

Asces Interactive Dashboard Project (PostgreSQL and Power Bi Project)

Name: Kevin Dinh (Quyet Xuan
Dinh)

1) Projects overview

a) Project vision

- For this project, we will simulate a real business requirements that need us to complete
- Project source:
https://absentdata.com/data-analysis/end-to-end-data-analysis-project/#google_vignette

b) Business overview

- Asces Sound is a leading innovator in the audio technology industry, dedicated to providing high quality audio solutions for professionals and enthusiasts alike. With a focus on exceptional sound clarity, advanced engineering, and user-friendly designs.

c) Business request

- Constructs a high-level product analytics dashboards that provides key product performance metrics. This dashboards will support strategic decision-making and allow executives to track preformance trends effectively. What we need in the dashboard are:
 1. Revenue by Country: Top-performing regions with corresponding revenue
 2. Revenue by Date and Year: Comparative trends
 3. Profit and Unit Sales Year-over-Year(YoY) Change: High-level summary of YoY growth
 4. Revenue Breakdown by Discount Band: Distribution of revenue across different discount
 5. Detailed Table View: Revenue and profit details by country and year
And other metrics necessary.

d) Datasets

- Link:
https://absentdata.com/data-analysis/end-to-end-data-analysis-project/#google_vignette
- Direct link:
<https://github.com/KevinDinh03/SQL-and-PowerBI-Project/tree/main/Raw%20datasets>

1) Product_data.csv

- Columns description:
 - + **Product ID:** ID of the product

- + **Product:** Name of the each product
- + **Category:** Type of product
- + **Cost Price:** Cost of the product
- + **Sale Price:** Price to purchase the product
- + **Brand:** Brand name
- + **Description:** Product description
- + **Image url:** Url to the product image

2) discount_data.csv

- Columns description:
 - + **Month:** Month of the discount
 - + **Discount Band:** Band type of each discount
 - + **Discount:** The amount of the discount

3) product_sales.csv

- Columns description:
 - + **Date:** Date of the sale
 - + **Customer Type:** Type of customer
 - + **Country:** Country destination of the sale
 - + **Product:** The product id
 - + **Discount Band:** Band type of the sale's discount
 - + **Units Sold:** Quantity of the product in the sale

2)Join the dataset using Postgresql database

a) Overview about the datasets.

- There are three datasets that I have to join.

1) discount_data.csv

- This dataset stores the discount information of the product
- The columns in this dataset are **Month, Discount Band, Discount**

2) Product_data.csv

- This dataset stores the basic information of each product type
- The columns in this dataset are **Product ID, Product, Category, Cost Price, Sale Price, Brand, Description, Image url**

3) product_sales.csv

- This dataset stores the informations of every product orders from 2022 to 2023
- The columns in this dataset are **Date, Customer Type, Country, Product, Discount Band, Units Sold**

b) Joins the datasets

1) Load the dataset

- For this specific task, I will use the **Postgresql** database to store the datasets and join the data.
- Because Postgresql doesn't have built-in functions that can import CSV files into a sql database, first I have to create the tables corresponding to the columns in the CSV file and then use the COPY function from Postgresql to import the datasets.
- During creating tables, I make sure that the data types are correct to the corresponding columns

```
--Create discount_data table and copy its dataset
CREATE TABLE discount_data (
    "Month" VARCHAR(50),
    "Discount_Band" VARCHAR(50),
    "Discount" INT
);

COPY discount_data
FROM '/Users/xuanquyetdinh/Desktop/Data Analysis Project November 2024 - Copy/discount_data.csv'
DELIMITER ','
CSV HEADER;
```

```
--Create product_data table and copy its dataset
CREATE TABLE product_data (
    "Product_ID" VARCHAR(50),
    "Product" VARCHAR(50),
    "Category" VARCHAR(50),
    "Cost_Price" MONEY,
    "Sale_Price" MONEY,
    "Brand" VARCHAR(50),
    "Description" VARCHAR(250),
    "Image_url" VARCHAR(100)
);

COPY product_data
FROM '/Users/xuanquyetdinh/Desktop/Data Analysis Project November 2024 - Copy/Product_data.csv'
DELIMITER ','
CSV HEADER ENCODING 'WIN1252';
```

- For **product_sales data**, It has a **Date** column that are in the DD/MM/YYYY in the CSV file, and if import that to Postgresql, It will be understand as MM/DD/YYYY which will inaccurate to the original dataset.

- In order to tackle this problem, I create two tables, one as the **product_sales_temp** and the other as **product_sales**.
- I first load the data into the temporary dataset and store the **Date** as **TEXT**.

```
CREATE TABLE product_sales_temp (
  "Date" TEXT,
  "Customer_Type" VARCHAR(50),
  "Country" VARCHAR(50),
  "Product" VARCHAR(50),
  "Discount_Band" VARCHAR(50),
  "Units_Sold" INT
);

COPY product_sales_temp
FROM '/Users/xuanquyetdinh/Desktop/Data Analysis Project November 2024 - Copy/product_sales.csv'
DELIMITER ','
CSV HEADER;

CREATE TABLE product_sales (
  "Date" DATE,
  "Customer_Type" VARCHAR(50),
  "Country" VARCHAR(50),
  "Product" VARCHAR(50),
  "Discount_Band" VARCHAR(50),
  "Units_Sold" INT
);
```

- Then I use the **INSERT INTO** function and use **TO_DATE** function to insert the data from the **product_sales_temp** table to the **product_sales** table
- Because it is first store as **TEXT**, it lets **TO_DATE** function to see the column as text, and re-format it as "DD-MM-YYYY" and store into the **product_sales** table which the column **Date** has been store as **DATE**

```
INSERT INTO product_sales ("Date", "Customer_Type", "Country", "Product", "Discount_Band",
"Units_Sold")
SELECT
  TO_DATE("Date", 'DD-MM-YYYY'),
  "Customer_Type",
  "Country",
  "Product",
  "Discount_Band",
  "Units_Sold"
FROM product_sales_temp
```

2) Join the dataset

- First, I will observe through the three datasets for common attributes to join
- For **product_data** and **product_sales**, there are two columns that store the same value but with different name - **Product_ID** column from **product_data** and **Product** column from **product_sales** - so I will join the two tables first
- For the **discount_data**, the **Discount_Band** column is the same with the **Discount_Band** column from the **product_sales**, but it is not enough uniqueness to

join the the two datasets because each month will have the same discount bands but the amount of discount will be varied differently each month as shown in the discount_data table.

```
Raw datasets > discount_data.csv
```

1	Month,Discount Band,Discount
2	January, none, 0
3	January, low, 6
4	January, medium, 10
5	January, high, 5
6	February, none, 0
7	February, low, 6
8	February, medium, 11
9	February, high, 5
10	March, none, 0
11	March, low, 6
12	March, medium, 4
13	March, high, 15
14	April, none, 0
15	April, low, 3
16	April, medium, 8
17	April, high, 15
18	May, none, 0
19	May, low, 7
20	May, medium, 11
21	May, high, 9
22	June, none, 0
23	June, low, 6
24	June, medium, 4
25	June, high, 7
26	July, none, 0
27	July, low, 8
28	July, medium, 9
29	July, high, 20
30	August, none, 0
31	August, low, 8
32	August, medium, 8
33	August, high, 17
34	September, none, 0
35	September, low, 5
36	September, medium, 7
37	September, high, 12
38	October, none, 0
39	October, low, 9
40	October, medium, 14
41	October, high, 20
42	November, none, 0
43	November, low, 15
44	November, medium, 19
45	November, high, 24
46	December, none, 0
47	December, low, 14
48	December, medium, 18
49	December, high, 20
50	

- In order to solve this problem, I firstly store the two previous datasets as a Common Table Expression (CTE). Then I will use the Date column from the CTE to extract a Year and a Month column for further visualisations and join actions using TO_CHAR.
- Then, I will join the discount_data table to the CTE using Discount_Band and Month column.

```

/*
- Firstly, I will join two tables - "product_data" and "product_sales" - based on the Product_ID
from the product_data table and Product from the "product_sales" table.

- Then, I will create two columns - Revenue and Total_cost - to gain some more insights from
these data

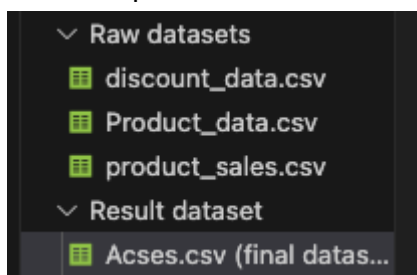
- Next, I will extract two extra columns - Month and Year - So that I can join the aggregated
dataset with "discount_data" table

- To make sure that the further joins are easier, I will store it as a
Common Table Expression (CTE) using the With functions*/
WITH cte as(
SELECT
pd."Product",
pd."Category",
pd."Cost_Price",
pd."Sale_Price",
pd."Brand",
pd."Description",
pd."Image_url",
ps."Date",
ps."Customer_Type",
ps."Country",
ps."Discount_Band",
ps."Units_Sold",
"Sale_Price" * "Units_Sold" as "Revenue",
"Cost_Price" * "Units_Sold" as "Total_cost",
TRIM(TO_CHAR("Date", 'Month')) as "Month",
TRIM(TO_CHAR("Date", 'YYYY')) as "Year"
FROM product_data pd
JOIN product_sales ps
ON pd."Product_ID" = ps."Product")

SELECT *, (1 - "Discount"*1.0/100) * "Revenue" AS "Discounted_Revenue"
FROM cte c
JOIN discount_data d
ON LOWER(c."Month") = LOWER(d."Month")
and LOWER(TRIM(c."Discount_Band")) = LOWER(TRIM(d."Discount_Band"))

```

- After the join has done, I will have a complete dataset with all of the datas required
- Then I export it into a CSV file to further import it into Power BI

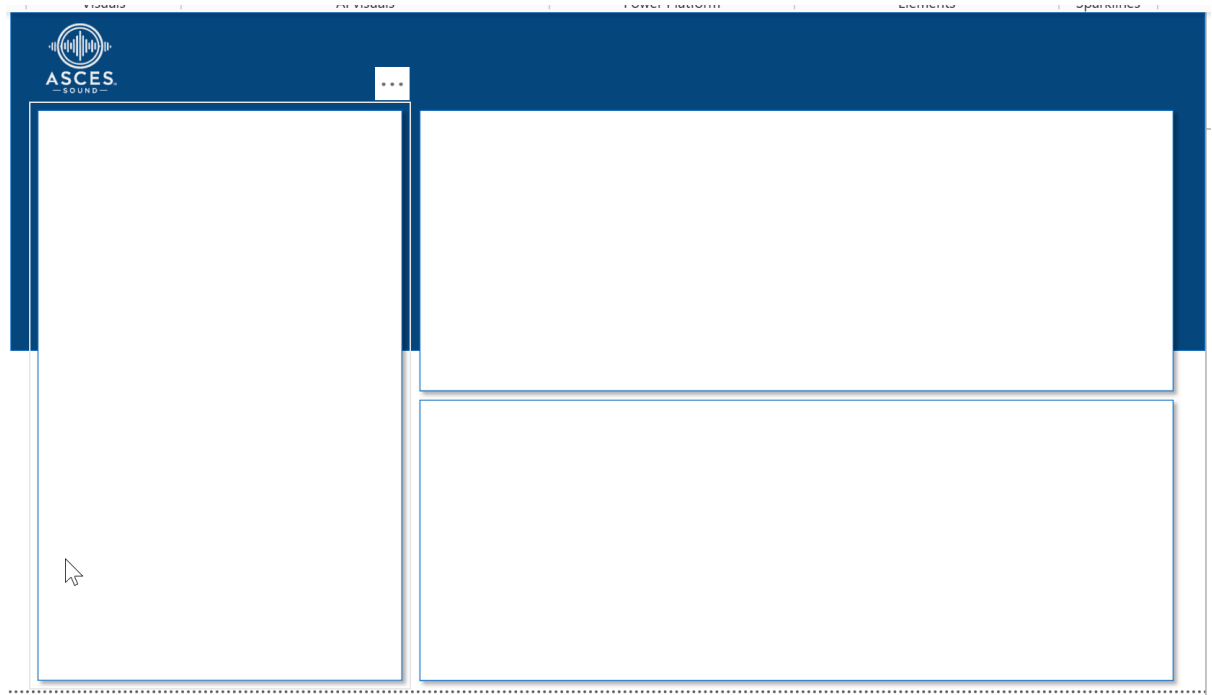


- Note: While there is no need to convert into CSV file and I can access SQL database using Power Bi, because the device I am using is Mac and I am running Power Bi through VMWare Fushion, the best way to do this is to export as a CSV file, upload it on Github and I will use VMWare to download it back for PowerBi visualisations.

3) Visualise data in PowerBI

a) Dashboard's frame

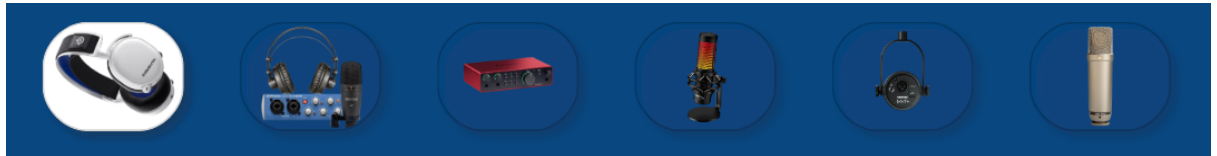
- Firstly for the dashboard, I construct a frame for the dashboards so that areas are easy to visualise.
- I use basic rectangular shape to create the frame so that I can visualise what I would put on there that makes sense
- I format the background to be in a dark blue color and added the logo of the company from resources that have been provided



b) Image slicer

- For deep insights of every product, I use a **Button slicer** for filtering the information of each product and it connects to the **Image_url** field
- The visual of the buttons are replaced by the image of the products for more interactivity
- The buttons are formatted with rounded-rectangle shape with borders and shadows for appealing visuals

- Also, I make sure the color of the button will change when I hover or selected it



- The basic image, description and names of the selected items are shown using Card (new) visualisations in the top-left of the vertical rectangular area
- It is accommodated by two interactive buttons that show selection between the two years. They use the same formats and visualisations as the slicer use for the items but with the **Year** field connection



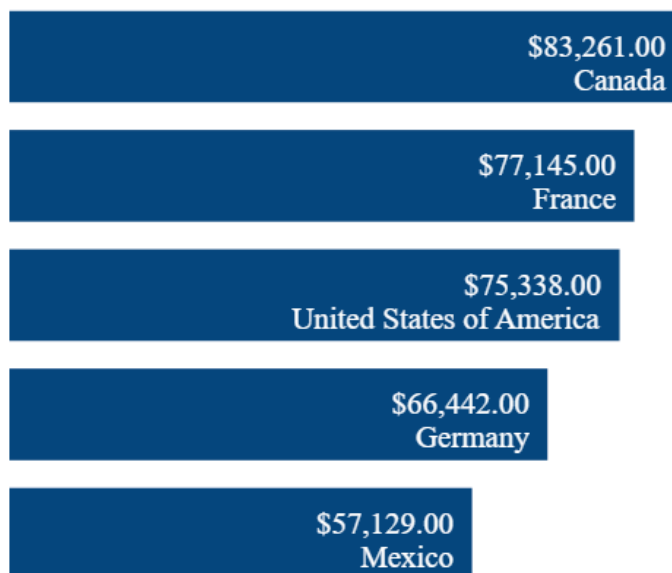
c) Main visualisations

- To meet the business request from Ascens, these are the main visualisations for the dashboards
 - 1) **Sum of Revenue by Country**
 - 2) **Sum of Revenue by Date and Year**
 - 3) **Discount_Band Breakdown**
 - 4) **Customer_Type summary table.**
 - 5) **Profit and Unit Sales Year-over-Year(YoY) Change**
 - 6) **Product tooltip**

1) Sum of Revenue by Country

- For the sum of revenue by each country, I will be choosing a simple **Clustered bar chart**
- The fields that are added to this charts are **Country (Y-axis)** and Sum of **Revenue (X-axis)**
- For this chart, the formats are:
 - + Eliminate all of the axis titles and data values
 - + Change the color to dark blue (Match with the color of the dashboard)
 - + Change the font of the title to **Times New Roman** and make it **Bold**
 - + Format the data labels and country names to be directly on to the bar
 - + No background

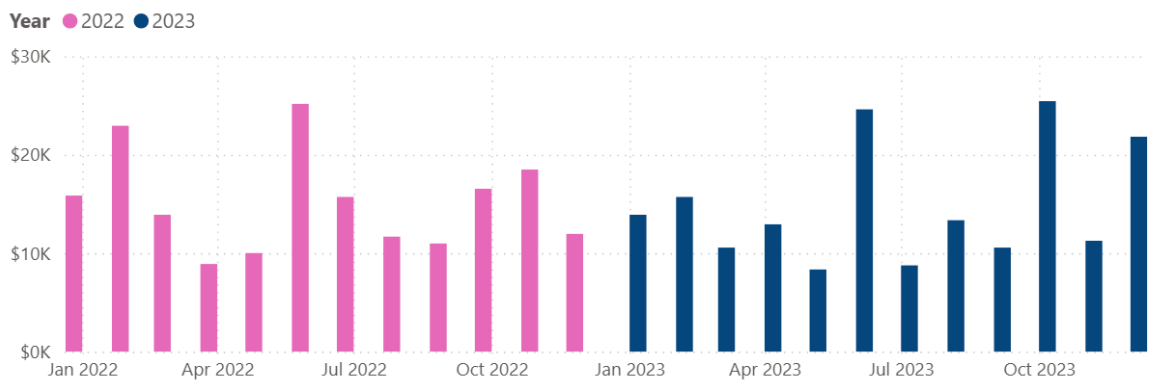
Sum of Revenue by Country



2) Sum of Revenue by Date and Year

- For the Sum of Revenue by Date and Year , I will use a **Clustered column chart**.
- The fields for this chart are **Date (X-axis)**, **Revenue (Y-axis)** and **Year (Legend)** for clarification.
- For this chart, the formats are:
 - + Eliminate all of the axis titles, keep the data values
 - + Change the color of the columns to pink and dark blue (Match with the color of the dashboard and more diversity to the two years)
 - + Change the font of the title to **Times New Roman** and make it **Bold**
 - + No background

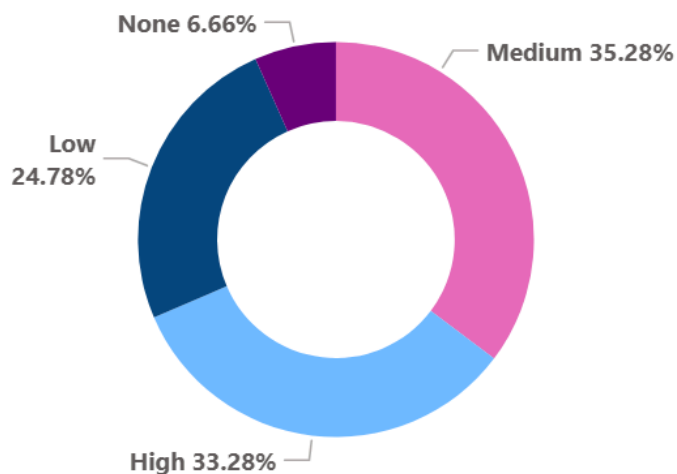
Sum of Revenue by Date and Year



3) Discount_Band Breakdown

- For the **Discount_Band Breakdown**, I will use a **Donut chart**
- The fields for this chart are **Discount_Band (Legend)** and **Revenue (Values)**
- For this chart, the formats are:
 - + Eliminate legend notations
 - + Change the color of the slice so that the color palette is in a more suitable combinations to the overall theme of the dashboard
 - + Change the font of the title to **Times New Roman** and make it **Bold**
 - + Change the title name to **Discount_Band Breakdown**
 - + Change the detail labels content to **Category, percent of total**
 - + Make the detail labels **Bold**.
 - + No background

Discount_Band Breakdown

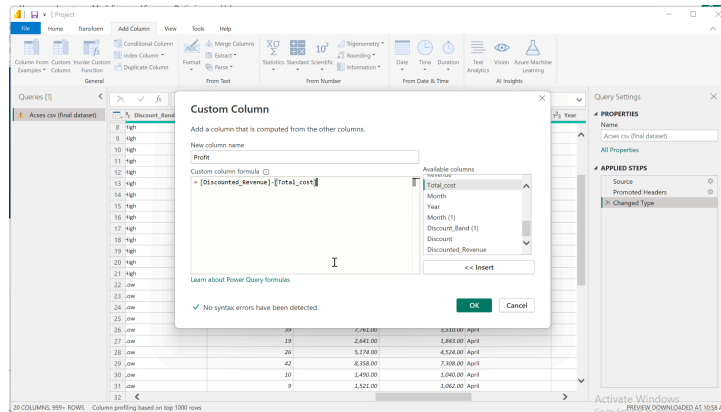


4) Customer_Type summary table

- For all the information of different **Customer_Type**, I will use the **Matrix** visualisation



- Before adding value to the chart, I will first add a **Profit** column to the dataset using Power Query Editor.
- The equation will be: $\text{Discounted_Revenue} - \text{Total_cost} = \text{Profit}$

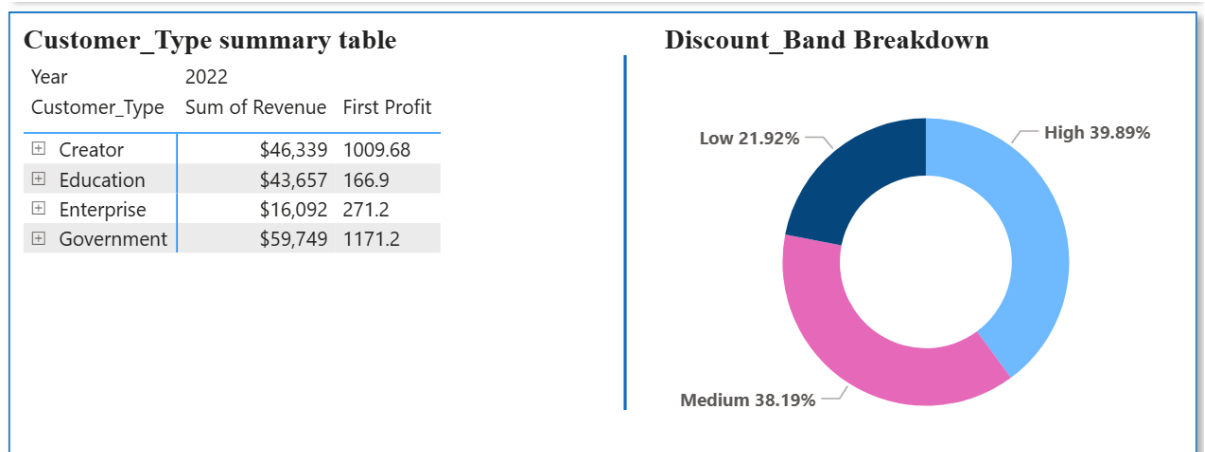


- The fields for this chart are:
 - + Rows: **Customer_Type, Product**
 - + Columns: **Year**
 - + Values: **Revenue and Profit**
- For this chart, the formats are:
 - + Turn off the **Column subtotals** and **Row subtotals**
 - + Add the title **Customer_Type summary table**
 - + Format the title's font to be in **Times New Roman** and in **Bold**

Customer_Type summary table

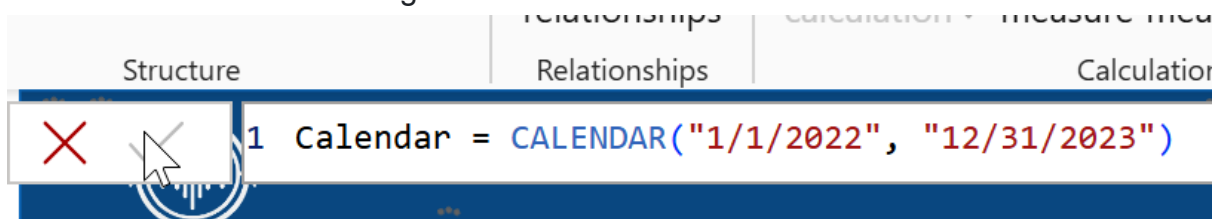
Year	2022		2023	
Customer_Type	Sum of Revenue	First Profit	Sum of Reven	
Creator				\$32,6
Education	\$18,706	1683.85		\$36,6
Enterprise	\$59,899	0		\$43,1
Government	\$99,699	1024.52		\$109,4
Small Business	\$53,332	1010.4		\$24,2

- This table needs a separator with the donut chart so I use the **Line** shape, rotate it vertically and change to dark blue to make it suitable for the dashboard



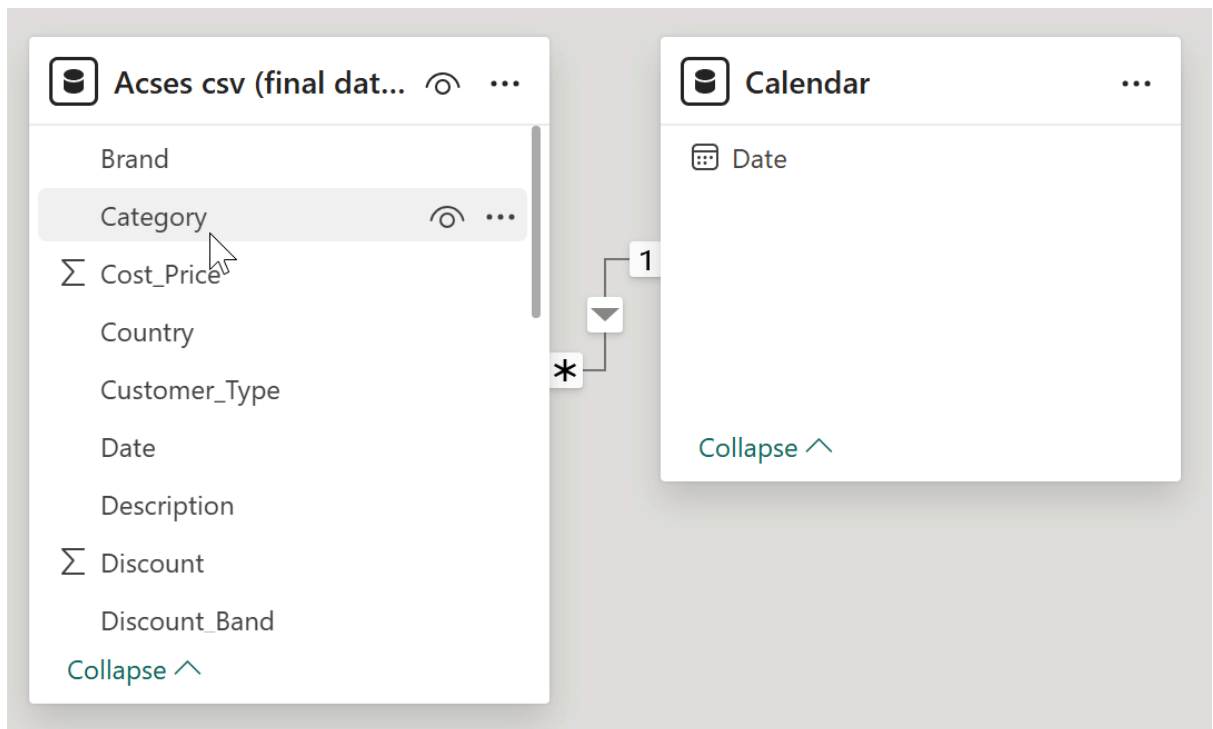
5) Profit and Unit Sales Year-over-Year(YoY)

- To do the Profit and Unit Sales Year-over-Year (YoY), I will use DAX functions to create new measure named **Profit_YoY** and **Unit_Sales_YoY**
- To be able to use the functions, firstly a table that contains continuous dates must be created.
- I created a new table named **Calendar** that contains continuous date from 1/1/2022 to 12/31/2023 using the **Calendar** functions



- Then I connect the relationships between the main table and the **Calendar** table. The relationship will be **one to many** between the **Calendar** table and

the main table, also the **direction** is single



- After finish creating the relationship, I use DAX functions to create new measures:
 - + **Profit YoY** = var last_year = CALCULATE(SUM('Acscs csv (final dataset)'[Profit]), DATEADD('Calendar'[Date], -1, YEAR))
return
(SUM('Acscs csv (final dataset)'[Profit])-last_year)/last_year
 - + **Units Sold YoY** =
var last_year = CALCULATE(SUM('Acscs csv (final dataset)'[Units_Sold]), DATEADD('Calendar'[Date], -1, YEAR))
return
(SUM('Acscs csv (final dataset)'[Units_Sold]) - last_year)/last_year

```
1 Units Sold YoY =  
2     var last_year = CALCULATE(SUM('Acscs csv (final dataset)'[Units_Sold]), DATEADD('Calendar'[Date], -1, YEAR))  
3     return  
4     (SUM('Acscs csv (final dataset)'[Units_Sold]) - last_year)/last_year  
5  
6
```

```
1 Profit YoY =  
2     var last_year = CALCULATE(SUM('Acscs csv (final dataset)'[Profit]), DATEADD('Calendar'[Date], -1, YEAR))  
3     return  
4     (SUM('Acscs csv (final dataset)'[Profit]) - last_year)/last_year
```

- After create the new measures, I discover that there is an issue with the format of the **Profit** field, leading to the **Card(new)** visualisation unable to visualise the new measures. So i go to Power Query Editor, and change the

data type of the **Profit** column into **Whole Number**. And that fix the problem

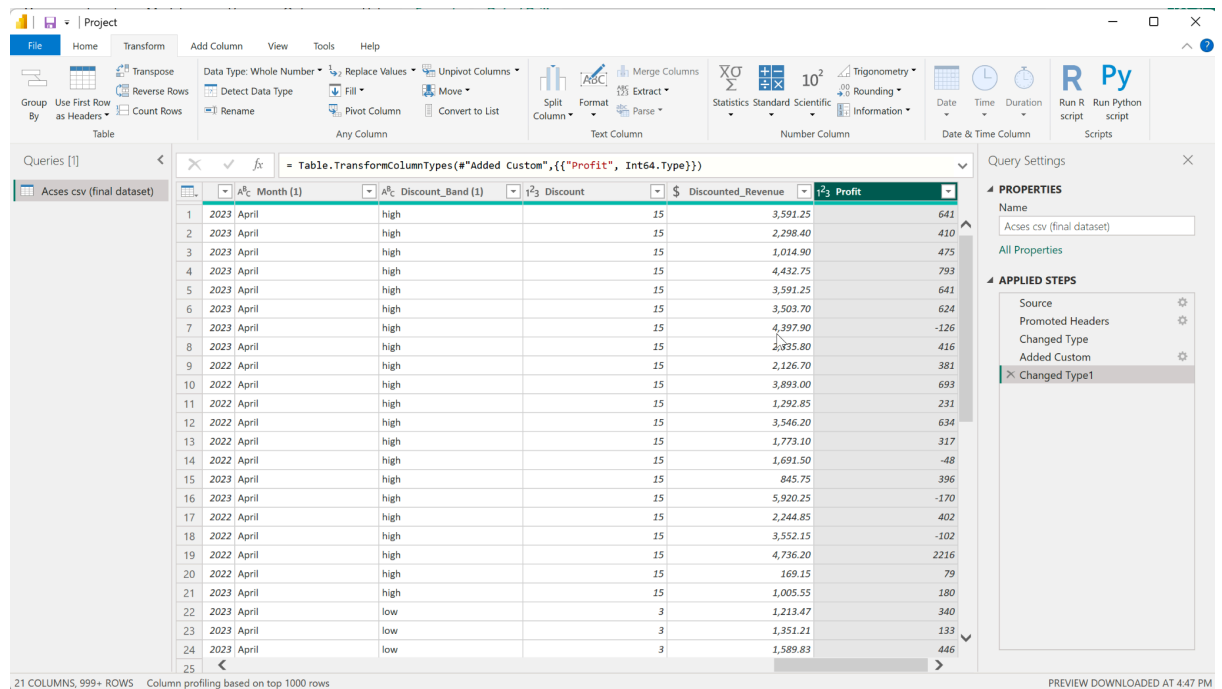


Table: TransformColumnTypes(#"Added Custom",{"Profit", Int64.Type})

	Month (1)	Discount_Band (1)	Discount	Discounted_Revenue	Profit
1	2023 April	high	15	3,591.25	641
2	2023 April	high	15	2,298.40	410
3	2023 April	high	15	1,014.90	475
4	2023 April	high	15	4,432.75	793
5	2023 April	high	15	3,591.25	641
6	2023 April	high	15	3,503.70	624
7	2023 April	high	15	4,397.90	-126
8	2023 April	high	15	2,935.80	416
9	2022 April	high	15	2,126.70	381
10	2022 April	high	15	3,893.00	693
11	2022 April	high	15	1,292.85	231
12	2022 April	high	15	3,546.20	634
13	2022 April	high	15	1,773.10	317
14	2022 April	high	15	1,691.50	-48
15	2023 April	high	15	845.75	396
16	2023 April	high	15	5,920.25	-170
17	2022 April	high	15	2,244.85	402
18	2022 April	high	15	3,552.15	-102
19	2022 April	high	15	4,736.20	2216
20	2022 April	high	15	169.15	79
21	2023 April	high	15	1,005.55	180
22	2023 April	low	3	1,213.47	340
23	2023 April	low	3	1,351.21	133
24	2023 April	low	3	1,589.83	446

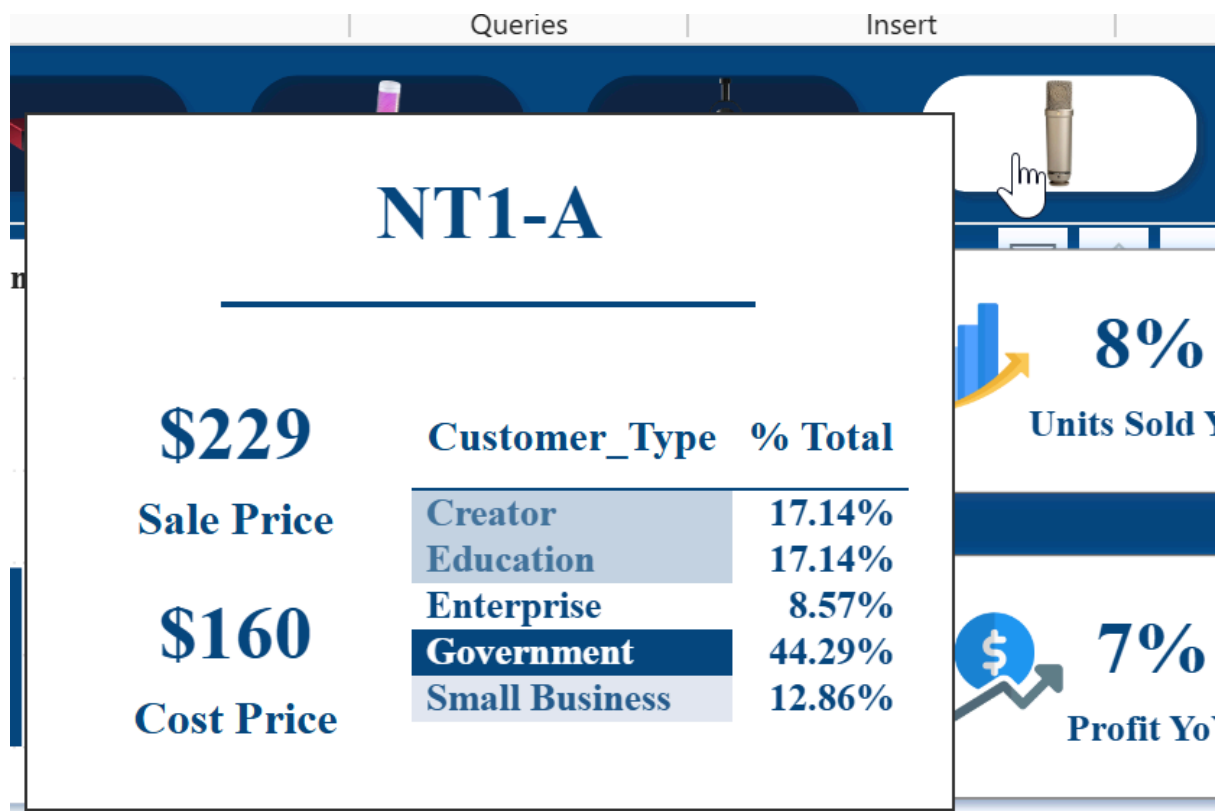
- After that, I format the **Units Sold YoY** & **Profit YoY** in the **Card(new)** visualisations. I added images for more appealing visualisations, and format its word sizes and color so that it fits to the color palette of the dashboards, and the numbers can clearly be seen
- Because the **Units Sold YoY** & **Profit YoY** measure the changes within one year, if the year filter is selected at 2022, there will be no data. So by using **Edit interaction**, I disable the year filter's interactions with these two metrics and only allow others to interact with them.



- All of the visualisations are almost done, the only last thing is the tooltip.

6) Product tooltip

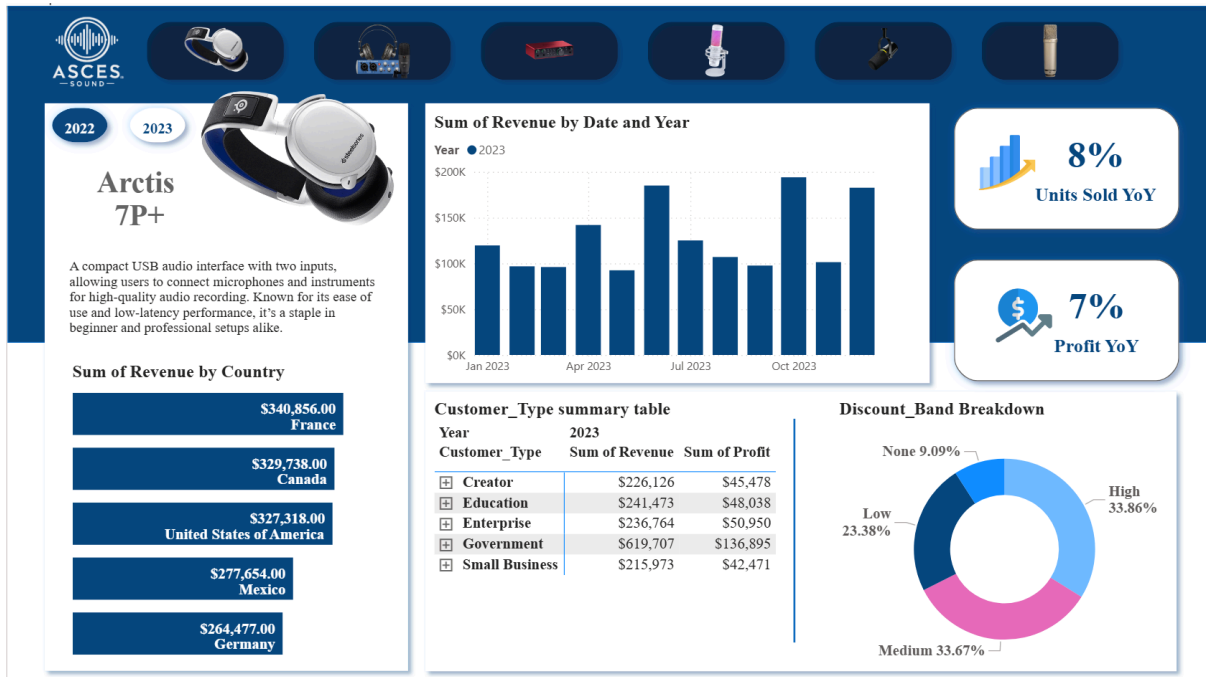
- For the tooltip, I want to add more details for each products that have not shown on the visualisations.
- The informations on the tooltip is product name, sale price, cost price and different percentage of customer types on those products
- The fields that are connected to the tooltip are **Sale_Price**, **Cost_Price**, **Product** and **Customer_Type**
- I format the tooltip on a different worksheet and then connect it to the **Button slicer** of **Product** from the very beginning
- The format of the tooltip is simple but clear and straight-forward. It has a gradient color change to highlight the smallest to the largest group of customers.



- After finishing the tooltip, I take one more time to rearrange the whole dashboard, format the text, shape, color and position of every visualisations so that the dashboard will be clear, concise but appealing for business application.
- Finally, the dashboard is finish.

d) Overview of the dashboard

1) Goal



- This dashboard is built to meet the business request of a high high-level product analytics dashboard that provides key product performance metrics.
- It aims to support strategic decision-making and allow executives to track performance trends effectively.

Key product metrics that are shown in this dashboards:

- 1) Sum of Revenue by Country**
 - 2) Sum of Revenue by Date and Year**
 - 3) Discount_Band Breakdown**
 - 4) Customer_Type summary table.**
 - 5) Profit and Unit Sales Year-over-Year(YoY) Change**
 - 6) Product tooltip**
- Sum of revenue of each country
 - Sum of revenue in every month in 2022 and 2023
 - Percentage of each discount type over the total revenue
 - Product, sum of revenue and sum of profit for each customer type.
 - Percentage change of Profits and Units Sales from 2022 to 2023
 - Tooltip that have information of each product such as sale price, cost price, and percentage of each customer types that buy each product type.

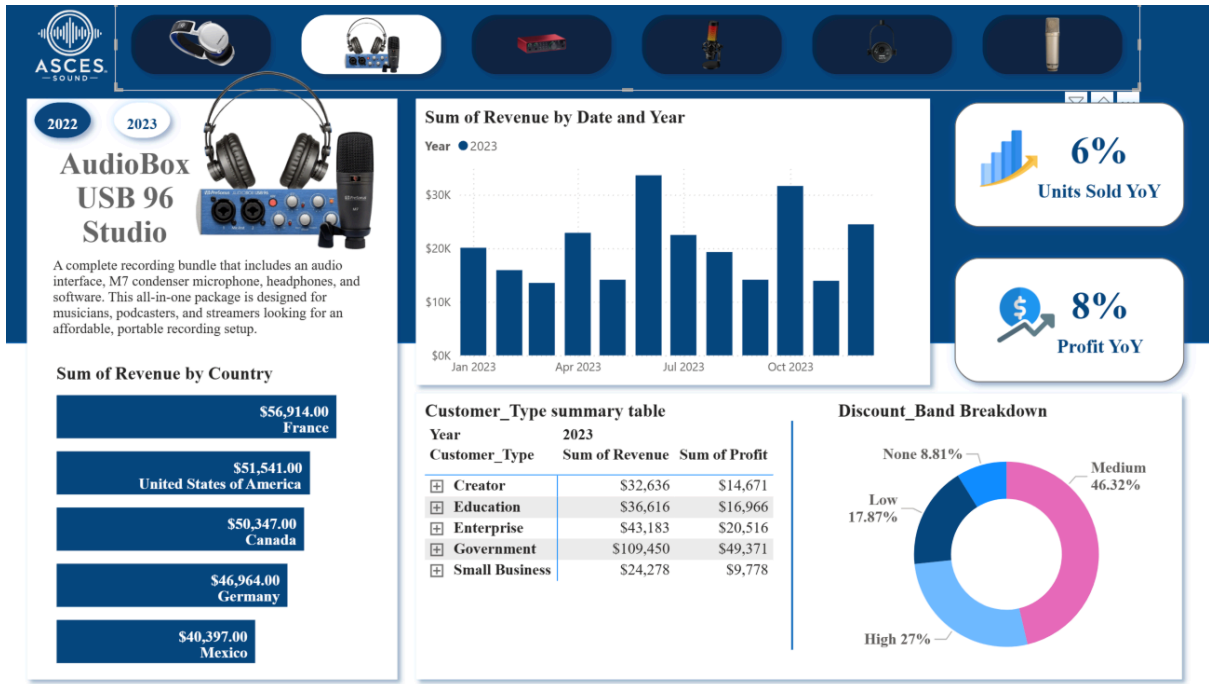
2) Interactive features

- Product button slicer:

- + Product slicer formats as buttons that filter the dashboard information so that it tailors to that specific product



* The dashboard information changes to that specific product.



- + Also when hovering on a specific product, a tooltip containing all of the basic information of that product will appear.



- Year button slicer: filtering informations that are from the selected year.



- Drop-down product list: For each customer type, there is a drop-down button to see specific products' information in that customer type.

Customer_Type summary table

Year	2023	
Customer_Type	Sum of Revenue	Sum of Profit
<input type="checkbox"/> Creator		
Arctis 7P+	\$28,310	
AudioBox USB 96 Studio	\$32,636	
MV7	\$38,009	
NT1-A	\$52,670	
QuadCast S	\$29,885	
Scarlett 2i2	\$44,616	
<input type="checkbox"/> Education	\$241,473	
<input type="checkbox"/> Enterprise	\$236,764	

- Other interactive features: Every bar and slice can be used as filters as well.

4) Key takeaways from the project

a) Pros

- Helps to learn how to:
 - + Navigate through the datasets, ensure the right data types of the each data points and execute joins action through Postgresql database
 - + Create visualisations and format them to ensure interactivity and attractiveness for the dashboard
 - + Use Power Query and DAX functions to create new columns and measures that helps
 - + Do a project from end to end that meets a business requirements, simulating the task of a Data Analyst and a Business Intelligence

b) Cons

- There are a few limitations while doing this project:
 - + Different operating system requires Virtual Machine (VM) to be able to finish the project makes the workflow not as seamless as I wanted
 - + Using a VM slows down the process of creating the dashboard as there are lots of glitches and lags during formatting and manipulating the datasets in powerbi

- + No built-in function to import CSV files to SQL databases so a more manual way must be done to import those. (Which can be a pros to learn)

c) What can be done better?

- Clearer plans from the beginning through better datasets observations and dashboard planning, must do it in a clearer framework
- More consistent title format through datasets and columns.