SUMMARIZE

- SUMMARIZE() function used to create summary table based on specific expression.
- You can include your expressions to calculate values for each group in the resulting summary table.
- It 4 different parameter. First is table expression you want to summarize.
- Groupby_columnname – columns which you want to group the data.
- Name- name of your calculated column in the summary table
- Expression – the expression to calculate for grouping.

Karthimeena

SUMMARIZE

- In the below code, grouped the table by calculating sales amount for each products in a table .

```
1 Summary_by_Product = SUMMARIZE(Sales, Sales[ProductKey], "Sales", SUM(Sales[Sales Amount]))
```

ProductKey	Sales
349	$1,254,722.7326
350	$1,365,852.3783
351	$1,234,276.0309
344	$1,291,868.701
345	$1,186,494.8505
346	$1,217,210.3605
347	$1,019,657.0012
229	$12,603.2548
235	$13,670.3496
218	$6,060.3884

Karthimeena

SUMMARIZE

- In the below code, grouped the table by calculating sales amount for each products and reseller in a table .

```
Summary_by_Product = SUMMARIZE(
                    Sales,  Sales[ProductKey],Sales[ResellerKey],
                    "Sales", SUM(Sales[Sales Amount]))
```

ProductKey	Sales	ResellerKey
484	$9.54	492
484	$47.7	167
484	$104.94	24
484	$52.47	61
484	$28.62	77
484	$28.62	328
484	$9.54	59
484	$9.54	138
484	$9.54	444
484	$19.08	480
484	$66.78	479

Karthimeena

GROUPBY

- GROUPBY() function used to create groups based on the values of one or more columns in a table and this allow you to perform calculations on the group.
- It returns a table, that containing the groups along with aggregated values of each group.
- It 4 different parameter. First is source table from which you want to create groups.
- Group_by_column – The Column which you want to create group.
- Name- name of your aggregated column
- Expression – Aggregation or calculation to be performed on the group.

Karthimeena

GROUPBY

```
Sale_by_Category = GROUPBY(Sales,
                    'Product'[Category],
                    "Total Sales",
                    SUMX(CURRENTGROUP(), 'Sales'[Sales Amount])
                    )
```

Source table name

Group by column

Aggregated column name

Aggregation type

Product_Category	Total Sales
Bikes	$94,620,526.2077
Accessories	$1,272,057.8878
Clothing	$2,117,613.4491
Components	$11,799,076.6584

- In the example, GROUPBY() function creates groups based on the category column from Product table and calculates the total sales amount for each category in the product table.

Karthimeena

GROUPBY

- When using ‘Groupby’ in DAX, aggregation functions like SUM, MAX should operate within the context of the current group.
- To define the current group, here CURRENTGROUP() function used inside the SUMX().

Karthimeena

GROUPBY

```
Sale_by_Category_city = GROUPBY(SALES,
                                'Date'[Fiscal Year],
                                'Product'[Category],
                                 "Total Sales",
                                 SUMX(CURRENTGROUP(), Sales[Sales Amount])
                               )
```

Total Sales	Date_Fiscal Year	Product_Category
$28,544,881.6158	FY2019	Bikes
$22,590,983.47	FY2018	Bikes
$43,484,661.1219	FY2020	Bikes
$36,814.8464	FY2018	Accessories

- In this example, groups are created based on two different columns from different tables.

Karthimeena

TOPN

- TOPN() function used to get the top or bottom N rows from the result set based on specified ordering criteria.
- It returns top/bottom rows in table.
- It 4 different parameter. First is number of rows to return.
- Table – table or virtual table from where to extract the specified number of rows.
- OrderBy_Expression- Values used for ordering the rows.
- Order - Sorting order either ASC or DESC.

TOPN

```
1 Top5_Products_byOrder = TOPN(10, 'Product', [Total_Order_Qty], DESC)
```

Product	ProductKey	Standard Cost	Color	List Price	Model
Bike Wash - Dissolver	484	$2.9733	NA	$7.95	Bike Wash
Full-Finger Gloves, L	470	$15.6709	Black	$37.99	Full-Finger Gloves
Patch Kit/8 Patches	480	$0.8565	NA	$2.29	Patch kit
Short-Sleeve Classic Jersey, XL	491	$41.5723	Yellow	$53.99	Short-Sleeve Classic Jersey
Classic Vest, S	471	$23.749	Blue	$63.5	Classic Vest
Water Bottle - 30 oz.	477	$1.8663	NA	$4.99	Water Bottle
Sport-100 Helmet, Red	214	$13.0863	Red	$34.99	Sport-100
Sport-100 Helmet, Black	217	$13.0863	Black	$34.99	Sport-100
AWC Logo Cap	225	$6.9223	Multi	$8.99	Cycling Cap
Sport-100 Helmet, Blue	222	$13.0863	Blue	$34.99	Sport-100

- In this example, TOPN() fetches the top 10 products based on their ordered quantity from the customer's transactions.

Karthimeena

TOPN

- Here is a scenario to GET the top N product category name based on their sales.

```
Top5_Products_byOrder = TOPN(3,
                        VALUES('Product'[Category]),
                        [Total_Revenue],
                        DESC)
```

Category
Clothing
Components
Bikes

- There is no category table for products, so we are creating a virtual table using VALUES() .

Karthimeena