_Sales	Moderate_Sales	Low_Sales
1	801	3670	2318
4	1284	4824	3282
6	852	3240	1768
10	630	2756	1707
8	600	2270	1516
9	1185	4764	3538
7	498	2296	1442
5	0	20	15
3	0	8	6
2	3	14	7

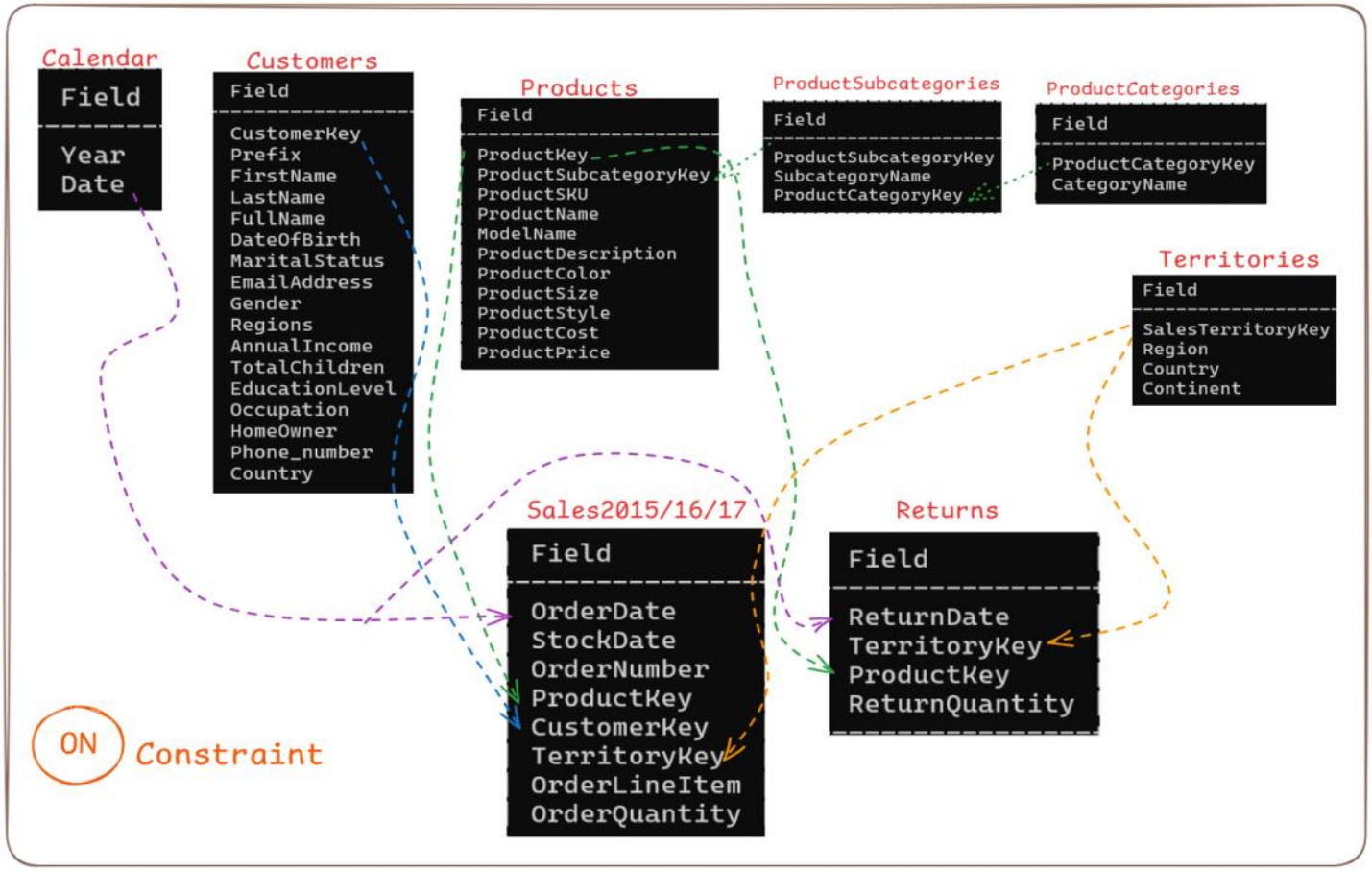
```
SELECT
    TerritoryKey,
    SUM(CASE WHEN OrderQuantity > 2 THEN OrderQuantity ELSE 0 END) AS High_Sales,
    SUM(CASE WHEN OrderQuantity BETWEEN 1 AND 2 THEN OrderQuantity ELSE 0 END) AS Moderate_Sales,
    SUM(CASE WHEN OrderQuantity < 1 THEN OrderQuantity ELSE 0 END) AS Low_Sales
FROM Sales2017
GROUP BY TerritoryKey;
```

TerritoryKey	High_Sales	Moderate_Sales	Low_Sales
1	801	5988	0
4	1284	8106	0
6	852	5008	0
10	630	4463	0
8	600	3786	0
9	1185	8302	0
7	498	3738	0
5	0	35	0
3	0	14	0
2	3	21	0

```sql
SELECT
    TerritoryKey,
    SUM(CASE WHEN OrderQuantity = 3 THEN OrderQuantity ELSE 0 END) AS High_Sales,
    SUM(CASE WHEN OrderQuantity = 2 THEN OrderQuantity ELSE 0 END) AS Moderate_Sales,
    SUM(CASE WHEN OrderQuantity = 1 THEN OrderQuantity ELSE 0 END) AS Low_Sales
FROM (
    SELECT * FROM Sales2015
    UNION
    SELECT * FROM Sales2016
    UNION
    SELECT * FROM Sales2017
) AS sub
GROUP BY TerritoryKey;
```

TerritoryKey	High_Sales	Moderate_Sales	Low_Sales
1	1515	6472	4526
4	2172	8560	6459
9	2076	8316	7559
6	1578	5934	3382
10	1191	4954	3549
7	936	3998	2928
8	1032	3946	2972
5	3	26	20
2	3	22	15
3	3	16	11

Challenge 1 : Segment the customers by Return Behaviour



```
SELECT
    c.CustomerKey,
    c.FullName,
    SUM(r.ReturnQuantity) AS TotalReturnQty
FROM Customers c
LEFT JOIN Sales2015 s
ON c.CustomerKey = s.CustomerKey
LEFT JOIN Returns r
ON r.ProductKey = s.ProductKey
GROUP BY 1,2; -- 2062 row(s) returned
```

CustomerKey	FullName	TotalReturnQty
11215	ANA PERRY	NULL
11216	JASMINE TORRES	8
11217	NATALIE ADAMS	4
11218	OLIVIA BROWN	NULL
11219	CHARLES COOK	NULL
11220	ERICA HUANG	NULL
11221	NATHAN PERRY	NULL
11222	ALEXANDRA RIVERA	NULL
11223	HAILEY PATTERSON	NULL
11224	TIFFANY LI	4
11226	SYDNEY ROSS	NULL
11227	MARSHALL CHAVEZ	5
11228	ASHLEY JONES	NULL
11229	ADRIAN STEWART	NULL

```
SELECT
    c.CustomerKey,
    c.FullName,
    SUM(COALESCE(r.ReturnQuantity,0)) AS TotalReturnQty,
    CASE
        WHEN SUM(COALESCE(r.ReturnQuantity,0)) > 5 THEN 'Frequent Returner'
        WHEN SUM(COALESCE(r.ReturnQuantity,0)) BETWEEN 1 AND 5 THEN 'Moderate Returner'
        ELSE 'Non Returner'
    END AS ReturnBehavior
FROM Customers c
LEFT JOIN Sales2015 s
ON c.CustomerKey = s.CustomerKey
LEFT JOIN Returns r
ON r.ProductKey = s.ProductKey
GROUP BY 1,2; -- 2062 row(s) returned
```

Label Revenue by Category & Region

Calculate the Total Revenue per Category-Region Pair and tag them as Low, Moderate or High Revenue

>= 2Lakh -> High Revenue
>= 50k -> Moderate Revenue
ELSE 'Low Revenue'

```
SELECT
    pc.CategoryName,
    t.Region,
    ROUND(SUM(p.ProductPrice * s.OrderQuantity),0) AS TotalRevenue,
    CASE
        WHEN ROUND(SUM(p.ProductPrice * s.OrderQuantity),0) > 200000 THEN 'High Revenue'
        WHEN ROUND(SUM(p.ProductPrice * s.OrderQuantity),0) > 50000 THEN 'Moderate Revenue'
        ELSE 'Low Revenue'
    END AS RevenueCategory
FROM ProductCategories pc
JOIN ProductSubcategories ps
ON pc.ProductCategoryKey = ps.ProductCategoryKey
JOIN Products p
ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
JOIN Sales2017 s
ON s.ProductKey = p.ProductKey
JOIN Territories t
ON t.SalesTerritoryKey = s.TerritoryKey
GROUP BY 1,2;
```

CategoryName	Region	TotalRevenue	RevenueCategory
Accessories	Northwest	76725	Moderate Revenue
Bikes	Northwest	1096794	High Revenue
Accessories	Southwest	106170	Moderate Revenue
Clothing	Southwest	46631	Low Revenue
Accessories	Canada	70377	Moderate Revenue
Accessories	United Kingdom	55848	Moderate Revenue
Clothing	United Kingdom	19433	Low Revenue
Accessories	Germany	47030	Low Revenue
Bikes	Germany	953155	High Revenue
Accessories	Australia	102356	Moderate Revenue
Bikes	United Kingdom	1043807	High Revenue
Bikes	Australia	2261783	High Revenue
Clothing	Australia	43966	Low Revenue
Bikes	Southwest	1758417	High Revenue
Clothing	Germany	15472	Low Revenue
Clothing	Northwest	35372	Low Revenue
Accessories	France	47967	Low Revenue
Bikes	France	810084	High Revenue
Clothing	Canada	32218	Low Revenue

Classify the CategoryName by its Return Volume

- Identify the Products with High, Moderate or Low Returns across regions and categories

```
>= 50 -> "High Returns"
>=25 -> ''Moderate Returns
Else [0-24] -> 'Low Returns'
```

```
SELECT
    pc.CategoryName,
    ps.SubCategoryName,
    t.Region,
    ROUND(SUM(r.ReturnQuantity),0) AS TotalReturnQty
FROM ProductCategories pc
JOIN ProductSubcategories ps
ON pc.ProductCategoryKey = ps.ProductCategoryKey
JOIN Products p
ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
JOIN Returns r
ON r.ProductKey = p.ProductKey
JOIN Territories t
ON t.SalesTerritoryKey = r.TerritoryKey
GROUP BY 1,2,3;
```

CategoryName	SubCategoryName	Region	TotalReturnQty
Accessories	Helmets	United Kingdom	23
Accessories	Helmets	Australia	34
Accessories	Helmets	Germany	16
Accessories	Helmets	France	26
Accessories	Helmets	Canada	22
Accessories	Helmets	Southwest	43
Accessories	Helmets	Northwest	24
Clothing	Caps	United Kingdom	5
Clothing	Caps	Australia	8
Clothing	Caps	Germany	6
Clothing	Caps	France	4
Clothing	Caps	Canada	5
Clothing	Caps	Southwest	9
Clothing	Caps	Northwest	9

CASE STATEMENT

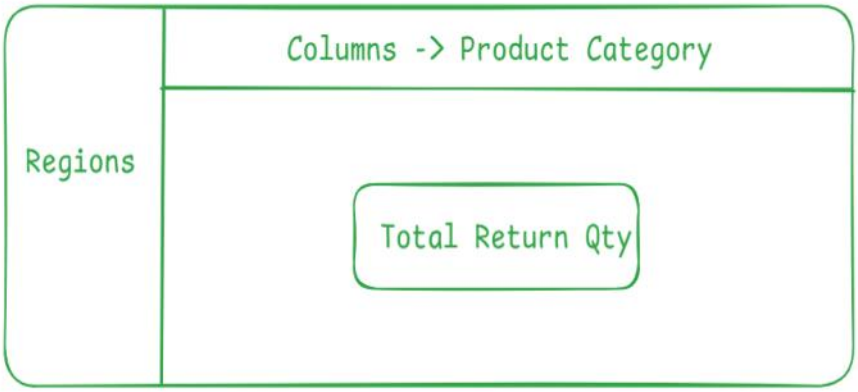
```
SELECT
    pc.CategoryName,
    ps.SubCategoryName,
    t.Region,
    SUM(r.ReturnQuantity) AS TotalReturnQty,
    CASE
        WHEN SUM(r.ReturnQuantity) >= 50 THEN 'High Returns'
        WHEN SUM(r.ReturnQuantity) >= 25 THEN 'Moderate Returns'
        ELSE 'Low Returns'
    END AS ReturnCateogry
FROM ProductCategories pc
JOIN ProductSubcategories ps
ON pc.ProductCategoryKey = ps.ProductCategoryKey
JOIN Products p
ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
JOIN Returns r
ON r.ProductKey = p.ProductKey
JOIN Territories t
ON t.SalesTerritoryKey = r.TerritoryKey
GROUP BY 1,2,3;
```

CategoryName	SubCategoryName	Region	TotalReturnQty	ReturnCateogry
Accessories	Helmets	United Kingdom	23	Low Returns
Accessories	Helmets	Australia	34	Moderate Returns
Accessories	Helmets	Germany	16	Low Returns
Accessories	Helmets	France	26	Moderate Returns
Accessories	Helmets	Canada	22	Low Returns
Accessories	Helmets	Southwest	43	Moderate Returns
Accessories	Helmets	Northwest	24	Low Returns
Clothing	Caps	United Kingdom	5	Low Returns
Clothing	Caps	Australia	8	Low Returns
Clothing	Caps	Germany	6	Low Returns
Clothing	Caps	France	4	Low Returns
Clothing	Caps	Canada	5	Low Returns
Clothing	Caps	Southwest	9	Low Returns
Clothing	Caps	Northwest	9	Low Returns
Clothing	Jerseys	United Kingdom	7	Low Returns
Clothing	Jerseys	Australia	24	Low Returns

CONDITIONAL AGGREGATION WITH CASE SYNTAX:

```
SELECT column,
       SUM(CASE WHEN condition THEN value ELSE 0 END) AS label
FROM table
GROUP BY column;
```

```
-- Pivot Return Data by Region And Category
-- Show the total Return Quantity for each product category as columns by Regions.
```




```
SELECT
    t.Region,
    SUM(CASE WHEN pc.CategoryName = 'Bikes' THEN r.returnQuantity ELSE 0 END ) AS BikeReturns,
    SUM(CASE WHEN pc.CategoryName = 'Clothing' THEN r.returnQuantity ELSE 0 END ) AS ClothingReturns,
    SUM(CASE WHEN pc.CategoryName = 'Accessories' THEN r.returnQuantity ELSE 0 END ) AS AccessoriesReturns,
    SUM(CASE WHEN pc.CategoryName = 'Components' THEN r.returnQuantity ELSE 0 END ) AS ComponentsReturns
FROM ProductCategories pc
JOIN ProductSubcategories ps
ON pc.ProductCategoryKey = ps.ProductCategoryKey
JOIN Products p
ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
JOIN Returns r
ON r.ProductKey = p.ProductKey
JOIN Territories t
ON t.SalesTerritoryKey = r.TerritoryKey
GROUP BY 1;
```

Region	BikeReturns	ClothingReturns	AccessoriesReturns	ComponentsReturns
Australia	125	56	223	0
United Kingdom	51	24	129	0
Germany	53	22	88	0
Southwest	78	53	231	0
Canada	22	51	165	0
Northwest	58	40	172	0
France	42	23	121	0
Southeast	0	0	1	0

CTE SYNTAX:

```
WITH CTE_NAME AS (
  SELECT column1, column2 FROM table_name WHERE condition
)
SELECT * FROM CTE_NAME;
```

```
WITH Region_ReturnCategory AS (
        SELECT
            pc.CategoryName,
            ps.SubCategoryName,
            t.Region,
            SUM(r.ReturnQuantity) AS TotalReturnQty,
            CASE
                    WHEN SUM(r.ReturnQuantity) >= 50 THEN 'High Returns'
                    WHEN SUM(r.ReturnQuantity) >= 25 THEN 'Moderate Returns'
                    ELSE 'Low Returns'
            END AS ReturnCateogry
        FROM ProductCategories pc
        JOIN ProductSubcategories ps
        ON pc.ProductCategoryKey = ps.ProductCategoryKey
        JOIN Products p
        ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
        JOIN Returns r
        ON r.ProductKey = p.ProductKey
        JOIN Territories t
        ON t.SalesTerritoryKey = r.TerritoryKey
        GROUP BY 1,2,3
)
SELECT * FROM Region_ReturnCategory;
```

CategoryName	SubCategoryName	Region	TotalReturnQty	ReturnCateogry
Accessories	Helmets	United Kingdom	23	Low Returns
Accessories	Helmets	Australia	34	Moderate Returns
Accessories	Helmets	Germany	16	Low Returns
Accessories	Helmets	France	26	Moderate Returns
Accessories	Helmets	Canada	22	Low Returns
Accessories	Helmets	Southwest	43	Moderate Returns
Accessories	Helmets	Northwest	24	Low Returns
Clothing	Caps	United Kingdom	5	Low Returns
Clothing	Caps	Australia	8	Low Returns
Clothing	Caps	Germany	6	Low Returns
Clothing	Caps	France	4	Low Returns
Clothing	Caps	Canada	5	Low Returns
Clothing	Caps	Southwest	9	Low Returns
Clothing	Caps	Northwest	9	Low Returns

Now, Applying filter in this case is much easier.

```sql
WITH Region_ReturnCategory AS (
        SELECT
            pc.CategoryName,
            ps.SubCategoryName,
            t.Region,
            SUM(r.ReturnQuantity) AS TotalReturnQty,
            CASE
                    WHEN SUM(r.ReturnQuantity) >= 50 THEN 'High Returns'
                    WHEN SUM(r.ReturnQuantity) >= 25 THEN 'Moderate Returns'
                    ELSE 'Low Returns'
            END AS ReturnCateogry
        FROM ProductCategories pc
        JOIN ProductSubcategories ps
        ON pc.ProductCategoryKey = ps.ProductCategoryKey
        JOIN Products p
        ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
        JOIN Returns r
        ON r.ProductKey = p.ProductKey
        JOIN Territories t
        ON t.SalesTerritoryKey = r.TerritoryKey
        GROUP BY 1,2,3
)
SELECT * FROM Region_ReturnCategory WHERE ReturnCateogry = 'High Returns';
```

CategoryName	SubCategoryName	Region	TotalReturnQty	ReturnCateogry
Bikes	Road Bikes	Australia	67	High Returns
Accessories	Bottles and Cages	Australia	62	High Returns
Accessories	Bottles and Cages	Southwest	72	High Returns
Accessories	Tires and Tubes	United Kingdom	59	High Returns
Accessories	Tires and Tubes	Australia	98	High Returns
Accessories	Tires and Tubes	France	57	High Returns
Accessories	Tires and Tubes	Canada	88	High Returns
Accessories	Tires and Tubes	Southwest	90	High Returns
Accessories	Tires and Tubes	Northwest	94	High Returns

```
058  SELECT
059      ProductSubcategoryKey,
060      ROUND(AVG(ProductPrice),0) AS AvgPrice
061  FROM Products
062  GROUP BY 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ProductSubcategoryKey	AvgPrice
31	34
23	9
19	9
21	51
14	672
12	644
2	1530
1	1637
10	184
11	87
4	69

Comparing it with Products.

```
064 •  SELECT * FROM Products;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell

ProductKey	ProductSubcategoryKey	ProductSKU	ProductName	ModelName
214	31	HL-U509-R	Sport-100 Helmet, Red	Sport-100
215	31	HL-U509	Sport-100 Helmet, Black	Sport-100
218	23	SO-B909-M	Mountain Bike Socks, M	Mountain Bike Socks
219	23	SO-B909-L	Mountain Bike Socks, L	Mountain Bike Socks
220	31	HL-U509-B	Sport-100 Helmet, Blue	Sport-100
223	19	CA-1098	AWC Logo Cap	Cycling Cap
226	21	LJ-0192-S	Long-Sleeve Logo Jersey, S	Long-Sleeve Logo Jersey
229	21	LJ-0192-M	Long-Sleeve Logo Jersey, M	Long-Sleeve Logo Jersey
232	21	LJ-0192-L	Long-Sleeve Logo Jersey, L	Long-Sleeve Logo Jersey
235	21	LJ-0192-X	Long-Sleeve Logo Jersey, XL	Long-Sleeve Logo Jersey
238	14	FR-R92R-62	HL Road Frame - Red, 62	HL Road Frame

```
WITH AvgProductSubcategory AS (
    SELECT
        ProductSubcategoryKey,
        ROUND(AVG(ProductPrice),0) AS AvgPrice
    FROM Products
    GROUP BY 1
)
SELECT
    ProductKey,
    ProductName,
    ProductPrice,
    AvgPrice
FROM Products p
JOIN AvgProductSubcategory a
ON p.ProductSubcategoryKey = a.ProductSubcategoryKey
WHERE ProductPrice > AvgPrice
ORDER BY ProductPrice DESC;
```

ProductKey	ProductName	ProductPrice	AvgPrice
356	Mountain-200 Silver, 46	2071.4196	1637
358	Mountain-200 Black, 38	2049.0982	1637
360	Mountain-200 Black, 42	2049.0982	1637
362	Mountain-200 Black, 46	2049.0982	1637
580	Road-350-W Yellow, 40	1700.99	1530
581	Road-350-W Yellow, 42	1700.99	1530
582	Road-350-W Yellow, 44	1700.99	1530
583	Road-350-W Yellow, 48	1700.99	1530
291	HL Mountain Frame - Si...	1364.5	644
292	HL Mountain Frame - Si...	1364.5	644
299	HL Mountain Frame - Bl...	1349.6	644
300	HL Mountain Frame - Bl...	1349.6	644
437	HL Road Frame - Black,...	1301.3636	672
439	HL Road Frame - Black,...	1301.3636	672
441	HL Road Frame - Black,...	1301.3636	672
443	HL Road Frame - Black,...	1301.3636	672
238	HL Road Frame - Red, 62	1263.4598	672
241	HL Road Frame - Red, 44	1263.4598	672
244	HL Road Frame - Red, 48	1263.4598	672