Data Insights Mystery: Transforming Analysis into Action

Tasks overview:

1. Connect to the data source and get the Fact Company sales\_2017 CSV file.
2. Clean, transform, and load the data.

* Shape and transform tables: remove unwanted columns, remove the first 2 blank rows (‘Remove Rows’ from the home ribbon), and use the first row as the header.
* Make user-friendly naming conventions: remove unnecessary prefixes and signs.
* Resolving the inconsistencies and nulls: use ‘replace values’, change data types, remove duplicates, and filter off the blank values.
* Evaluate and transform column data types: filter out NA values from the columns to detect correct data types. Then select all columns ctrl+A, and from the Transform ribbon select ‘Detect datatype.
* Append tables: add 2 more tables named Sales 2018 and Sales 2019. Do the transformations we have done in the Sales 2017 table, to make sure the column names are the same.
* Merging tables: we merge a table called Sales details; the common column is the order ID.
* Pivot and unpivot table: we have another table called Cost table in CSV format, get that CSV file. Unpivot the 2021 and 2022 columns and rename them as year and cost amount. Then pivot the Type column with the cost amount column.
* Conditional column: add a column named Month number with month numbers.
* Data Profiling: This will help us to understand how our data is structured, like to find some anomalies, value distribution, etc. It contains Column distribution, column profile, and Column quality. It's under the view tab.
* Files from folders: Get data<more<Folder<Browse.

3. Model the data

* Creating relationships: We make relationships in the Model View. Here we are focusing on the Stores table, Products table, and the Sales table. Bring the Sales table in the middle because it’s the fact table. One relationship will be autodetected by Power BI, here an auto relation be detected between the sales table and the sales details table. Since we don’t want that relationship in our report, we deleted it. Hide the sales details table as well. Hide all other unwanted tables. Now drag and drop store\_id from the sales table to the store's table. Now a relationship has been created between 2 tables using the foreign key of the Fact table and the primary key of the dimension table. Similarly, drag and drop product\_id from the products table to the products\_id of the sales table.

To see this in the report view, drag Type from the products table, and order\_id in the sales table(change the aggregation to count). Using this we get how many orders we got in every of these product types.[screenshots 7 and 8]

* Common date table: It is a best practice to always add a date dimension table in our data model, this will help us to analyze the data more precisely. Add a new table option, and write DAX function: Date table = CALENDERAUTO(). We can add new columns to this, e.g.: Month Number = Month(DateTable[Date]).

Or we can add more columns including day, month, quarter, and year using the following Dax formula:

DateTable = ADDCOLUMNS(CALENDAR(DATE(2018,1,1),DATE(2018,12,31)),

"MonthNo", MONTH([Date]),

"MonthName", FORMAT([Date],"MMMM"),

"Quarter", QUARTER([Date]),

"Year", YEAR([Date]),

"Day", DAY([Date]),

"WeekdayNum", WEEKDAY([Date]),

"WeekdayName",FORMAT([Date],"DDDD"))

[Screenshot9]

* Mark as DateTable: Select the Date table<Mark as Date Table<Select Date column. When we do this the built-in date table(hidden date table by Power BI) will be removed. This will make Data size smaller and better performance.
* Setting Hierarchies: Right-click on the Year column in the Date table< create hierarchies< rename it as Date Hierarchies. We also want the Quarter included. Right-click on the Quarter column < add to hierarchy<date hierarchy, we have added the Day also into the Date hierarchy in a similar way. [screenshot 10]
* Data Granularity or the level of detail: we need to connect the costs table with the Date table. But when we go to the table view of the costs table the granularity is based on the year and month, while in the date table, the granularity is based on the year, so the relationship is not possible. In this case, we need to create an additional column with month and year. Open the Query editor< select costs table< select the columns year and month no< add columns< columns from examples< from selection<rename the column as year month.

Make the column in the Date table using the DAX formula: “Year-Month”, FORMAT([Date],” YYYY-M”))

Now we have the same granularity in both tables. Now go back to the Model view and make the relationship between the date table and costs table using the yearMonth.[screenshot 11]

* Decrease the model size by reducing the granularity: we can reduce the granularity by grouping the data. Go to the Query editor< select the table Sales<select the column order date<add column<Date<year & month. Now we have new 2 columns year and month. Now we wanna a group, go to the transform ribbon< select Group by< advanced< select year and month< we can add the aggregation there. This reduces the data size.
* Role-playing dimensions: sometimes we require multiple relationships between the same two tables. In this situation, we can make inactive relationships. [screenshot12]
* Create model calculations by using DAX: we can add additional calculations with measures and calculated columns using DAX

1. Calculated columns: in the sales table, we need to create a revenue column(sales\*price). Select sales table< select new column< in the bar write the DAX formula “ Revenue calculated = sales[price] \* sales[sales] < press enter. Our new column Revenue Calculated is created. It calculates row by row.

Disadvantage: it increases the data size [screenshot 13]

1. Measures: it will not be physically available in the table, so the advantage is it does not increase the data size. It calculates at the aggregation level. Here we need to create a measure in the sales table. Select the sales table < New measure < in the bar, write “ sum of sales = SUM(sales[sales]) < click enter. A new measure is now created. [screenshot 14]
2. Iterator functions: it is also called x functions. This function calculates in the row-based level and then aggregates.

“Revenue measure = SUMX(sales, Sales[sales] \* sales[price])

[screenshot 15].

1. Calculate: calculate function modifies or overwrites filter context of calculations. Here we are going to limit our revenue measure to the year 2018. Select sales table < new measure < in the tab write, Revenue for 2018 = CALCULATE ([revenue measure], Datetable[year] = 2018). [screenshot 16]
2. Filter: here we include a Filter function in the Boolean expression. It is a table function. Here we need the 2018 year, so it filters rows in the year 2018 only.

Revenue for 2018 = CALCULATE([Revenue measure], FILTER(DateTable, DateTable[Year] = 2018 )) [screenshot 17].

1. All function: like the filter function an All function is also a table function. Therefore we can use it in the calculate function as a filter. All function basically ignores all filter functions in a given table or column. Select the sales table < select Revenue for 2018 < Revenue percentage = [Revenue measure] / CALCULATE([Revenue measure], ALL(DateTable))

This function will be useful in a situation where we want to calculate percentages.[screenshot 18]

1. Time intelligence: we need to use the year-to-date time intelligence in DAX. So let's create a new measure in the sales table. Write DAX formula: Year-to-date revenue = CALCULATE ([Revenue measure], DATESYTD(DateTable[Date])). Drag revenue measure and date into the field, use a line chart. Now add our newly created year-to-date revenue function into the value pane, year-to-date will be shown in the figure.

The starting date and end date may differ in some companies, for example: In fiscal year. In this situation, we can rearrange the DAX: Year-to-date revenue = CALCULATE ([Revenue measure], DATESYTD(DateTable[Date], “6/30”))

[screenshot 19]

1. DateAdd function: we want to see how this has been different in the previous year, ie, we want to compare the result with the previous year. So the DAX formula will be: Previous revenue = CALCULATE([Revenue measure], DATEADD(DateTable[Date], -1, Year)) [screenshot 20]
2. Semi-additive measures: when we need to aggregate values up to a certain limit, we use this function. In this situation, we use Date hierarchy instead of date.

DAX formula: Products in stock = CALCULATE(SUM (sales [stock]), LASTDATE(DateTable[DATE]))

1. Statistical functions: DAX functions are; Min Price = MIN (sales[price]);

Median Price = MEDIAN(sales[price]); variance of price = VAR.P(sales[price]); Average = AVERAGE (sales[price])

1. Optimize model performance: we can achieve it through 3 ways; DAX measures, small model size, and low level of cardinality.

4. Visualize and analyze the data

* Area chart: we want to display categorical data. Here we want to show how much revenue we have across different product categories. Drag ‘category & subcategory column from the Products table. But here our category and subcategory column is separated by a slash delimiter, we want to separate it into 2 separate columns by a delimiter. So, open the query editor < select Product table< select category and sub-category column< select ‘split column by delimiter’< rename both columns. Go back to the power bi desktop. We need to make a relationship between the sales table and the products table, for this go to the model view< make the relationship between products and sales using the ‘product\_id’. Jump back to the report view< select Area chart < drag category to the axis < drag revenue measures to value pane< drag year to the legends section [screenshot 22].
* Combo chart and treemap: select the combo chart< drag category to the axis & revenue measure to the column values < year to the column series. Now we can see there is a correlation between sales and revenue. Ie, if there is a high sales in a certain category then revenue will also be high.

Now we can make a tree map. It saves lots of space in our chart. drag revenue to the values< drag categories to group< drag subcategory to the Details section.

[screenshot 23]

* Card and gauge visual: we can use the card visual to show an important whole number, for example here revenue measure. Then convert the whole number to currency dollar.

Next, select the gauge visual< drag average to the value. Now we are going to create a measure ‘target value = 16’ < drag target value to the field. [screenshot 24]

* Custom tooltips: we need to create a new page to add custom tooltips. Create a new page < in the format pane select page information< turn on the tooltip< select the type as tooltip, this makes the page into a tooltip.

Select the stacked bar chart < drag category to the axis < Revenue measure to the values

Rename the page as ‘tooltip page’. Now we want this tooltip to appear in other visuals, so go back to the overview page. we are going to make this tooltip attached to the area chart. Select the area chart < in the formatting pane < select tooltip < set the type as ‘report page’<select the page as ‘tooltip page’. When we hover over to the area chart, we can now see the tooltip visual accordingly. [screenshot 25]

* Slicing and filtering: the slicer visual allows the end user to interactively apply some quick and easy filters on the report. The best place to place the slicer visual is on the left side of our report. Here we are going to filter by stores in a certain state. Drag the state column to the slicer visual. Now when we click on a certain place the data in our page will change accordingly. [screenshot 26]
* Sync slicers: we can use the same slicer visual working on multiple pages. To achieve this, copy and paste the slicer visual on another page < open the view ribbon < select sync slicer < in the sync slicers pane check the pages that contain slicers. Now when you select a particular place in the slicer visual it will be selected in the other synced page also.
* Drillthrough & drilldowns: this tool allows the end user to see the data only if they choose to see it. Here we are going to apply this to the visual called ‘Revenue measure and sales by category’. First, create a new page and name it ‘details page’. Select a table visually and drag order ID, order date, revenue measure, state, and city. These are the details that we are interested in. Add revenue measure as a card visual as well. Add category column to the drillthrough pane Go to the overview page < right click on the ‘snacks & sweets’ bar < select drillthrough < details page. we can see all of the orders from scacks and sweets. [screenshot 27]

Drilldowns enables us to use hierarchies and visuals. Here we use the bar chart. We want to see the subcategories also. So drag the subcategory into the axis. Now controls will appear in the visual

* Adding bookmark and buttons: we are going to add a ‘weekday name’ bookmark to our page no.1. drag the weekday name column to the field. But the column is not sorted in the right way. For this select the column < click Sort by column from the column tools ribbon. Now Sunday is the starting day. We select Sunday and Saturday only as a bookmark. To achieve this we need to create a button first. Click Insert tab < click buttons < select blank button. Now a blank button appears on the page. Now we are going to name the button. Select the format button pane < select text < name ‘show weekends. Now select the bookmarks pane under the view ribbon< select the selection section < in the selection pane hide the slicer visual. Now in the bookmarks pane < click add bookmark < name it as ‘show weekends’< click on the 3 dots < click update (make sure that the data option is selected), now the bookmark is activated. select the action in the same pane and turn it on < expand the action button < select type as bookmark < select ‘show weekends. Now when we click on the ‘show weekends' button we will see the visuals accordingly. Make a ‘show all’ button in the same way which shows data of all days in a week. [screenshot 28]
* Page navigation: it's an action we can link to a button. Create a new page and name it‘ page navigation’, next we need to create a logo, select insert ribbon< select image< browse an image < select a blank button from the insert ribbon<name the button as ‘overview’< in the format button pane select an action and turn it on < select type as ‘page navigation’< select the destination page as ‘overview’. now it's created, when you click on the button it will lead to the overview page.[screenshot 29]

Add a ‘Back button’ on the overview page. [screenshot 30]

* Edit interactions: through this, we can make the visuals in our report more interactive. Ie, when we select a field in one visual, the other visuals also should change accordingly. Select format ribbon < select Edit interactions [screenshot 31]
* Analyze feature: the built-in analyze feature in Power BI is enabled only in a few visuals like bar charts as well as line charts. This helps us to understand where a certain distribution is coming from. Ie, why certain values are high, and why certain values are lower. Right-click on visual somewhere < analyze < explain increase/decrease. [screenshot 32]
* Identify outliers: we use a scatter plot to find outliers. Drag products into the details pane < drag revenue measure to the y-axis < drag sales to the x-axis (aggregation should be sum). Now we can see the product ‘white chocolate bar’ is an outlier in terms of revenue and sales. Now we want to make these outliers stand out more. For this drag the column category into the legend field. And now we can see most of these outliers are from a category named ‘snacks & sweets’. Add revenue into the size pane, this results in the outliers standing out even more.

Now we can label outliers by creating certain groups. Make a column in products, DAX expression is: Outliers = IF ([Revenue measure] > 5\* AVERAGE (Products, [Revenue measure]), “outlier”, “no outlier”).

If a particular product has a revenue that is 5 times greater than the average revenue, we wanna label it as an outlier.

Click a table< drag the outlier column into the table < drag products into the table (set the aggregation to count). Now we can see only 14 products that are outliers

[screenshot 33]

* Create groups, bins, and clusters: in the date table we going to make a group for week names. Ie, for weekends and weekdays. Right-click in the weekday name column < select new group < select Sunday and Saturday < select group < name it as weekend < select rest of the days and group it and rename it as week < name it as weekend flag< ok.

In binning, it applies to numerical fields only. In sales table < select price column < right click< new group < select group type as bins < bin type as a number of bins< ok.

* Forecast feature: this is available in the line chart. Let's check how our sales/revenue in the future using this tool. add a new page< drag a line chart < drag date to the axis < drag revenue to the value < go to the analytics pane < select ‘forecast’ < select forecast length and set it to 21 (ie, it will show for 21 days) < set ignore last as 7 points (ie, the last 7 points will be ignored, since it is not relevant)< set confidence interval as 75%< turn on the zoom slicer. [screenshot 34]