رواد مصر الرقمية

وزارة الاتصالات
وتكنولوجيا المعلومات

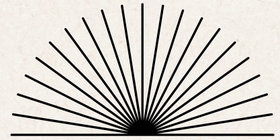
GROUP:
RS-2367 ALX1_DAT1_G1e

PROJECT:
Supply Chain Dataset Analysis

DATA ANALYSIS FINAL PROJECT
BRIEF DECK

PRESENTED BY:
Bahaa Moahmed
Mohamed Idres
Hussien Khaled
Mohamed Nabil

PRESENTED TO:
Eng : Hafssa , Eng : Mayar



Our team

Bahaa Mohamed

Mohamed Idress

Hussien Khaled

Mohamd Nabil

Objective

The project is designed to deliver actionable insights for decision-makers by addressing critical analysis and forecasting questions. It will culminate in the development of a visualization dashboard that highlights key findings. Leveraging tools like SQL, Python (with pandas, matplotlib, seaborn, and numpy), and Tableau, the project will enable data-driven decisions through thorough analysis, trend forecasting, and compelling visualizations.

Key Questions:

- Which product types generate the highest revenue?
- How does stock availability impact sales?
- What are the future sales trends for top-selling products?
- Which suppliers have the most consistent lead times?
- How do shipping costs affect profitability across regions?

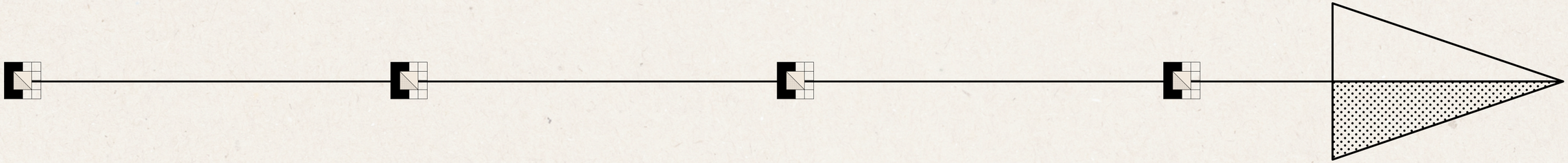
By answering these questions, the project will support decision-makers in making informed, strategic choices based on trends and data forecasts.

Agenda

WEEK 1	Build Data Model, Data Cleaning and Preprocessing
WEEK 2	Visualization Dashboard and Final Presentation
WEEK 3	Forecasting Questions Phase
WEEK 4	Visualization Dashboard and Final Presentation

Timeline

Briefly discuss the key dates for the project.



WEEK 1

- Understanding Data
- Exploration using **Python**
- Split data to tables and building data modeling and make ERD and mapping using **ERDplus** website
- Build a data model and clean and preprocess the data using **SQL**

WEEK 2

- Determine all possible analysis questions that can be deducted from the given dataset
- Set of analysis questions that can be answered via the dataset

WEEK 3

- Determine a set of forecasting questions and answer them using the trends found in the given dataset using **Python** , **Excel**

WEEK 4

- Convert all insights and answers to Visualization Dashboard using **Tableau**

Build Data Model, Data Cleaning and Preprocessing

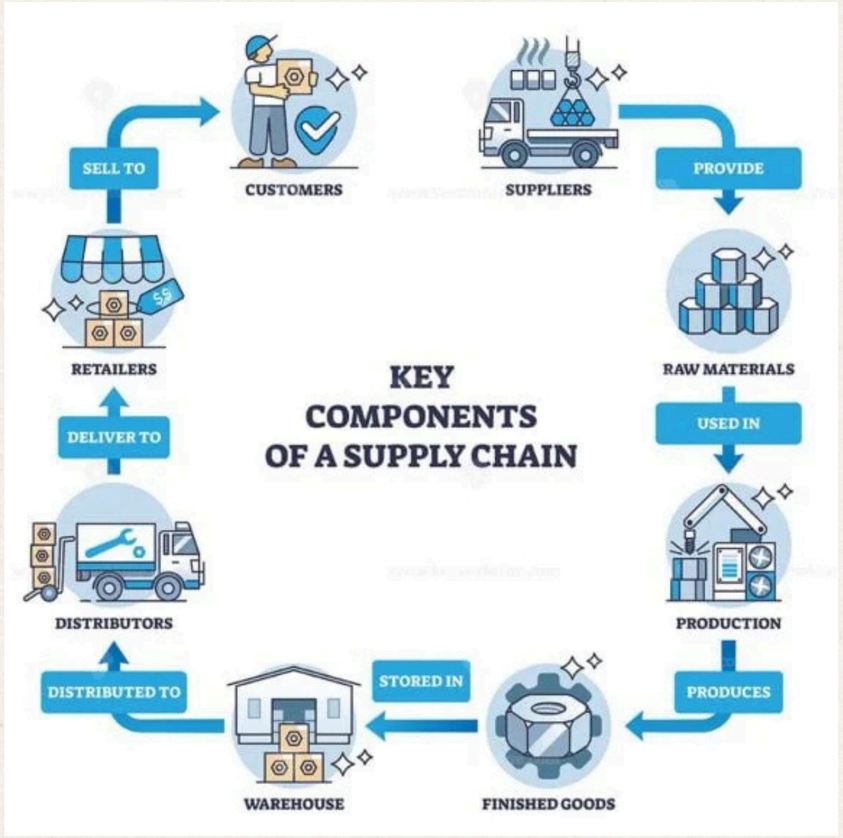
WEEK 1

UNDERSTANDING THE INDUSTRY

WEEK 1

WHAT IS SUPPLY CHAIN ?

The supply Chain is the network of production and logistics involved in producing and delivering goods to customers. And Supply Chain Analysis means analyzing various components of a Supply Chain to understand how to improve the effectiveness of the Supply Chain to create more value for customers



UNDERSTANDING THE INDUSTRY

Dataset exploration and understanding what columns refer to .

The dataset consists of 24 columns and 100 entries, covering various aspects of supply chain data. Here's a brief overview of key columns:

- **Product type, SKU, Price**: Basic product information.
- **Availability, Stock levels, Order quantities**: Inventory-related data.
- **Number of products sold, Revenue generated**: Sales data.
- **Shipping times, Shipping costs, Shipping carriers**: Logistics data.
- **Supplier name, Location, Lead time**: Supplier-related details.
- **Manufacturing costs, Defect rates**: Production and quality control metrics.
- **Transportation modes, Routes**: Transportation data.
- **Customer demographics**: Customer-related insights**.**

Exploration & Manipulation using Python

WEEK 1

Import library and understand the ship of data then make some statistics to deep understanding data trend and behiver and check for **nulls** and **duplicates** values

- Importing necessary libraries for data manipulation, analysis

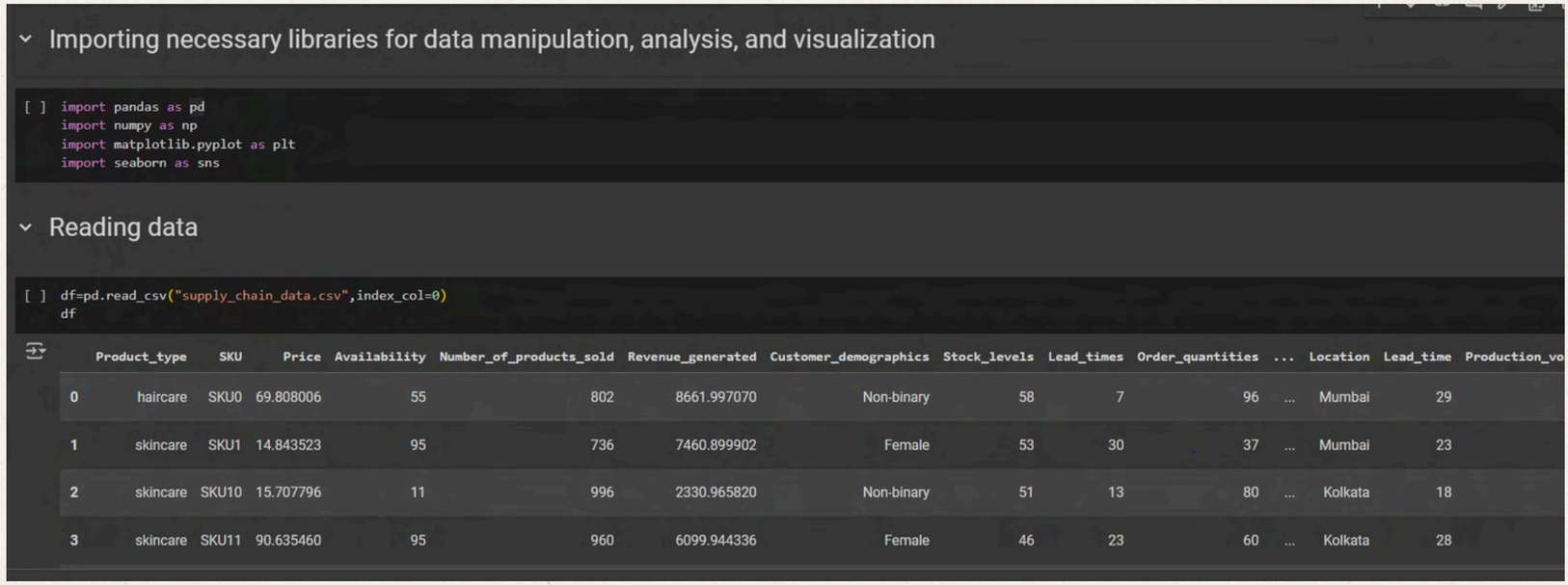
Code :

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

- Reading data

Code :

```
df=pd.read_csv
("supply_chain_data.csv",index_col=0)
df
```



Importing necessary libraries for data manipulation, analysis, and visualization

```
[ ] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
```

Reading data

```
[ ] df=pd.read_csv("supply_chain_data.csv",index_col=0)
    df
```

	Product_type	SKU	Price	Availability	Number_of_products_sold	Revenue_generated	Customer_demographics	Stock_levels	Lead_times	Order_quantities	...	Location	Lead_time	Production_vo
0	haircare	SKU0	69.808006	55	802	8661.997070	Non-binary	58	7	96	...	Mumbai	29	
1	skincare	SKU1	14.843523	95	736	7460.899902	Female	53	30	37	...	Mumbai	23	
2	skincare	SKU10	15.707796	11	996	2330.965820	Non-binary	51	13	80	...	Kolkata	18	
3	skincare	SKU11	90.635460	95	960	6099.944336	Female	46	23	60	...	Kolkata	28	

Exploration & Manipulation using Python

Reading Data

Code : df.head()

```
df.head()
```

	Product_type	SKU	Price
0	haircare	SKU0	69.808006
1	skincare	SKU1	14.843523
2	skincare	SKU10	15.707796
3	skincare	SKU11	90.635460
4	haircare	SKU12	71.213387

5 rows × 24 columns

Code : df.describe(include="all")

uct_type	SKU	Price	Availability	Number_of_pr
100	100	100.000000	100.000000	
3	100	NaN	NaN	
skincare	SKU0	NaN	NaN	
40	1	NaN	NaN	
NaN	NaN	49.462461	48.400000	
NaN	NaN	31.168193	30.743317	
NaN	NaN	1.699976	1.000000	
NaN	NaN	19.597823	22.750000	
NaN	NaN	51.239830	43.500000	
NaN	NaN	77.198231	75.000000	
NaN	NaN	99.171326	100.000000	

olumns

Code : df.info()

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 100 entries, 0 to 99
Data columns (total 24 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Product_type            100 non-null    object
 1   SKU                     100 non-null    object
 2   Price                   100 non-null    float64
 3   Availability            100 non-null    int64
 4   Number_of_products_sold 100 non-null    int64
 5   Revenue_generated       100 non-null    float64
 6   Customer_demographics   100 non-null    object
 7   Stock_levels            100 non-null    int64
 8   Lead_times              100 non-null    int64
 9   Order_quantities        100 non-null    int64
 10  Shipping_times          100 non-null    int64
 11  Shipping_carriers       100 non-null    object
 12  Shipping_costs          100 non-null    float64
 13  Supplier_name           100 non-null    object
 14  Location                100 non-null    object
 15  Lead_time               100 non-null    int64
 16  Production_volumes      100 non-null    int64
 17  Manufacturing_lead_time 100 non-null    int64
 18  Manufacturing_costs     100 non-null    float64
 19  Inspection_results      100 non-null    object
 20  Defect_rates            100 non-null    float64
 21  Transportation_modes    100 non-null    object
 22  Routes                  100 non-null    object
 23  Costs                   100 non-null    float64
dtypes: float64(6), int64(9), object(9)
memory usage: 19.5+ KB
```

Exploration & Manipulation using Python

Checking for nulls

The isnull() function checks for missing (null) values in the DataFrame df and returns a DataFrame of the same shape with True where values are null and False where they're not. The sum() function then counts the total number of null values in each column.

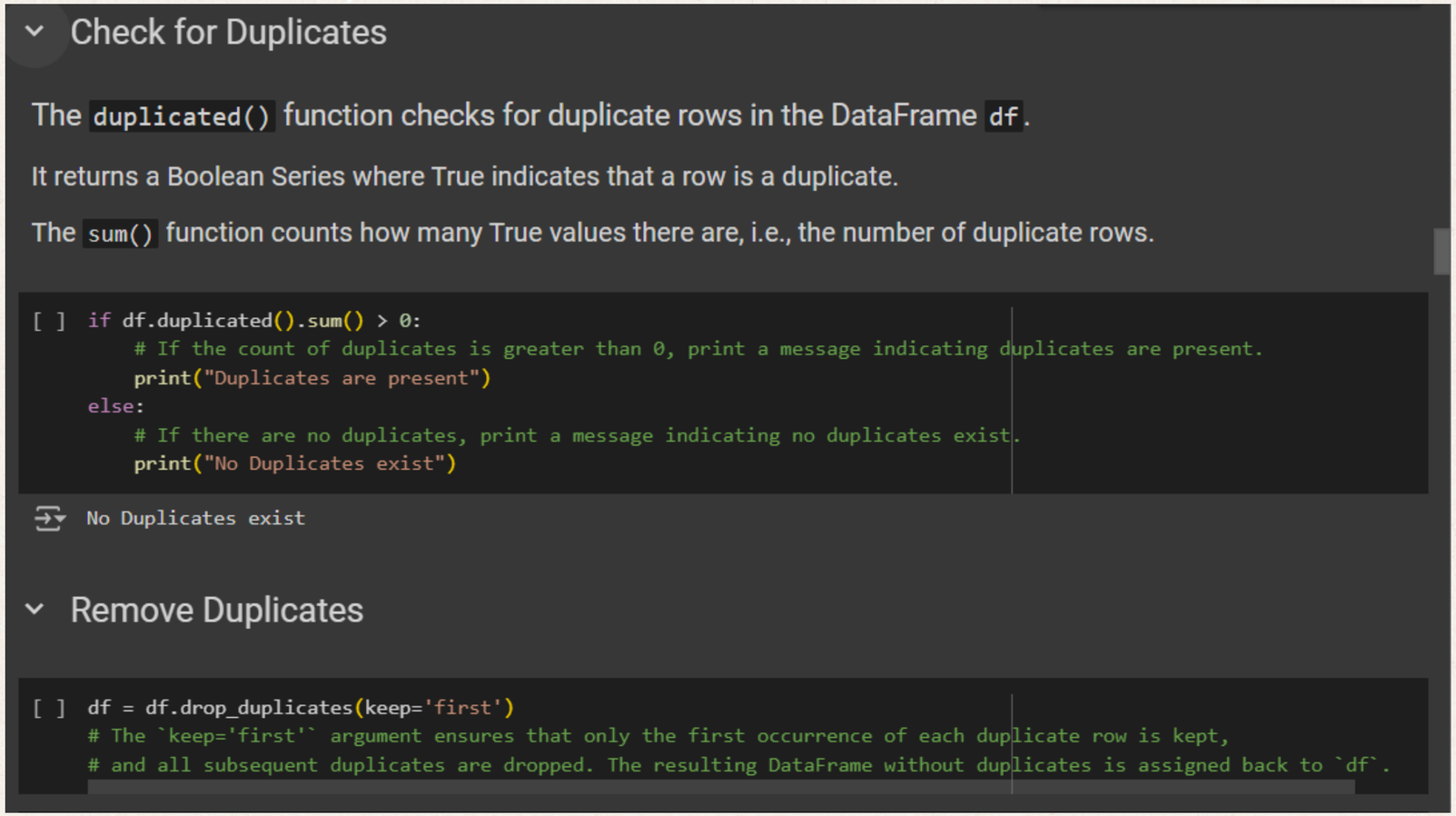
```
df.isnull().sum()>0
```

```
Product_type              False
SKU                       False
Price                     False
Availability              False
Number_of_products_sold   False
Revenue_generated         False
Customer_demographics     False
Stock_levels              False
Lead_times                False
Order_quantities          False
Shipping_times            False
Shipping_carriers         False
Shipping_costs            False
Supplier_name             False
Location                  False
Lead_time                 False
Production_volumes        False
Manufacturing_lead_time   False
Manufacturing_costs       False
Inspection_results        False
Defect_rates              False
Transportation_modes      False
Routes                    False
Costs                     False
dtype: bool
```

Exploration & Manipulation using Python

Check and Remove Duplicates

The duplicated() function checks for duplicate rows in the DataFrame df. It returns a Boolean Series where True indicates that a row is a duplicate. The sum() function counts how many True values there are, i.e., the number of duplicate rows.



Exploration & Manipulation using Python

Check that dataset has no null or duplicate

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 100 entries, 0 to 99
Data columns (total 24 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Product_type             100 non-null    object 
 1   SKU                      100 non-null    object 
 2   Price                    100 non-null    float64
 3   Availability             100 non-null    int64  
 4   Number_of_products_sold  100 non-null    int64  
 5   Revenue_generated        100 non-null    float64
 6   Customer_demographics    100 non-null    object 
 7   Stock_levels             100 non-null    int64  
 8   Lead_times               100 non-null    int64  
 9   Order_quantities         100 non-null    int64  
 10  Shipping_times           100 non-null    int64  
 11  Shipping_carriers        100 non-null    object 
 12  Shipping_costs           100 non-null    float64
 13  Supplier_name            100 non-null    object 
 14  Location                 100 non-null    object 
 15  Lead_time                100 non-null    int64  
 16  Production_volumes       100 non-null    int64  
 17  Manufacturing_lead_time  100 non-null    int64  
 18  Manufacturing_costs      100 non-null    float64
 19  Inspection_results       100 non-null    object 
 20  Defect_rates             100 non-null    float64
 21  Transportation_modes     100 non-null    object 
 22  Routes                   100 non-null    object 
 23  Costs                    100 non-null    float64
dtypes: float64(6), int64(9), object(9)
```

```
df.isnull().sum()
```

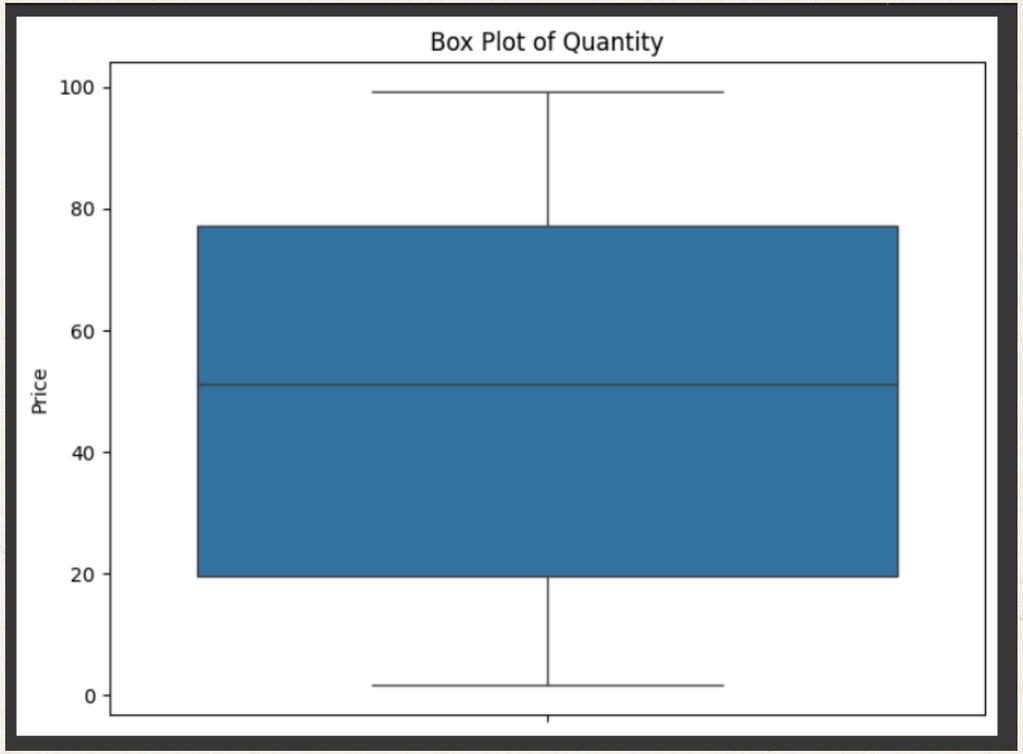
```
Product_type               0
SKU                        0
Price                      0
Availability               0
Number_of_products_sold    0
Revenue_generated          0
Customer_demographics      0
Stock_levels               0
Lead_times                 0
Order_quantities           0
Shipping_times             0
Shipping_carriers          0
Shipping_costs             0
Supplier_name              0
Location                   0
Lead_time                  0
Production_volumes         0
Manufacturing_lead_time    0
Manufacturing_costs        0
Inspection_results         0
Defect_rates               0
Transportation_modes       0
Routes                     0
Costs                      0
dtype: int64
```

Exploration & Manipulation using Python

Create boxplot graph to detect outliers of the data

Create boxplot graph to detect outliers of the data

```
# Box plot for Quantity
plt.figure(figsize=(8, 6))

# Use Seaborn's `boxplot()` function to create a box plot.
# The `data` parameter specifies the DataFrame `df`, and `y='Quantity'` indicates that
# we want to plot the 'Quantity' column on the y-axis.
sns.boxplot(data=df, y='Price')

# Set the title of the plot to 'Box Plot of Quantity'.
plt.title('Box Plot of Quantity')

# Set the label for the y-axis to 'Quantity'.
plt.ylabel('Price')

# Display the plot.
plt.show()
```



Exploration & Manipulation using Python

Calculate outliers and their index

Calculate outliers and their index

```
[ ]  # Calculate the first quartile (Q1), which is the 25th percentile of the ' Price' column.
     Q1 = df['Price'].quantile(0.25)

     # Calculate the third quartile (Q3), which is the 75th percentile of the ' Price' column.
     Q3 = df['Price'].quantile(0.75)

     # Calculate the Interquartile Range (IQR), which is the difference between Q3 and Q1.
     IQR = Q3 - Q1

     # Calculate the lower bound for detecting outliers, which is Q1 minus 1.5 times the IQR.
     lower_bound = Q1 - 1.5 * IQR

     # Calculate the upper bound for detecting outliers, which is Q3 plus 1.5 times the IQR.
     upper_bound = Q3 + 1.5 * IQR

     # Detect outliers in the ' Price' column.
     # Rows where ' Price' is less than the lower bound or greater than the upper bound are considered outliers.
     outliers = df[(df['Price'] < lower_bound) | (df['Price'] > upper_bound)]

     # Print the outliers detected in the ' Price' column.
     print("Outliers in Price:")
     print(outliers)
```

```
Outliers in Price:
Empty DataFrame
Columns: [Product_type, SKU, Price, Availability, Number_of_products_sold, Revenue_generated, Customer_demographics,
Index: []

[0 rows x 24 columns]
```

Exploration & Manipulation using Python

Data types and converting to the right types

```
df.dtypes
```

```
Product_type                object
SKU                         object
Price                      float64
Availability                 int64
Number_of_products_sold      int64
Revenue_generated          float64
Customer_demographics       object
Stock_levels                 int64
Lead_times                   int64
Order_quantities             int64
Shipping_times               int64
Shipping_carriers           object
Shipping_costs             float64
Supplier_name               object
Location                    object
Lead_time                    int64
Production_volumes           int64
Manufacturing_lead_time      int64
Manufacturing_costs        float64
Inspection_results          object
Defect_rates               float64
Transportation_modes        object
Routes                      object
Costs                      float64
dtype: object
```

Exploration & Manipulation using Python

Create the Revenue column

```
# Create the Revenue column
df['Profit'] = df['Revenue_generated'] - df['Costs']
df.head()
```

	Product_type	SKU	Price	Availability	Number_of_products_sold	Revenue_generated	Customer_demographics
0	haircare	SKU0	69.808006	55	802	8661.997070	Non-binary
1	skincare	SKU1	14.843523	95	736	7460.899902	Female
2	skincare	SKU10	15.707796	11	996	2330.965820	Non-binary
3	skincare	SKU11	90.635460	95	960	6099.944336	Female
4	haircare	SKU12	71.213387	41	336	2873.741455	Unknown

5 rows × 25 columns

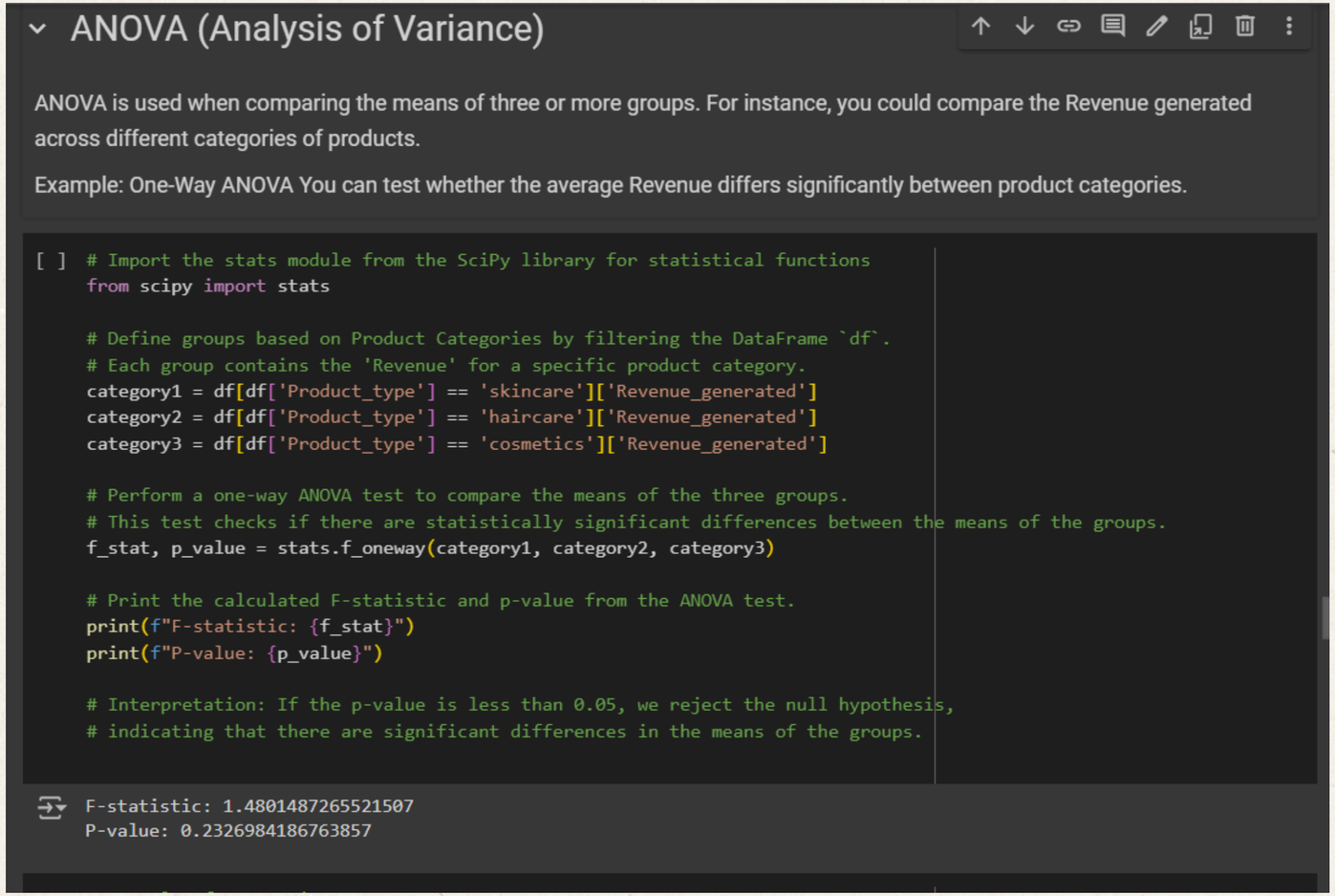
```
[ ] df.tail()
```

	Product_type	SKU	Price	Availability	Number_of_products_sold	Revenue_generated	Customer_demographics
95	haircare	SKU95	77.903931	65	672	7386.363770	Unknown
96	cosmetics	SKU96	24.423132	29	324	7698.424805	Non-binary

Exploration & Manipulation using Python

ANOVA (Analysis of Variance)

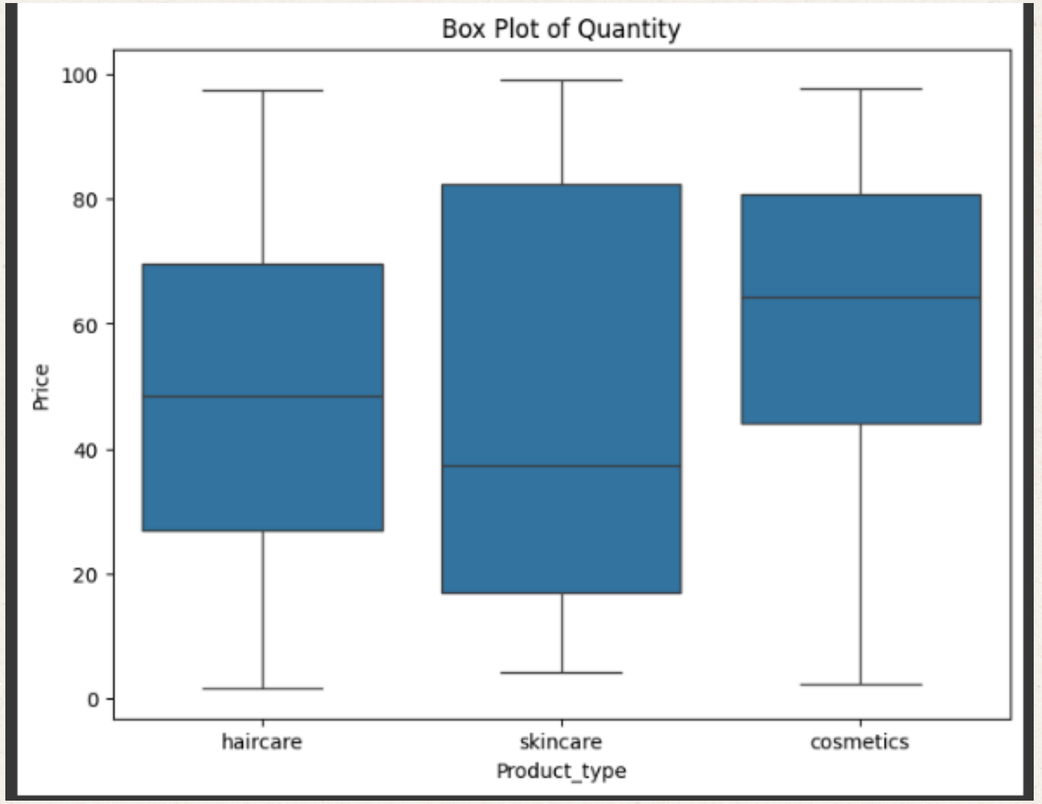
ANOVA is used when comparing the means of three or more groups. For instance, you could compare the Revenue generated across different categories of products.
Example: One-Way ANOVA You can test whether the average Revenue differs significantly between product categories.



Exploration & Manipulation using Python

Box plot for Quantity

```python
# Box plot for Quantity
plt.figure(figsize=(8, 6))

# Use Seaborn's `boxplot()` function to create a box plot.
# The `data` parameter specifies the DataFrame `df`, and `y='Quantity'` indicates that
# we want to plot the 'Quantity' column on the y-axis.
sns.boxplot(data=df, x='Product_type' , y='Price')

# Set the title of the plot to 'Box Plot of Quantity'.
plt.title('Box Plot of Quantity')

# Set the label for the y-axis to 'Quantity'.
plt.ylabel('Price')

# Display the plot.
plt.show()
```

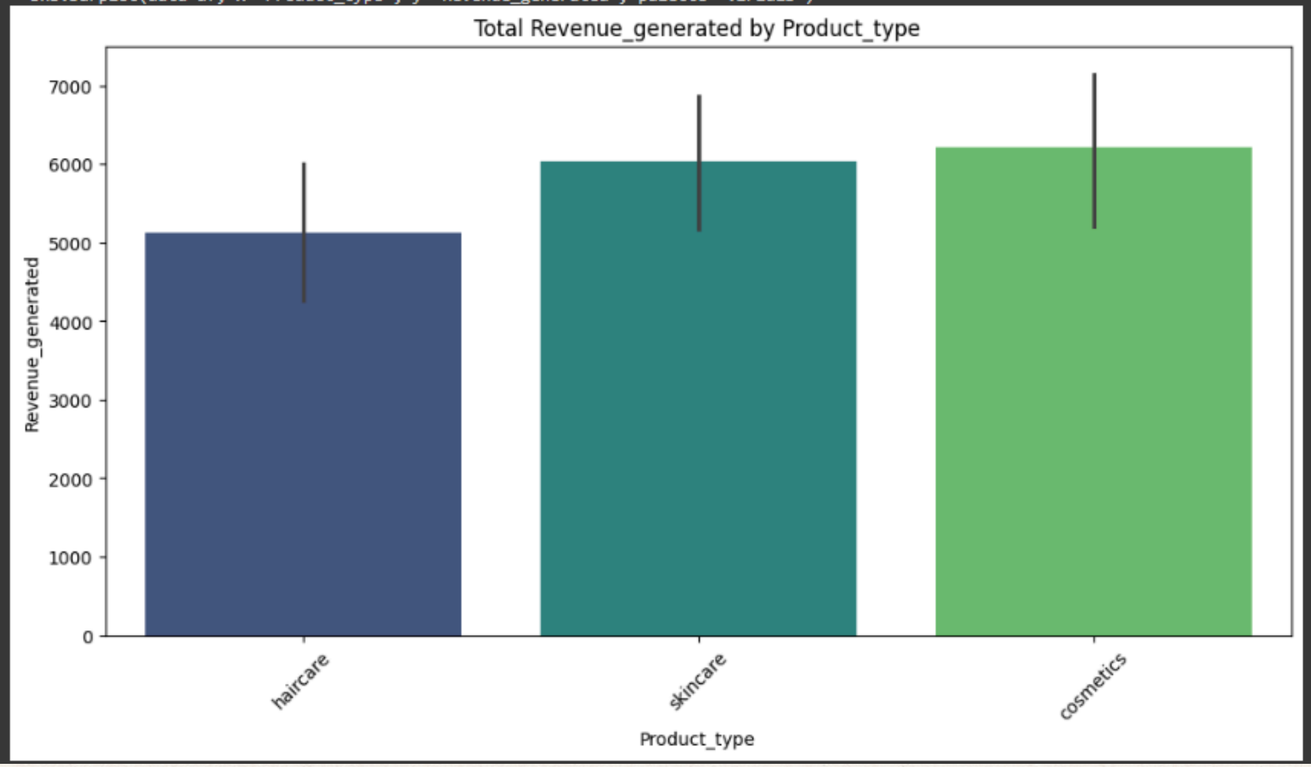


Exploration & Manipulation using Python

Plotting Revenue by Category

```python
# Plotting Revenue by Category
plt.figure(figsize=(12, 6))
sns.barplot(data=df, x='Product_type', y='Revenue_generated', palette='viridis')
plt.title('Total Revenue_generated by Product_type')
plt.xticks(rotation=45)
plt.xlabel('Product_type')
plt.ylabel('Revenue_generated')
plt.show()
```

```
C:\Users\bahaa\AppData\Local\Temp\ipykernel_16804\2575802955.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` fo

  sns.barplot(data=df, x='Product_type', y='Revenue_generated', palette='viridis')
```



Exploration & Manipulation using Python

Extract file to csv file

```
[ ]  # to save file to csv
     df.to_csv("Output . csv" , index=False)
```

Exploration & Manipulation using SQL

WEEK 1

Import data into Microsoft SQL Server Management studio and creating relational database with splitted tables then insert data into this new tables

```sql
Create database supply_chain
use supply_chain

select * from supply_chain_data
-----------------------------------split the data into multiple tables--
CREATE TABLE Products (
    SKU VARCHAR(50) PRIMARY KEY,
    ProductType VARCHAR(50),
    Price FLOAT,
    StockLevels INT,
    Availability INT,
    LeadTimes INT,
    OrderQuantities INT)

CREATE TABLE Sales (
    SKU VARCHAR(50) PRIMARY KEY,
    NumberOfProductsSold INT,
    RevenueGenerated FLOAT,
    CustomerDemographics VARCHAR(50),
    ShippingTimes INT,
    ShippingCosts FLOAT,
    FOREIGN KEY (SKU) REFERENCES Products(SKU))
```

```sql
CREATE TABLE Suppliers (
    SKU VARCHAR(50) PRIMARY KEY,
    SupplierName VARCHAR(50),
    Location VARCHAR(50),
    LeadTime INT,
    ProductionVolumes INT,
    ManufacturingLeadTime INT,
    ManufacturingCosts FLOAT)

CREATE TABLE Inspections (
    SKU VARCHAR(50) PRIMARY KEY,
    InspectionResults VARCHAR(50),
    DefectRates FLOAT,
    FOREIGN KEY (SKU) REFERENCES Products(SKU))

CREATE TABLE Shipping (
    SKU VARCHAR(50) PRIMARY KEY,
    ShippingCarriers VARCHAR(50),
    TransportationModes VARCHAR(50),
    Routes VARCHAR(50),
    Costs FLOAT,
    FOREIGN KEY (SKU) REFERENCES Products(SKU))
```

Exploration & Manipulation using SQL

WEEK 1

Insert data into this new tables and Check for it

```
----------------------------------------------Inserting data into tables ----------------------------------------

-- Inserting data into Products table from supply_chain_data

INSERT INTO Products (SKU, ProductType, Price, StockLevels, Availability, LeadTimes, OrderQuantities)
SELECT SKU, Product_type, Price, Stock_levels, Availability, lead_time, Order_quantities
FROM supply_chain_data

select * from Products


-- Inserting data into Sales table from supply_chain_data

INSERT INTO Sales (SKU, NumberOfProductsSold, RevenueGenerated, CustomerDemographics, ShippingTimes, ShippingCosts)
SELECT SKU, Number_of_products_sold, Revenue_generated, Customer_demographics, Shipping_times, Shipping_costs
FROM supply_chain_data

select * from Sales
```

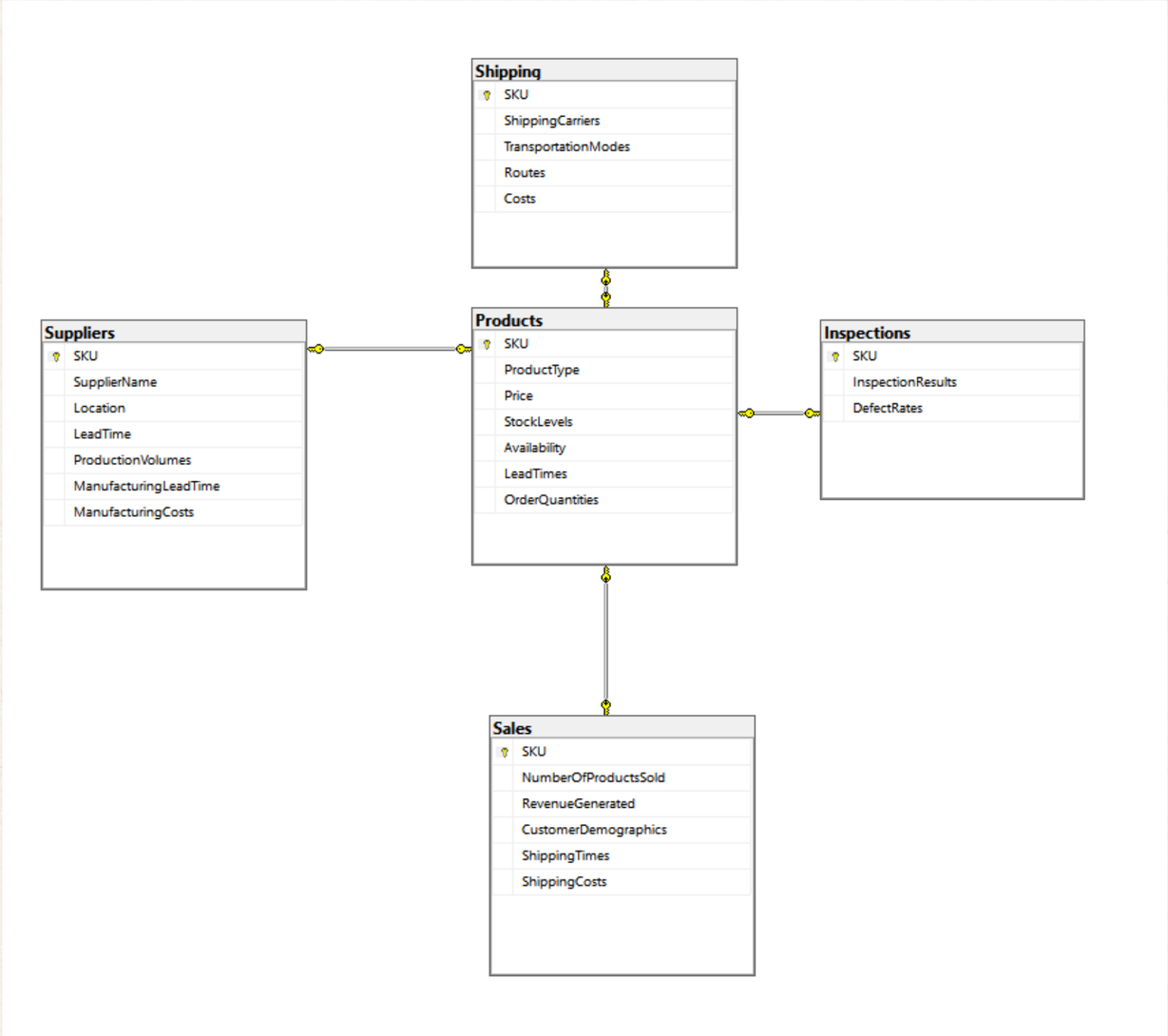
```
-- Inserting data into Suppliers table from supply_chain_data

INSERT INTO Suppliers (SKU,SupplierName, Location, LeadTime, ProductionVolumes, ManufacturingLeadTime, ManufacturingCosts)
SELECT SKU,Supplier_name, Location, Lead_time, Production_volumes, Manufacturing_lead_time, Manufacturing_costs
FROM supply_chain_data

select * from Suppliers

-- Inserting data into Inspections table from supply_chain_data

INSERT INTO Inspections (SKU, InspectionResults, DefectRates)
SELECT SKU, Inspection_results, Defect_rates
FROM supply_chain_data

select * from Inspections

-- Inserting data into Shipping table from supply_chain_data

INSERT INTO Shipping (SKU, ShippingCarriers, TransportationModes, Routes, Costs)
SELECT SKU, Shipping_carriers, Transportation_modes, Routes, Costs
FROM supply_chain_data

select * from Shipping
```

```
--------------------------
select * from Inspections
select * from Products
select * from Sales
select * from Shipping
select * from Suppliers
```

Exploration & Manipulation using SQL

WEEK 1

Create new degrame and relation between new tables



Questions Phase

WEEK 2

Questions Phase

WEEK 2

Determine all possible analysis questions that can be deducted from the given dataset and would be of interest to the organization’s decision makers

Sales and Revenue

What is the impact of product type on sales and revenue?
How does price correlate with the number of products sold?
Which products/SKUs generate the most revenue and why?

Shipping and Logistics

How do shipping times and costs vary by transportation mode and carrier?
What are the most efficient shipping routes for minimizing costs and delivery times?
How do shipping delays impact customer satisfaction and future orders?

Inventory and Stock Management

How does stock availability influence revenue generation?
What are the optimal stock levels for maximizing sales while avoiding overstock?
How do lead times affect stock levels and overall inventory management?

Customer Demographics and Behavior

How do customer demographics influence purchasing behavior?
What is the relationship between customer demographics and product preference?
How does customer location affect shipping times and costs?

Supplier and Manufacturing Performance

Which suppliers have the most reliable lead times and lowest manufacturing costs?
How do defect rates vary by supplier or product type?
What impact does manufacturing lead time have on order fulfillment?

Cost and Profitability

What is the relationship between manufacturing costs and profitability?
Which products or suppliers have the highest cost-to-revenue ratio, and how can this be optimize

Forecasting Questions Phase

WEEK 3

Forecasting Questions Phase

WEEK 3

the objective is to use historical data to predict future trends in sales, stock, and other relevant metrics. Here are forecasting questions you can focus on, with a summary of each.

Forecasting Summary for Product Sales:

- Skincare: Currently the top-selling category with a total of **20,731** units sold. It is expected to continue performing well.
- Haircare: Currently has sold **13,611** units, with future sales predicted to rise slightly .
- Cosmetics: Currently at **11,757** units sold, future sales are projected to remain steady .

Recomendation :

- Skincare remains the dominant product category, with strong projected sales.
- Haircare shows moderate growth potential.
- Cosmetics may see steady but slightly lower sales compared to the other categories.

Row Labels	Sum of Number of products sold
skincare	20,731
haircare	13,611
cosmetics	11,757
Grand Total	**46,099**

Top 5 Product Sold

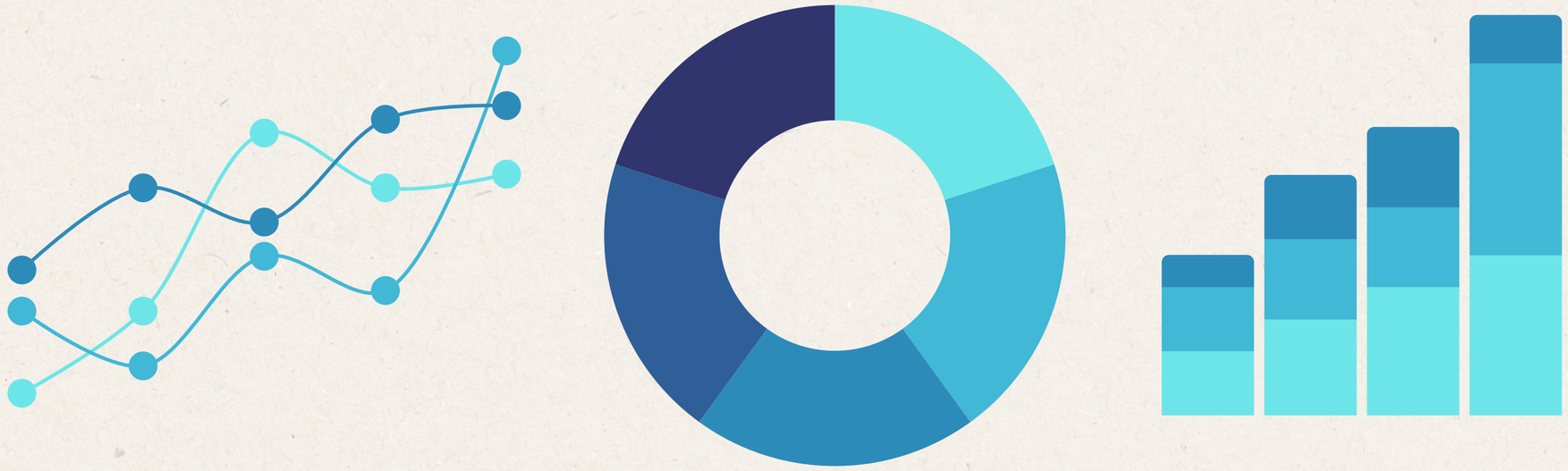
Row Labels	Sum of Number of products sold
SKU10	996
SKU94	987
SKU9	980
SKU36	963
SKU37	963
Grand Total	**4,889**

Lowest 5 Product Sold

Row Labels	Sum of Number of products sold
SKU2	8
SKU45	24
SKU85	25
SKU48	29
SKU70	32
Grand Total	**118**

Visualization Dashboard and Final Presentation

WEEK 4

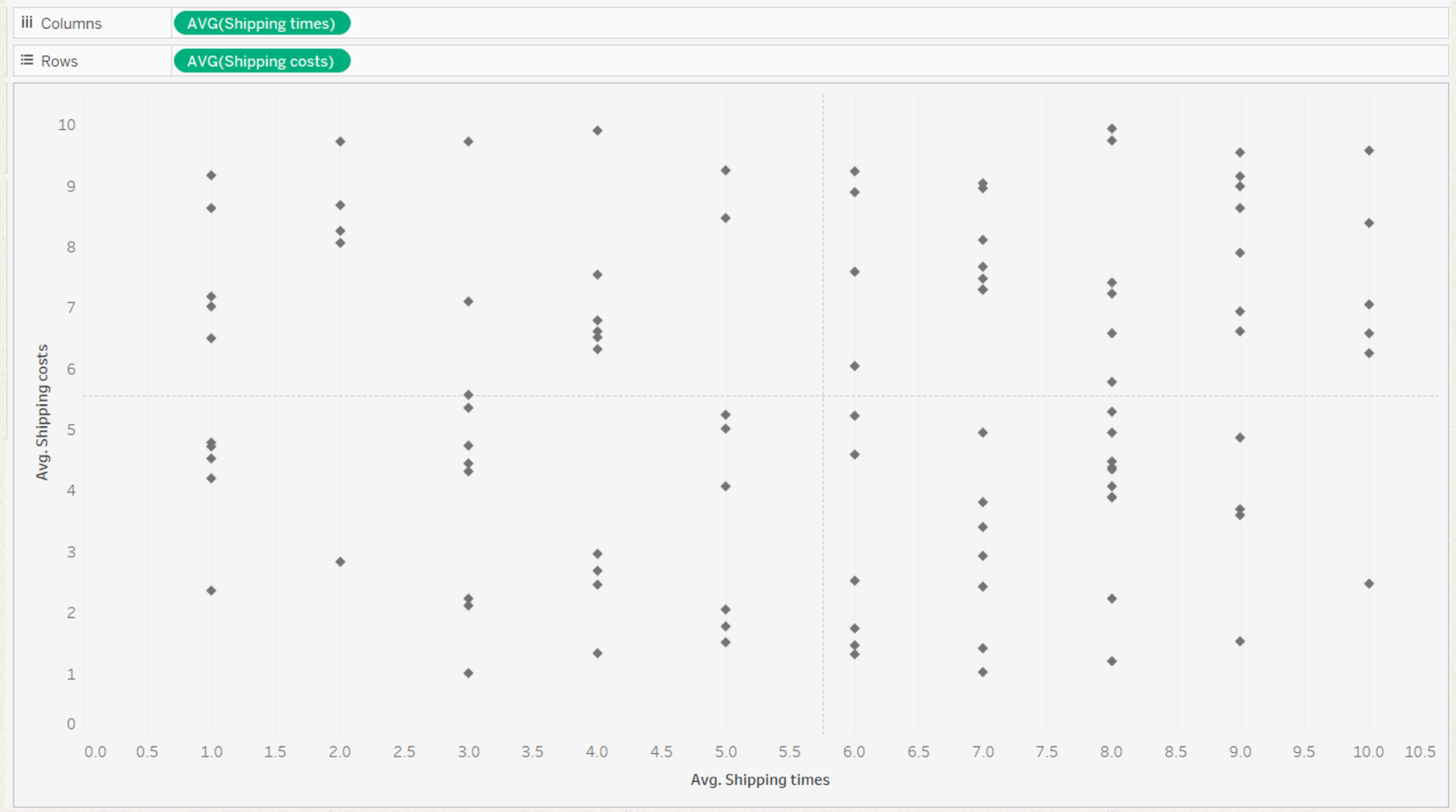


Visualization Dashboard using Tableau

WEEK 4

Build a Visualization Dashboard: Build a Tableau visualization dashboard that visualize the answers to all answered questions.

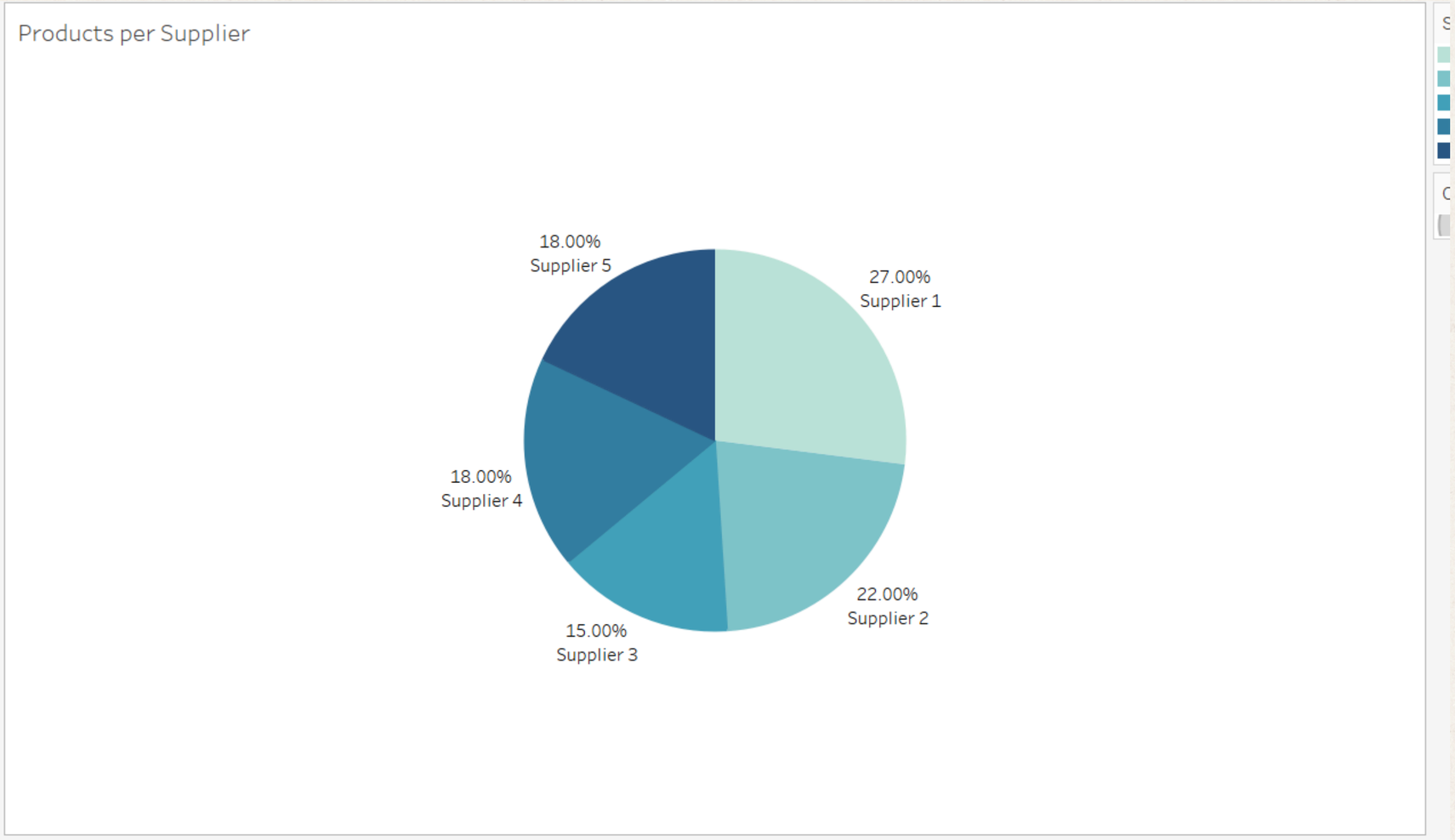
Shipping Costs Vs Shipping Times



Visualization Dashboard using Tableau

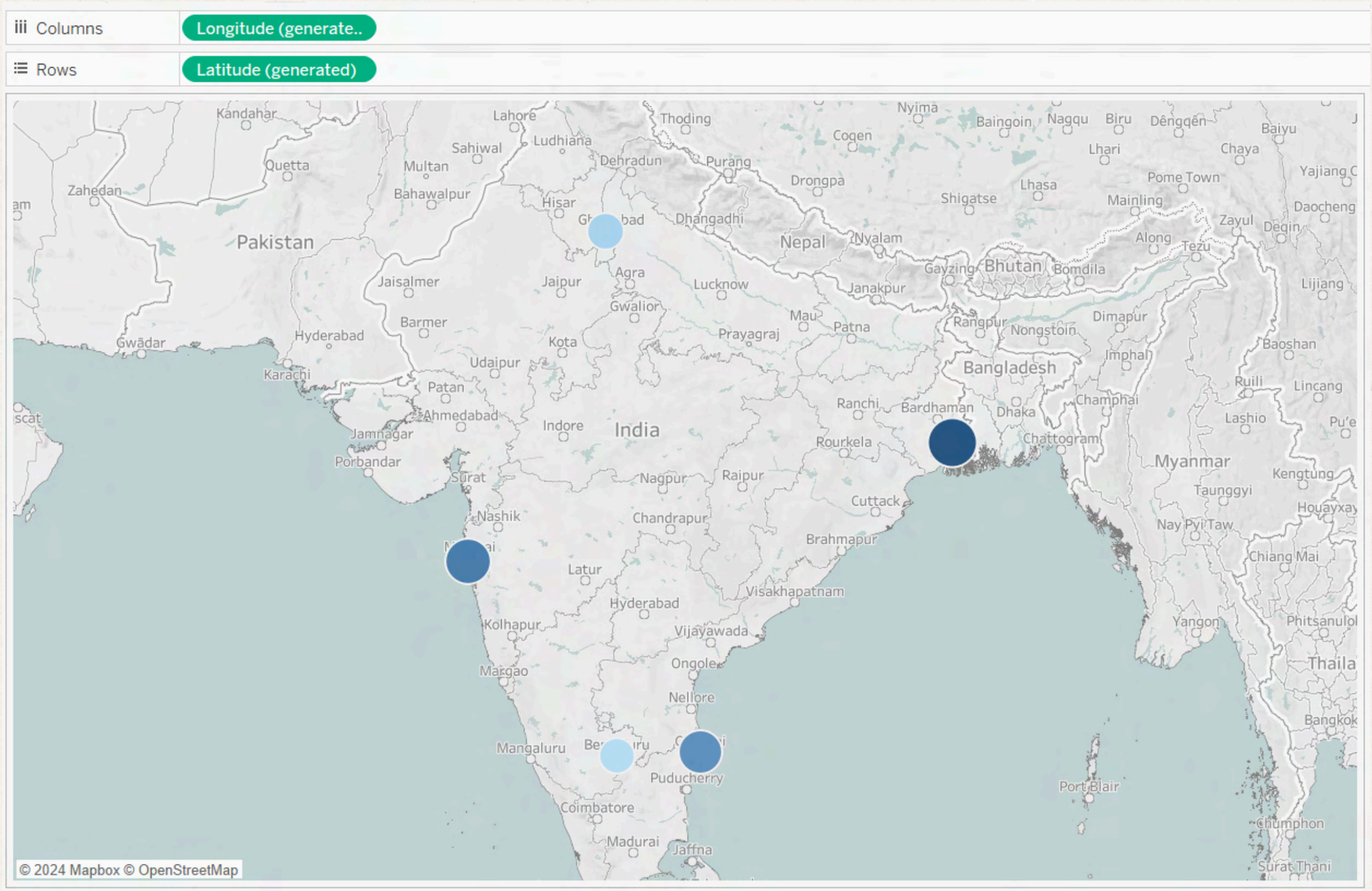
WEEK 4

Products per Supplier



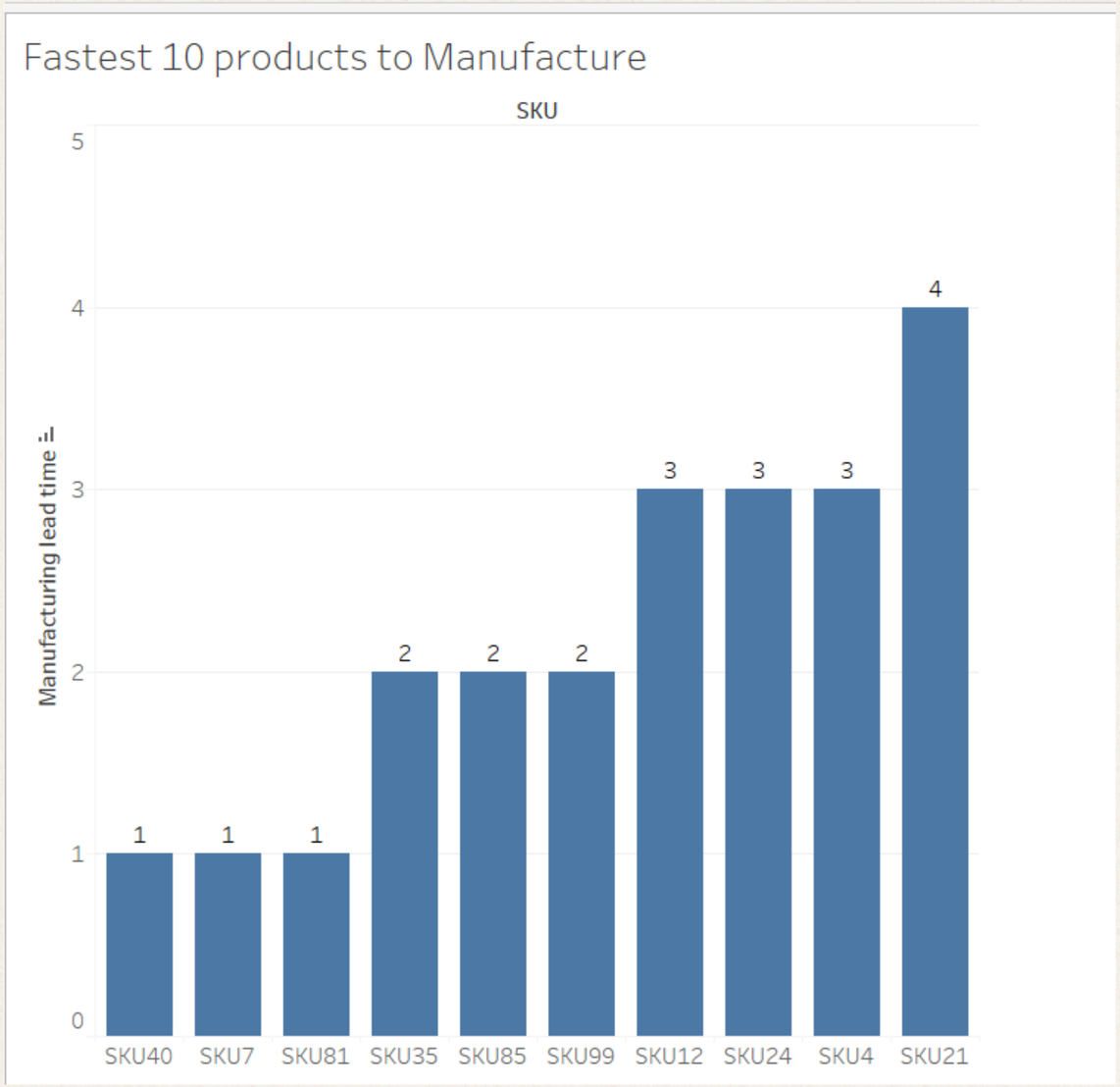
Visualization Dashboard using Tableau

Location



Visualization Dashboard using Tableau

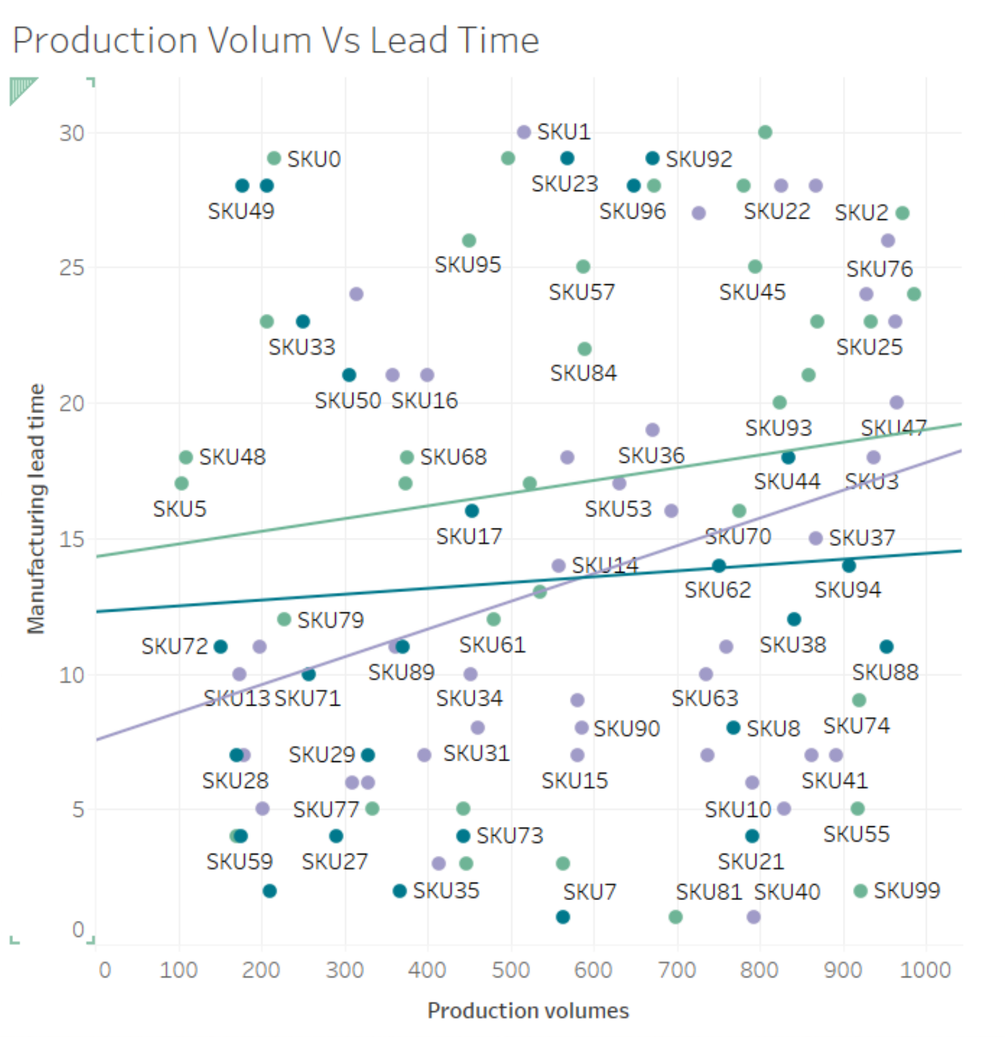
Fastest 10 products to Manufacture



Visualization Dashboard using Tableau

WEEK 4

Production Volum Vs Lead Time



Visualization Dashboard using Tableau

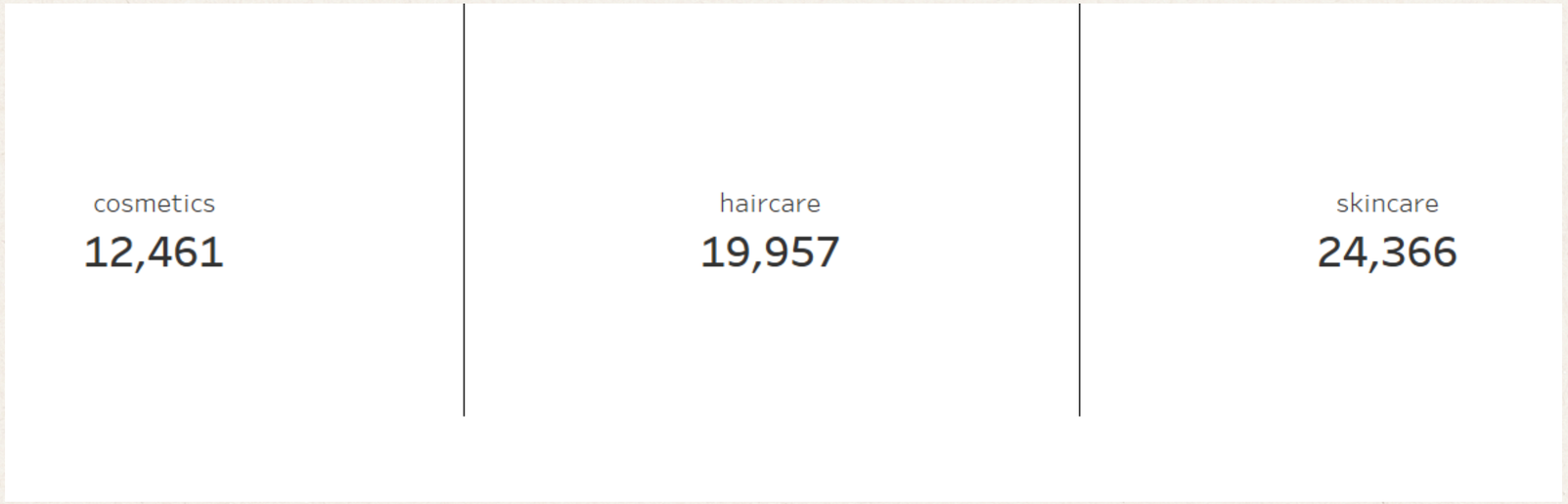
WEEK 4

Total Manufacturing Cost

Visualization Dashboard using Tableau

WEEK 4

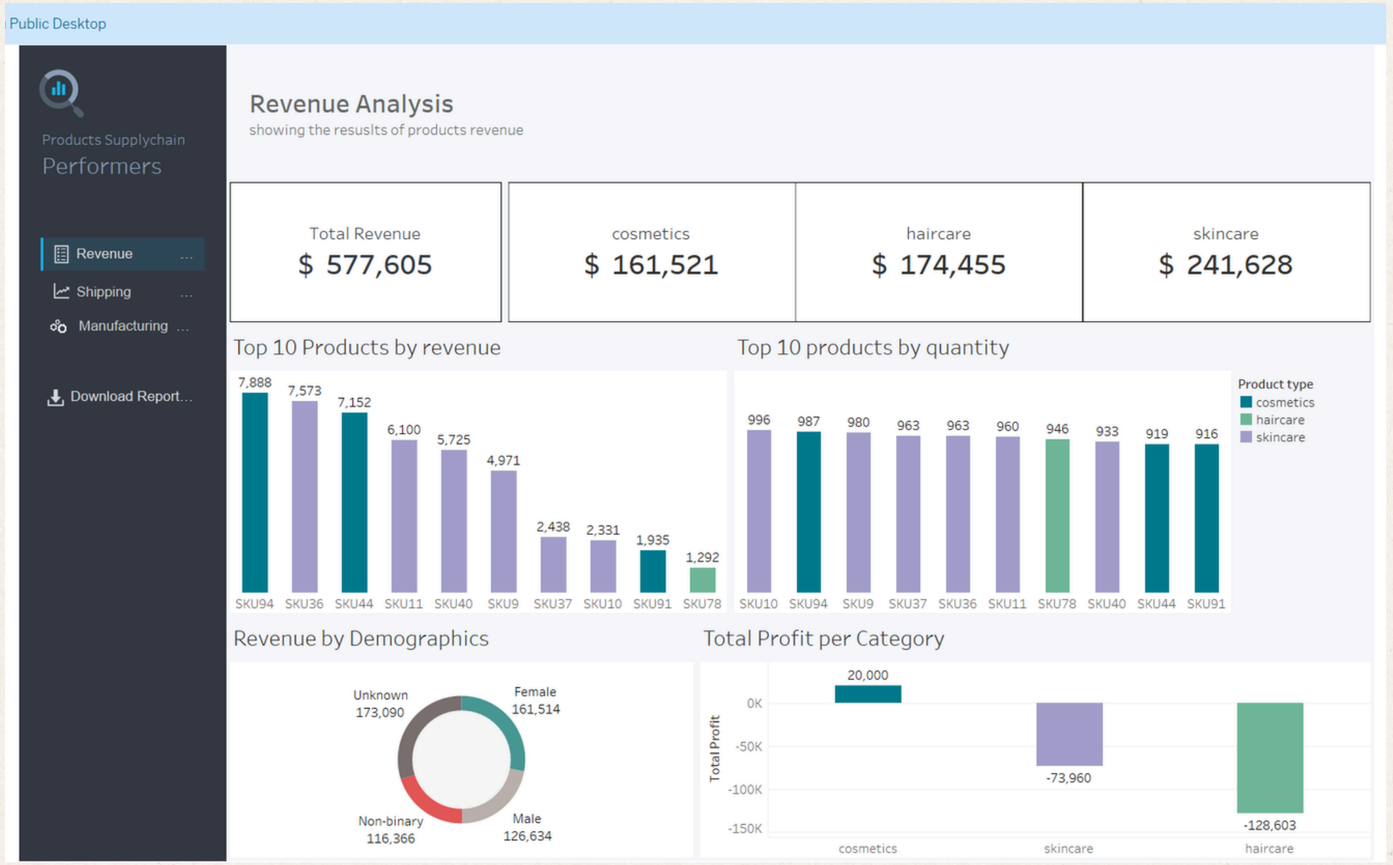
Total Production Volum



Visualization Dashboard using Tableau

WEEK 4

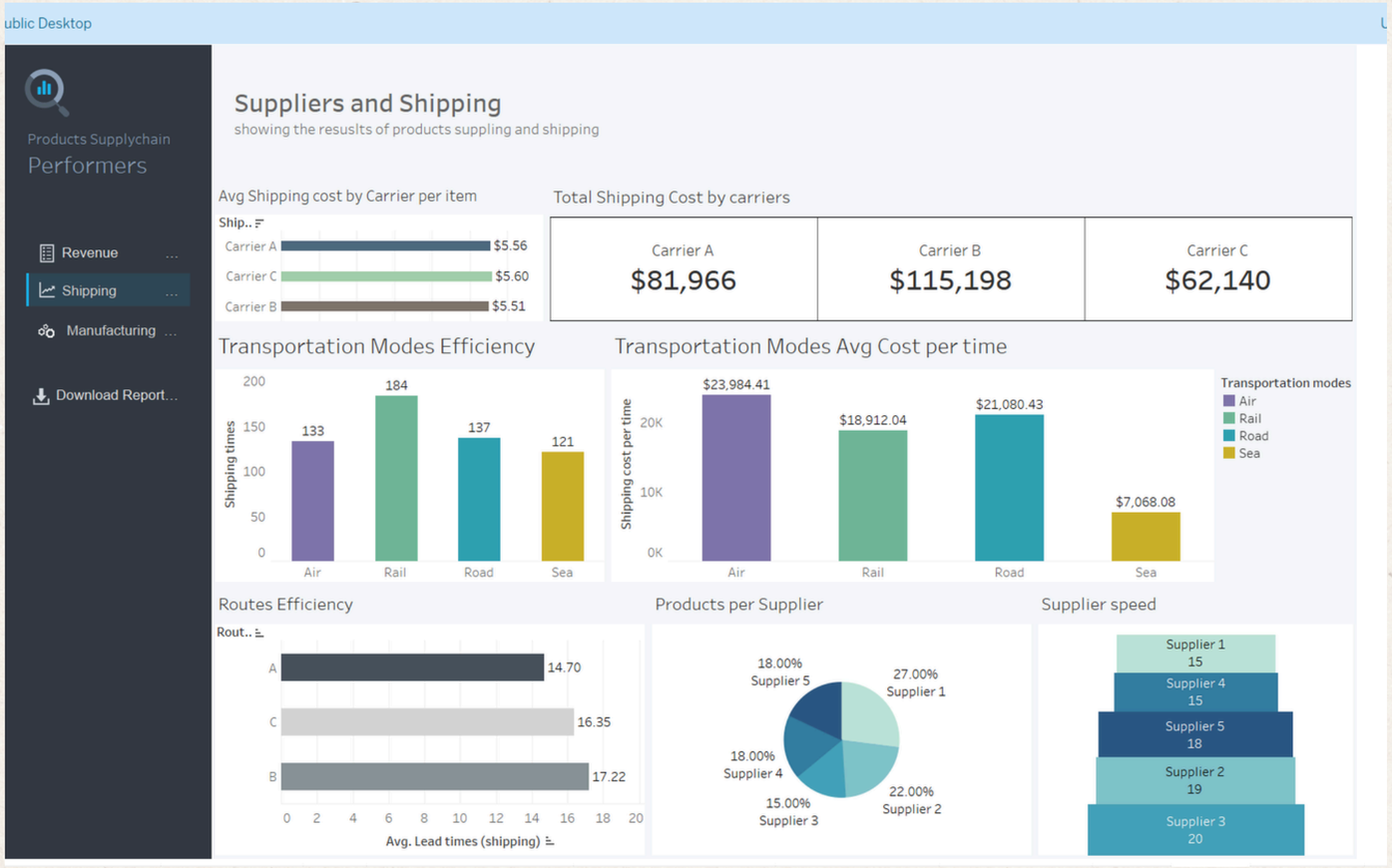
Revenue Dashboard



Visualization Dashboard using Tableau

WEEK 4

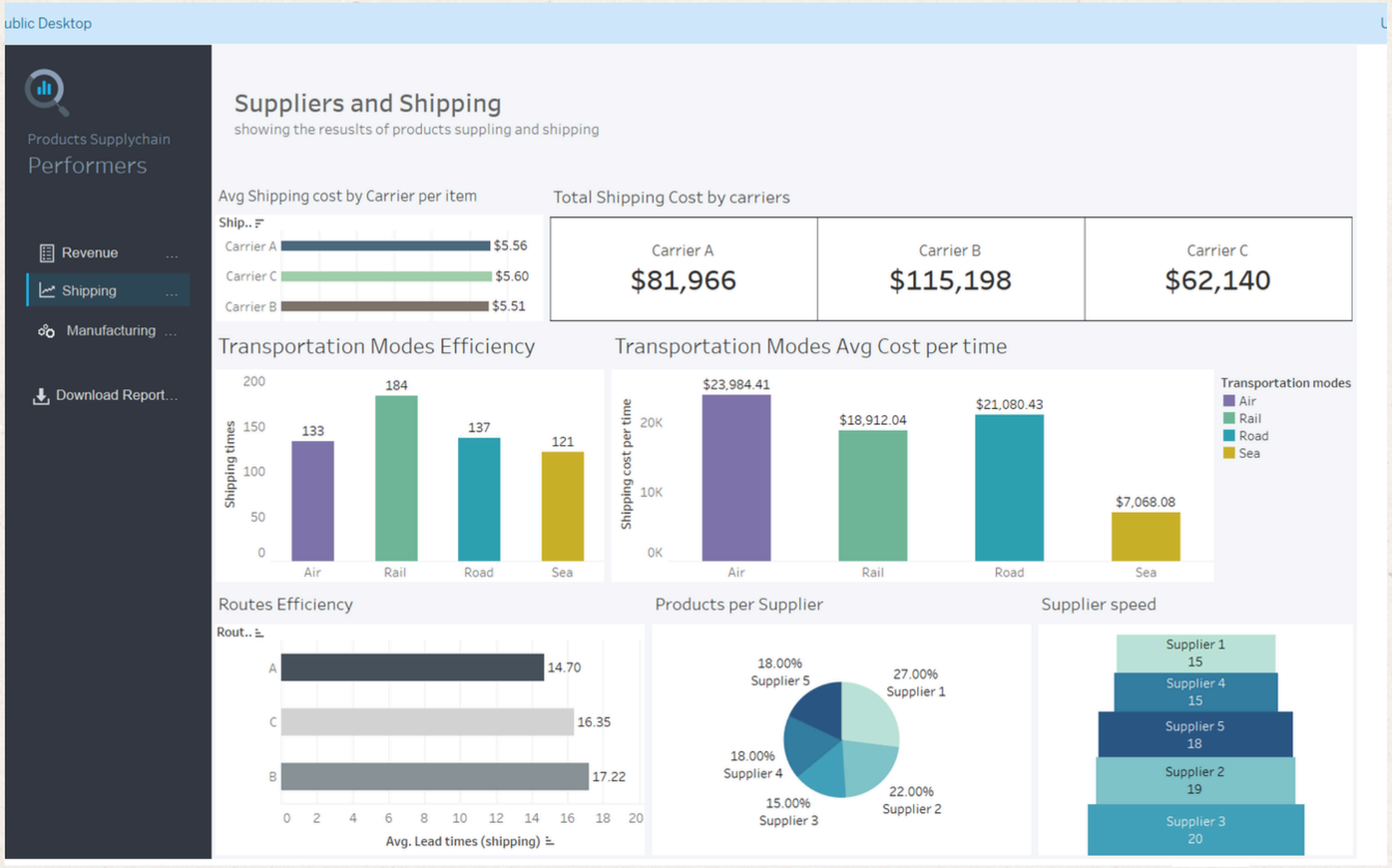
Revenue Dashboard



Visualization Dashboard using Tableau

WEEK 4

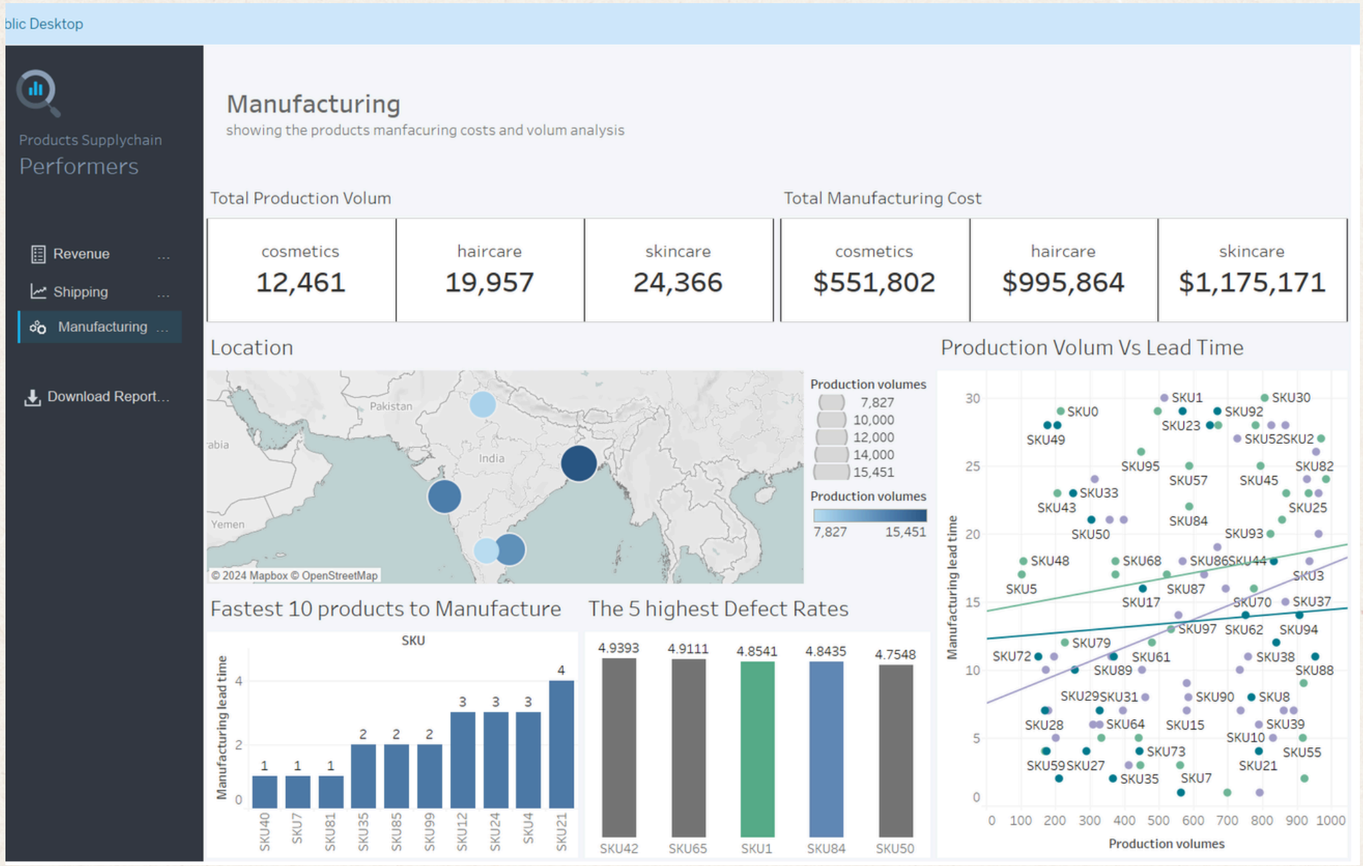
Shipping Dashboard



Visualization Dashboard using Tableau

WEEK 4

Manufacturing Dashboard



recommendation and conclusion

Top Revenue-Generating Product And Categories:

- Skincare is the leading revenue generator with $241,628 in total, from 20,731 units sold.
- Haircare follows, contributing $174,455 in revenue, with 13,611 units sold.
- Cosmetics brings in $161,521 from 11,757 units sold.

For example, the product type "haircare" (SKU0) generates $8,661 from 802 units sold, while "skincare" (SKU1) generates $7,460 from 736 units sold

Conclusion:

Skincare products dominate both in terms of revenue and units sold. The other two categories, haircare and cosmetics, also perform well but generate slightly less revenue and have fewer units sold.

Recommendation:

- Prioritize skincare products for restocking and production due to their strong sales performance and high revenue generation.
- For haircare and cosmetics, focus on improving inventory efficiency, marketing, or product variety to drive higher sales and revenue to match skincare's performance.
- Prioritize high-revenue products like these for production, while reducing stock for lower-selling products to optimize resources.

recommendation and conclusion

Cost Optimization for Transportation:

- Transportation by Sea has the lowest cost at $7,103, followed by Air at $14,605.
- Rail incurs a cost of $15,169, while Road is the most expensive at $16,048.

for example Products transported via Route C by Air (e.g., SKU2) have a lower transportation cost of $141.92, compared to Route A by Rail (e.g., SKU3), which costs $254.78.

Recommendation:

- Shift more shipments to Sea transport when possible to take advantage of its lower costs. Consider reducing reliance on Road transport, which is the most expensive mode, and explore more efficient planning for Air and Rail to balance cost and speed.

recommendation and conclusion

Sales Driven by City:

- Mumbai generates the highest revenue at $137,755.03, closely followed by Kolkata with $137,077.55.
- Chennai also performs well, contributing $119,142.82 to the total revenue.
- Bangalore and Delhi bring in $102,601.72 and $81,027.70, respectively.

Recommendation:

- Tailor marketing efforts toward Mumbai and Kolkata, as these cities generate the most significant revenue. Consider running targeted promotions or expanding product availability in these cities to further capitalize on their sales potential. Additionally, explore opportunities to increase revenue in Delhi and Bangalore by enhancing customer engagement or product offerings specific to these regions.

Thank you

رواد مصر الرقمية

وزارة الاتصالات
وتكنولوجيا المعلومات