Supply chain Dataset Analysis

Team members:
-Yasmin tarek
-basant mohamed
-ADHAM TAREK
-OMAR ASHRAF

SUBMITTED TO ENG. JAKIE ABOALEAM

Team Contributions

1-Yasmin : (Team Leader)

- Responsible for the creation of the database and data preprocessing. Yasmin utilized Python for cleaning and preparing the dataset, ensuring data quality and consistency.

- Developed and formulated the analysis questions using Python, focusing on extracting insights and patterns from the data.

2-Adham :

- Worked on developing the final presentation, ensuring that it accurately captured the project's objectives, methodology, and findings.

- Contributed to the visualization process using Python, creating visual plots and graphs to illustrate key insights from the data.

3-Bassant :

- Led the Tableau visualization component of the project, building interactive dashboard to display the analysis results and forecasting insights.

- Designed and implemented dynamic visualizations that supported data-driven decision-making and exploration.

4-Omar:

- Managed the documentation process, ensuring that each phase of the project was thoroughly recorded and explained.

- Developed additional analysis questions using SQL, focusing on extracting and querying relevant data from the database for deeper insights.

Contents

Data source:

Source: Kaggle
Dataset Title: Supply Chain Dataset
Description: This dataset provides comprehensive details of supply chain operations, including product demand, inventory levels, and supply chain performance metrics. It includes information relevant for supply chain management analytics, such as lead times, inventory costs, transportation times, and other related factors.
Usage: This dataset is used for modeling and analyzing supply chain processes, helping to optimize logistics, improve inventory management, and enhance overall supply chain efficiency.

Link

https: www.kaggle.com datasets amirmotefaker supply-chain-dataset discussion 534827

Dataset Features :

This dataset is based on the supply chain of Makeup products.

The supply chain dataset includes various features essential for analyzing and optimizing supply chain processes. Below is a breakdown of each feature :

-Product Type.

-SKU: Stock Keeping Unit, a unique identifier for each product.

--Price: The selling price of the product.

-Availability: Whether the product is available in stock or back-ordered.

-Number of Products Sold: Total units sold over a specific period.

-Revenue Generated: Revenue earned from the sale of each product.

-Customer Demographics: Data on customer characteristics, such as age, gender, and location.

-stock Levels: Current levels of inventory for each product.

-Lead Times: Time required to replenish stock (business and manufacturing lead times).

Order Quantities: Quantities ordered by customers or suppliers.

-Shipping Times: Time taken for shipping orders.

-Shipping Carriers: Information on carriers used for transportation.

-Shipping Costs: Costs associated with shipping orders.

-Supplier Name: Name of the supplier providing the product.

-Location: Supplier s location details.

-Business Lead Time: The time required for business processes to fulfill an order.

-Production Volumes: Quantities produced in each manufacturing cycle.

-Manufacturing Lead Time: Time taken for manufacturing processes.

-Manufacturing Costs: Costs associated with manufacturing products.

_Inspection Results: Outcome of quality checks on products.

-Defect Rates: Rate of defective products identified during inspections.

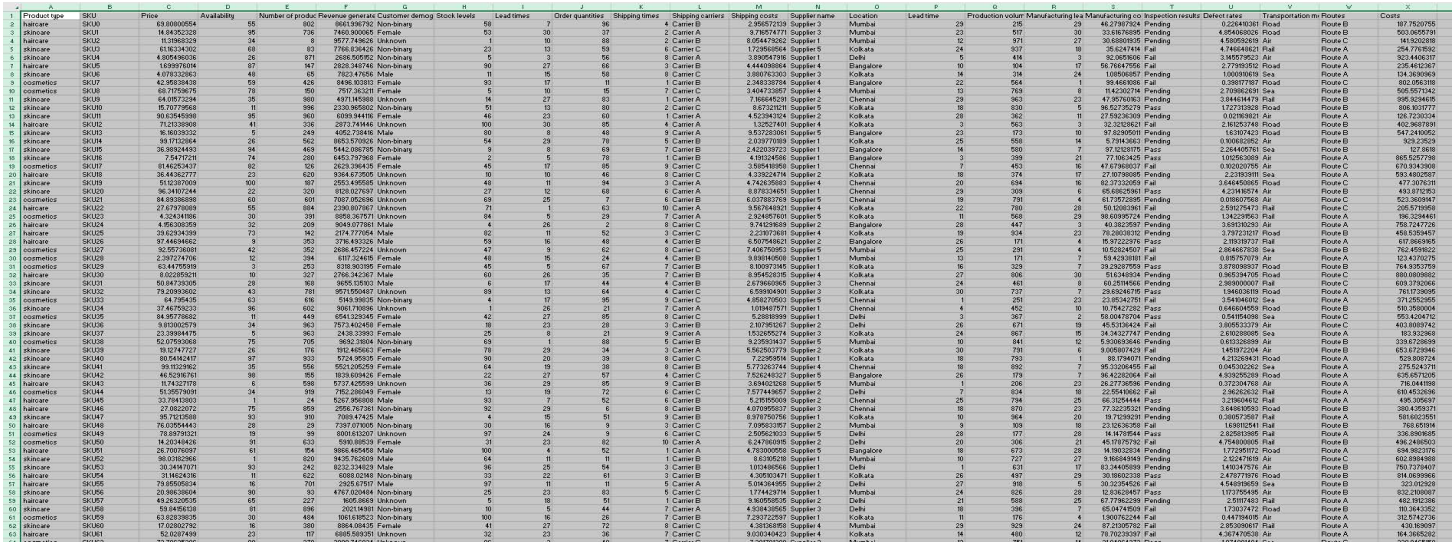
-Transportation Modes: Modes used for transportation (e.g., air, sea, rail, road).

-Routes: Shipping routes used for product delivery.

-Transportation Costs: Costs incurred for using various transportation methods.

Dataset Structure:

The dataset is structured as a single, integrated table containing all relevant supply chain information. This includes product details, sales data, customer demographics, inventory levels, supplier information,logistics data, and financial metrics. However, for efficient data management and analysis, this table needs to be normalized.



Product type	SKU	Price	Availability	Number of produc	Revenue generate	Customer demog	Stock levels	Lead times	Order quantities	Shipping times	Shipping carriers	Shipping costs	Supplier name	Location	Lead time	Production volum	Manufacturing lea	Manufacturing co	Inspection results	Defect rates	Transportation m	Routes	Costs
haircare	SKU0	69.80800654	55	802	8661996792	Non-binary	58	7	96	4	Carrier B	2.956572139	Supplier 3	Mumbai	29	215	29	46.27907924	Pending	0.22640381	Road	Route B	187.7520755
skincare	SKU1	14.84752328	95	736	7460.900065	Female	53	30	37	2	Carrier A	9.79574771	Supplier 3	Mumbai	23	517	30	33.6876895	Pending	4.854698926	Road	Route B	503.0655791
haircare	SKU2	11.39683229	34	8	9577.749626	Unknown	1	10	88	2	Carrier B	8.654479262	Supplier 1	Mumbai	12	971	27	30.68801835	Pending	4.580552619	Air	Route C	141.9202888
skincare	SKU3	61.16334302	68	83	7766.836426	Non-binary	23	13	59	6	Carrier C	1.729569564	Supplier 5	Kolkata	24	937	18	35.6247414	Fail	4.746849621	Rail	Route A	254.7767892
skincare	SKU4	6.805496036	26	871	2686.505762	Non-binary	5	3	56	5	Carrier A	3.890547916	Supplier 1	Delhi	5	416	12	92.0659506	Fail	3.985579523	Air	Route A	923.4408317
haircare	SKU5	1.699976094	87	147	2826.348746	Non-binary	90	27	66	3	Carrier B	4.444093864	Supplier 4	Bangalore	10	904	17	56.76647956	Fail	2.779953552	Road	Route A	235.4612367
skincare	SKU6	4.078332863	48	65	7823.47656	Male	11	15	56	8	Carrier C	3.880763393	Supplier 3	Kolkata	14	314	24	1.08596957	Pending	1.000818618	Sea	Route A	134.3690963
cosmetics	SKU7	42.95838438	59	426	8496.80883	Female	93	17	11	1	Carrier B	2.348338784	Supplier 4	Bangalore	22	564	1	99.4661686	Fail	0.39817787	Road	Route C	802.056318
cosmetics	SKU8	68.71753675	78	150	7517.363121	Female	5	10	15	7	Carrier C	3.404733857	Supplier 4	Mumbai	13	769	8	11.42302714	Pending	2.709862631	Sea	Route B	505.5571342
skincare	SKU9	64.09573294	35	969	4971.145986	Unknown	14	27	83	1	Carrier A	7.166645291	Supplier 2	Chennai	29	963	23	47.9576963	Pending	3.844614479	Rail	Route B	995.3294635
skincare	SKU10	15.70779568	11	996	2330.965802	Non-binary	51	13	80	2	Carrier C	5.87321211	Supplier 5	Kolkata	18	830	5	96.52735279	Pass	1.727313928	Road	Route B	898.1031777
skincare	SKU11	90.63545598	95	960	6093.94416	Female	46	23	60	1	Carrier A	4.523943124	Supplier 2	Kolkata	28	362	11	27.59236309	Pending	0.02763921	Air	Route A	126.7230334
haircare	SKU12	71.2133908	41	336	2972.744146	Unknown	100	30	85	4	Carrier A	1.32527401	Supplier 4	Kolkata	3	563	3	32.3232621	Fail	2.161253748	Road	Route B	402.9687891
skincare	SKU13	16.9033332	5	249	4052.738416	Male	80	8	48	9	Carrier A	9.537283061	Supplier 5	Bangalore	23	173	10	97.82905011	Pending	1.83997423	Road	Route B	547.2490852
skincare	SKU14	99.1713364	26	562	8053.570926	Non-binary	54	29	79	5	Carrier B	2.039770169	Supplier 1	Kolkata	25	558	14	5.7914.3663	Pending	0.906832952	Air	Route B	929.2529
skincare	SKU15	36.88924493	94	469	5442.086785	Non-binary	9	8	69	7	Carrier B	2.422039723	Supplier 1	Bangalore	14	580	7	97.1218375	Pass	2.264405761	Sea	Route B	127.8618
skincare	SKU16	7.5417231	74	280	6453.797968	Female	2	5	79	1	Carrier B	4.19124566	Supplier 1	Bangalore	3	399	21	77.0863425	Pass	1.002563068	Air	Route A	865.5257798
cosmetics	SKU17	81.46253437	82	126	2629.396435	Female	45	17	85	9	Carrier C	3.595419858	Supplier 1	Chennai	7	453	16	47.67968037	Fail	0.302020755	Air	Route C	670.9343908
haircare	SKU18	36.44362777	23	620	9364.673505	Unknown	10	10	46	8	Carrier C	4.339224714	Supplier 2	Kolkata	18	374	17	27.80798085	Pending	2.23193911	Sea	Route A	593.4802587
skincare	SKU19	51.12397609	100	167	2563.495585	Unknown	48	11	94	3	Carrier A	4.742635883	Supplier 4	Chennai	20	694	16	62.2732059	Fail	3.644450965	Road	Route C	477.3076311
skincare	SKU20	96.34107244	22	320	9128.027697	Unknown	27	12	68	6	Carrier A	8.878334651	Supplier 1	Chennai	29	309	6	65.66625961	Pass	4.231416574	Air	Route B	493.6712953
cosmetics	SKU21	84.68286888	60	691	7007.052696	Unknown	69	25	7	6	Carrier B	6.037883769	Supplier 5	Chennai	19	791	4	61.73572895	Pending	0.088607568	Air	Route C	523.3609947
haircare	SKU22	27.67978069	55	884	2390.807967	Unknown	71	1	63	10	Carrier A	9.567649321	Supplier 4	Kolkata	22	780	28	50.1208361	Fail	2.591275473	Rail	Route C	205.5719958
cosmetics	SKU23	4.324341866	30	391	9858.367571	Unknown	84	5	29	7	Carrier A	2.924857601	Supplier 5	Kolkata	11	568	29	98.60995724	Pending	1.342229563	Rail	Route A	196.3294461
haircare	SKU24	4.956306359	32	209	9049.077861	Male	4	26	2	8	Carrier C	9.74129569	Supplier 2	Bangalore	28	447	3	40.3827597	Pending	3.59126293	Air	Route A	758.7247726
haircare	SKU25	29.62394389	73	342	2174.777054	Male	62	11	52	3	Carrier C	2.23973681	Supplier 4	Kolkata	19	934	23	78.28038312	Pending	2.797231217	Road	Route B	459.5359457
haircare	SKU26	97.44634662	9	353	3716.493226	Male	59	16	48	4	Carrier B	6.507548621	Supplier 2	Bangalore	26	171	8	15.97222976	Pass	2.19039737	Rail	Route A	617.8663685
cosmetics	SKU27	92.55736091	42	352	2686.457224	Unknown	47	9	62	8	Carrier C	7.406750953	Supplier 5	Mumbai	25	291	4	10.52824507	Fail	2.864667838	Sea	Route B	762.4599822
cosmetics	SKU28	2.397274706	12	284	6117.324615	Female	48	15	24	4	Carrier B	9.890416506	Supplier 1	Mumbai	13	171	7	59.42838181	Fail	0.185757079	Air	Route A	123.4370275
cosmetics	SKU29	63.44795599	3	253	8318.963195	Female	45	5	67	7	Carrier B	8.190997195	Supplier 1	Kolkata	16	329	7	39.29287959	Pass	3.878998937	Road	Route B	764.9353759
haircare	SKU30	8.02285211	10	327	2766.342367	Male	60	26	35	7	Carrier B	8.954526315	Supplier 4	Kolkata	27	606	30	516349924	Pass	0.965394705	Road	Route C	880.0809882
skincare	SKU31	50.84726095	28	168	9695.15810	Male	6	17	44	4	Carrier B	2.678660965	Supplier 3	Chennai	24	461	8	60.2519566	Pending	2.369900007	Rail	Route C	699.1792066
skincare	SKU32	79.20993662	43	781	9571.550487	Unknown	89	13	64	4	Carrier C	6.59904901	Supplier 3	Kolkata	30	737	7	29.69246715	Pass	1.946036119	Road	Route A	761.1739095
cosmetics	SKU33	64.795435	63	616	5149.99635	Non-binary	4	17	95	9	Carrier C	4.656270563	Supplier 5	Chennai	1	251	23	22.85342751	Fail	3.54594092	Sea	Route A	571.0552955
skincare	SKU34	37.46759233	96	602	9061.79896	Unknown	1	26	21	7	Carrier A	1.01946757	Supplier 1	Chennai	4	452	10	10.75427282	Pass	0.644604559	Road	Route B	510.3589004
cosmetics	SKU35	84.95778662	11	449	6541.329345	Female	42	27	85	8	Carrier C	5.28818999	Supplier 1	Delhi	3	367	2	58.00478704	Pass	0.541554086	Sea	Route C	553.4204752
skincare	SKU36	9.810900579	34	963	7571.402459	Female	18	23	28	3	Carrier B	2.107952067	Supplier 2	Delhi	26	671	19	45.5139424	Fail	3.869533279	Air	Route C	460.8989742
skincare	SKU37	23.39984475	5	963	2438.33993	Female	25	8	21	9	Carrier A	1.532655274	Supplier 3	Kolkata	24	867	15	34.3432747	Pending	2.610289885	Sea	Route A	183.932968
skincare	SKU38	52.07593068	75	705	9692.28804	Non-binary	69	1	88	5	Carrier B	9.235503427	Supplier 5	Mumbai	16	841	12	5.930833646	Pending	0.653328899	Air	Route B	339.6728698
skincare	SKU39	19.9247727	26	176	1912.465663	Female	78	29	34	3	Carrier A	5.562503779	Supplier 2	Kolkata	30	791	6	9.005897429	Fail	1.45197204	Air	Route B	653.6729946
skincare	SKU40	80.54162417	97	933	5724.95935	Female	90	20	39	8	Carrier C	7.22959514	Supplier 1	Kolkata	18	793	1	93.1794071	Pending	4.21326431	Road	Route A	529.809724
skincare	SKU41	99.1523982	26	556	5521.205259	Female	64	19	38	8	Carrier B	5.773263744	Supplier 4	Chennai	18	892	7	95.53206455	Fail	0.045302262	Sea	Route A	275.5241791
skincare	SKU42	46.52916761	98	155	1839.609426	Female	22	27	57	4	Carrier C	7.526240327	Supplier 5	Bangalore	26	179	7	96.42282064	Fail	4.939255289	Road	Route A	626.6571205
haircare	SKU43	11.74327178	6	596	5737.425599	Unknown	36	29	95	9	Carrier B	3.694623268	Supplier 5	Mumbai	1	206	23	26.2736596	Pending	0.372394768	Air	Route A	716.044198
cosmetics	SKU44	51.9579091	34	919	7852.286049	Female	10	19	72	6	Carrier C	7.577449657	Supplier 2	Delhi	7	934	18	22.5540662	Fail	2.36262632	Rail	Route A	610.4532696
skincare	SKU45	33.78418803	1	24	5267.956068	Male	93	7	52	6	Carrier B	5.215150009	Supplier 2	Chennai	25	794	25	66.31254444	Pass	3.21804612	Rail	Route A	495.305697
haircare	SKU46	27.0822072	75	859	2556.767361	Non-binary	92	29	6	8	Carrier B	4.070995837	Supplier 3	Chennai	18	870	23	77.3223532	Pending	3.648836563	Road	Route B	380.4259371
skincare	SKU47	95.7521268	93	910	7089.47425	Male	4	15	51	9	Carrier B	8.978750756	Supplier 1	Kolkata	10	964	20	19.7529291	Pending	0.380573587	Rail	Route A	581.6023551
haircare	SKU48	76.03554442	28	29	7397.67605	Non-binary	39	16	9	3	Carrier C	7.095023957	Supplier 2	Mumbai	9	909	18	23.0263635	Fail	1.69012541	Rail	Route B	765.65194
cosmetics	SKU49	78.89791321	19	99	6001.613207	Unknown	97	24	9	6	Carrier C	2.595623033	Supplier 5	Delhi	28	177	28	14.1479744	Pass	2.825913965	Rail	Route A	336.8909685
cosmetics	SKU50	14.20349426	91	633	5990.89539	Female	31	23	92	10	Carrier A	6.247860935	Supplier 2	Delhi	20	306	21	45.17875792	Fail	4.754800905	Rail	Route B	496.2498503
skincare	SKU51	26.70076097	61	954	9866.485458	Male	100	4	52	1	Carrier A	4.783900558	Supplier 5	Bangalore	18	673	28	14.1903284	Pending	1.772395172	Road	Route A	694.9623076
skincare	SKU52	90.0362966	1	820	9435.782609	Male	64	11	11	1	Carrier B	8.62305218	Supplier 1	Mumbai	10	727	27	9.16884949	Pending	2.52471619	Air	Route C	602.8984988
skincare	SKU53	20.34167071	93	242	8232.334029	Male	96	25	54	3	Carrier B	1.810488566	Supplier 1	Delhi	1	631	17	83.34405899	Pending	1.440147576	Air	Route B	769.7379407
haircare	SKU54	31.1642416	11	622	6088.02148	Non-binary	33	22	61	3	Carrier B	4.305903471	Supplier 1	Kolkata	26	497	29	30.1860238	Pass	2.478771976	Road	Route B	814.0699966
haircare	SKU55	79.95505834	16	701	2925.67557	Male	97	11	11	5	Carrier A	5.014364955	Supplier 2	Delhi	27	918	5	30.2254526	Fail	4.540919659	Sea	Route B	322.012928
skincare	SKU56	29.9863804	90	93	4767.020464	Non-binary	25	23	63	5	Carrier C	1.77442974	Supplier 1	Mumbai	24	826	28	12.8362457	Pass	1.073795495	Air	Route B	832.2169887
haircare	SKU57	49.26320535	65	227	1605.8669	Unknown	5	18	51	1	Carrier B	9.160559525	Supplier 2	Delhi	21	588	25	67.7798299	Pending	2.51117483	Rail	Route A	482.1912386
skincare	SKU58	59.9456123	91	696	2021.14901	Non-binary	19	5	44	7	Carrier A	4.938435565	Supplier 2	Delhi	18	396	7	65.0474509	Fail	1.73037472	Road	Route B	110.3643052
cosmetics	SKU59	63.82839835	30	484	1061.618523	Non-binary	100	16	26	7	Carrier B	7.293722597	Supplier 1	Kolkata	11	176	4	1.90076224	Fail	0.44739405	Air	Route A	312.5742736
skincare	SKU60	17.02802792	16	380	8864.08435	Female	41	27	72	8	Carrier C	4.36136858	Supplier 4	Mumbai	29	929	24	87.2105782	Fail	2.852936617	Rail	Route A	430.1690977
haircare	SKU61	52.0207497	23	117	6885.569351	Unknown	32	23	36	7	Carrier C	9.030340423	Supplier 4	Kolkata	14	460	12	78.7023397	Fail	4.367470538	Air	Route A	164.3685282

Project Objectives :

The objective of this project is to analyze a supply chain dataset by building a data model,performing data cleaning and preprocessing, and deriving insights through analysis, and visualization. The project will be divided into three key phases:

1-Data Model Development, Cleaning, and Preprocessing :

Objective: Build a structured data model and clean the dataset to prepare it for further analysis. This step ensures data consistency, quality, and readiness for analytical tasks.
Tools:
SQL : For Data modeling.
Python (pandas, Matplotlib) :For cleaning and preprocessing.

2-Formulating Analysis Questions :

Objective: Identify key analysis questions that can be derived from the dataset to provide valuable insights for decision-makers. This includes exploring the impact of various factors (e.g., product categories) on revenue.
Tools:
SQL
Python (pandas, Matplotlib)

3-Building a Visualization Dashboard and Final Presentation :

Objective: Develop a Tableau dashboard to visually present the analysis and forecasting insights. The final report and presentation will summarize the project work, including data analysis, model development, and insights gained.
Tools: SQL, Python (pandas, Matplotlib), Tableau.
Deliverables: An interactive visualization dashboard, a comprehensive final report, and a presentation showcasing the project's findings.

organization's strategic interests.

Data Preprocessing

For the data preprocessing phase, we utilized Python to clean and prepare the dataset for analysis.We used (Pandas & Numby)Python's libraries

1-Data Import into Python

Loading Data Set

```
import pandas as pd
import numpy as np

data = pd.read_csv('supply_chain_data.csv')
```

2-Data exploration

Data Exploration

Feature	Description
Product type	The type of product. Options: skincare, haircare, or cosmetics.
SKU	Unique alphanumeric identifiers for each product.
Price	The selling price of each product.
Availability	The quantity of each product that is available for sale.
Number of products sold	The quantity of products sold.
Revenue generated	The revenue generated by each product.
Customer demographics	Information about customer demographics. Options: Female, Male, Non-binary, or Unknown.
Stock levels	Number of units physically present in the store. Availability can be more than stock levels if additional units are on order or in transit.
Lead times/ Business lead time	The time taken to replenish stock, measured in days. Includes shipping time.
Order quantities	The amount of items ordered.
Shipping times	The time required for shipping to the business.
Shipping carriers	The different carriers used by the business for collections. Options: Carrier A, Carrier B, or Carrier C.
Shipping costs	The costs of shipping what the business ordered, including packaging, handling, and delivery charges.
Supplier name	The names of different suppliers.
Location	The locations of the suppliers. Options: Mumbai, Kolkata, Delhi, Bangalore, or Chennai.
Lead time/ Supplier lead time	The time taken for the supplier to replenish stock, measured in days.
Production volumes	The amount of items produced by each supplier.
Manufacturing lead time	The time required for manufacturing by each supplier.
Manufacturing costs	The costs associated with manufacturing.
Inspection results	The outcome of product inspections. Options: Pending, Fail, or Pass.
Defect rates	The percentage of defects found in the products.
Transportation modes	The method used by the supplier to deliver goods to the business. Options: Road, Air, Rail, or Sea.
Routes	The route used by the supplier for shipment. Options: Route A, Route B, or Route C.
Costs / transportation cost	The costs associated with transport, including fuel, vehicle maintenance, driver wages, tolls, and insurance.

3- display the first 10 rows of the dataset with custom CSS styling, setting the background color to "#FFBECA", text color to black, and solid black border.

```
data.head(10).style.set_properties(**{'background-color':'#FFBECA','color':'black','border':'2px solid black'})
```

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times	Order quantities	Shipping times	Shipping carriers	Shipping costs	Supplier name	Location	Lead time
0	haircare	SKU0	69.808006	55	802	8661.996792	Non-binary	58	7	96	4	Carrier B	2.956572	Supplier 3	Mumbai	29
1	skincare	SKU1	14.843523	95	736	7460.900065	Female	53	30	37	2	Carrier A	9.716575	Supplier 3	Mumbai	23
2	haircare	SKU2	11.319683	34	8	9577.749626	Unknown	1	10	88	2	Carrier B	8.054479	Supplier 1	Mumbai	12
3	skincare	SKU3	61.163343	68	83	7766.836426	Non-binary	23	13	59	6	Carrier C	1.729569	Supplier 5	Kolkata	24
4	skincare	SKU4	4.805496	26	871	2686.505152	Non-binary	5	3	56	8	Carrier A	3.890548	Supplier 1	Delhi	5
5	haircare	SKU5	1.699976	87	147	2828.348746	Non-binary	90	27	66	3	Carrier B	4.444099	Supplier 4	Bangalore	10
6	skincare	SKU6	4.078333	48	65	7823.476560	Male	11	15	58	8	Carrier C	3.880763	Supplier 3	Kolkata	14
7	cosmetics	SKU7	42.958384	59	426	8496.103813	Female	93	17	11	1	Carrier B	2.348339	Supplier 4	Bangalore	22
8	cosmetics	SKU8	68.717597	78	150	7517.363211	Female	5	10	15	7	Carrier C	3.404734	Supplier 4	Mumbai	13
9	skincare	SKU9	64.015733	35	980	4971.145988	Unknown	14	27	83	1	Carrier A	7.166645	Supplier 2	Chennai	29

4-prints the shape of the dataset, which contains 100 rows and 24 columns, and then provides detailed information about the dataset's structure.

```
print(f'Shape of the dataset: {data.shape}')
```

```
Shape of the dataset: (100, 24)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 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: 18.9+ KB
```

5-Use data.describe() to provide an overview of measures such as count, mean, standard deviation, minimum, maximum, and quartiles for each numerical column.

```
data.describe()
```

	Price	Availability	Number of products sold	Revenue generated	Stock levels	Lead times	Order quantities	Shipping times	Shipping costs	Lead time	Production volumes	Manufacturing lead time	Manufact
count	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.00000	100.0(
mean	49.462461	48.400000	460.990000	5776.048187	47.770000	15.960000	49.220000	5.750000	5.548149	17.080000	567.840000	14.77000	47.2(
std	31.168193	30.743317	303.780074	2732.841744	31.369372	8.785801	26.784429	2.724283	2.651376	8.846251	263.046861	8.91243	28.9(
min	1.699976	1.000000	8.000000	1061.618523	0.000000	1.000000	1.000000	1.000000	1.013487	1.000000	104.000000	1.00000	1.0(
25%	19.597823	22.750000	184.250000	2812.847151	16.750000	8.000000	26.000000	3.750000	3.540248	10.000000	352.000000	7.00000	22.9(
50%	51.239831	43.500000	392.500000	6006.352023	47.500000	17.000000	52.000000	6.000000	5.320534	18.000000	568.500000	14.00000	45.9(
75%	77.198228	75.000000	704.250000	8253.976921	73.000000	24.000000	71.250000	8.000000	7.601695	25.000000	797.000000	23.00000	68.6(
max	99.171329	100.000000	996.000000	9866.465458	100.000000	30.000000	96.000000	10.000000	9.929816	30.000000	985.000000	30.00000	99.4(

6-Check for null values

Data Preprocessing

```
# 1. Check for missing values
print("Missing Values in each column:")
missing_values = data.isnull().sum()
print(missing_values)
```

```
Missing Values in each column:
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
```

```
data.duplicated().any()
```

```
False
```

5-The code identifies numerical columns in the dataset, calculates the interquartile range (IQR), defines outlier boundaries, flags rows with outliers, and outputs the rows with any detected outliers, though no outliers were found in this case.

```
# Step 1: Identify numerical columns
numerical_columns = data.select_dtypes(include=['float64', 'int64']).columns

# Step 2: Calculate Q1 (25th percentile), Q3 (75th percentile) and IQR for each numerical column
Q1 = data[numerical_columns].quantile(0.25)
Q3 = data[numerical_columns].quantile(0.75)
IQR = Q3 - Q1

# Step 3: Define outlier boundaries
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Step 4: Flag outliers
outliers = (data[numerical_columns] < lower_bound) | (data[numerical_columns] > upper_bound)

# Step 5: Output rows with any outliers
outliers_data = data[outliers.any(axis=1)]

# Show the result
print("Outliers found in the following rows:")
print(outliers_data)
```

```
Outliers found in the following rows:
Empty DataFrame
Columns: [Product type, SKU, Price, Availability, Number of products sold, Revenue generated, Customer demographics, Stock levels, Lead times, Order quantities, Shipping times, Shipping carriers, Shipping costs, Supplier name, Location, Lead time, Production volumes, Manufacturing lead time, Manufacturing costs, Inspection results, Defect rates, Transportation modes, Routes, Costs]
Index: []

[0 rows x 24 columns]
```

6-The code standardizes column names by converting them to lowercase and replacing spaces with underscores, and renames specific columns for clarity, such as changing "lead_times" to "business_lead_time" and "costs" to "transportation_costs."

```
data.columns = data.columns.str.lower().str.replace(' ','_')
data.columns
```

```
Index(['product_type', 'sku', 'price', 'availability',
       'number_of_products_sold', 'revenue_generated', 'customer_demographics',
       'stock_levels', 'lead_times', 'order_quantities', 'shipping_times',
       'shipping_carriers', 'shipping_costs', 'supplier_name', 'location',
       'lead_time', 'production_volumes', 'manufacturing_lead_time',
       'manufacturing_costs', 'inspection_results', 'defect_rates',
       'transportation_modes', 'routes', 'costs'],
      dtype='object')
```

```
data = data.rename(columns={'lead_times': 'business_lead_time', 'lead_time':'supplier_lead_time', 'costs': 'transportation_costs'})
data.columns
```

```
Index(['product_type', 'sku', 'price', 'availability',
       'number_of_products_sold', 'revenue_generated', 'customer_demographics',
       'stock_levels', 'business_lead_time', 'order_quantities',
       'shipping_times', 'shipping_carriers', 'shipping_costs',
       'supplier_name', 'location', 'supplier_lead_time', 'production_volumes',
       'manufacturing_lead_time', 'manufacturing_costs', 'inspection_results',
       'defect_rates', 'transportation_modes', 'routes',
       'transportation_costs'],
      dtype='object')
```

Structured and Normalized Table:

After applying the modeling and normalization steps, the original integrated dataset has been transformed into a set of related tables, each representing specific aspects of the supply chain. This approach minimizes redundancy and enhances data integrity, allowing for more efficient and accurate analysis. The key tables include:

1-Sales :

-Sku
-Number of products sold
-Revenue generated
-Order quantities
-Product type ID
-Customer demographics ID

```
SELECT* FROM Sales$
```

Results | Messages

sku	number_of_products_sold	revenue_generated	order_quantities	product_type_id	customer_demographics_id
SKU0	802	8661.99679239238	96	1	1
SKU1	736	7460.90006544585	37	2	2
SKU10	996	2330.96580209195	80	2	1
SKU11	960	6099.94411558145	60	2	2
SKU12	336	2873.74144602144	85	1	3
SKU13	249	4052.73841623787	48	2	4
SKU14	562	8653.5709264698	78	2	1
SKU15	469	5442.08678539767	69	2	1
SKU16	280	6453.79796817629	78	2	2
SKU17	126	2629.39643484526	85	3	2
SKU18	620	9364.67350507617	46	1	3
SKU19	187	2553.49558499121	94	2	3
SKU2	8	9577.74962586873	88	1	3
SKU20	320	8128.02769685119	68	2	3
SKU21	601	7087.05269635744	7	3	3
SKU22	884	2390.80786655617	63	1	3
SKU23	391	8858.36757101148	29	3	3
SKU24	209	9049.0778609399	2	1	4
SKU25	142	2174.77705435065	52	1	4
SKU26	353	3716.49332589404	48	1	4
SKU27	352	2686.45722357598	62	3	3
SKU28	394	6117.32461508399	24	3	2
SKU29	253	8318.90319461718	67	3	2
SKU3	83	7766.83642568523	59	2	1
SKU30	327	2766.34236686609	35	1	4
SKU31	168	9655.1351027194	44	2	4
SKU32	781	9571.55048732782	64	2	3
SKU33	616	5149.99835040804	95	3	1
SKU34	602	9061.71089550772	21	2	3
SKU35	449	6541.32934480247	85	3	2
SKU36	963	7573.40245784873	28	2	2
SKU37	963	2438.33993047003	21	2	2

2- Product :

-Sku
-price
-availability
-product
type ID

```
SELECT* FROM Product$
```

150 %

Results | Messages

	sku	price	availability	product_type_id
1	SKU0	69.8080055421158	55	1
2	SKU1	14.8435232750843	95	2
3	SKU10	15.7077956819121	11	2
4	SKU11	90.6354599822887	95	2
5	SKU12	71.2133890753601	41	1
6	SKU13	16.16039331738	5	2
7	SKU14	99.1713286386242	26	2
8	SKU15	36.9892449286269	94	2
9	SKU16	7.54717210979127	74	2
10	SKU17	81.462534369237	82	3
11	SKU18	36.4436277704609	23	1
12	SKU19	51.1238700879647	100	2
13	SKU2	11.3196832930906	34	1
14	SKU20	96.3410724399634	22	2
15	SKU21	84.8938689849508	60	3
16	SKU22	27.679780886502	55	1
17	SKU23	4.32434118586416	30	3
18	SKU24	4.15630835931111	32	1
19	SKU25	39.6293439850926	73	1
20	SKU26	97.4469466178928	9	1
21	SKU27	92.557360812402	42	3
22	SKU28	2.39727470559714	12	3
23	SKU29	63.4475591852073	3	3
24	SKU3	61.1633430164377	68	2
25	SKU30	8.02285921052639	10	1
26	SKU31	50.8473930517187	28	2
27	SKU32	79.2099360156567	43	2
28	SKU33	64.7954350001556	63	3
29	SKU34	37.4675923298425	96	2
30	SKU35	84.9577868163504	11	3

3-shipping ::

-Sku
-Shipping carriers ID
-Shipping times
-Shipping costs

```
SELECT* FROM Shipping$
```

Results | Messages

sku	shipping_carriers_id	shipping_times	shipping_costs
SKU0	2	4	2.95657213943081
SKU1	1	2	9.71657477143131
SKU10	3	2	8.67321121127861
SKU11	1	1	4.52394312431666
SKU12	1	4	1.32527401018452
SKU13	1	9	9.537283061108 34
SKU14	2	5	2.03977018944933
SKU15	2	7	2.4220397232752
SKU16	2	1	4.1913245857055
SKU17	3	9	3.58541895823234
SKU18	3	8	4.33922471411071
SKU19	1	3	4.74263588284188
SKU2	2	2	8.05447926173215
SKU20	1	6	8.87833465092684
SKU21	2	6	6.0378837692183
SKU22	1	10	9.56764892092304
SKU23	1	7	2.92485760114555
SKU24	3	8	9.74129168928437
SKU25	3	3	2.23107368128173
SKU26	2	4	6.50754862107855
SKU27	3	8	7.40675095299807
SKU28	2	4	9.89814050806922
SKU29	2	7	8.10097314539703
SKU3	3	6	1.72956856354343
SKU30	2	7	8.95452831531802
SKU31	2	4	2.67966096498141
SKU32	3	4	6.599104901238 58
SKU33	3	9	4.85827050343664
SKU34	1	7	1.01948757082212
SKU35	3	8	5.28818999032741
SKU36	2	3	2.10795126715908
SKU37	1	9	1.53265527359043

4-shiiping carriers :

-Sku
-Number of products sold
-Revenue generated
-Order quantities
-Product type ID
-Customer demographics ID

```
SELECT* FROM Shipping_carriers$
```

Results | Messages

shipping_carriers	shipping_carriers_id
Carrier A	1
Carrier B	2
Carrier C	3

:

5-Stock:

-Stock ID
-Sku
-Stock levels
-Business lead time

```
SELECT* FROM Stock$
```

esults | Messages

Stock_id	sku	stock_levels	business_lead_time
1	SKU0	58	7
2	SKU1	53	30
3	SKU2	1	10
4	SKU3	23	13
5	SKU4	5	3
6	SKU5	90	27
7	SKU6	11	15
8	SKU7	93	17
9	SKU8	5	10
10	SKU9	14	27
11	SKU10	51	13
12	SKU11	46	23
13	SKU12	100	30
14	SKU13	80	8
15	SKU14	54	29
16	SKU15	9	8
17	SKU16	2	5
18	SKU17	45	17
19	SKU18	10	10
20	SKU19	48	11
21	SKU20	27	12
22	SKU21	69	25
23	SKU22	71	1
24	SKU23	84	5
25	SKU24	4	26
26	SKU25	82	11
27	SKU26	59	16
28	SKU27	47	9
29	SKU28	48	15
30	SKU29	45	5
31	SKU30	60	26
32	SKU31	6	17

6-product types

-Stock ID
-Sku
-Stock levels
-Business lead time

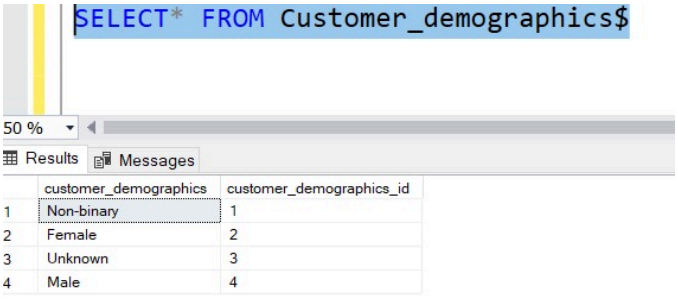
```
SELECT* FROM Product_type$
```

Results | Messages

product_type	product_type_id
haircare	1
skincare	2
cosmetics	3

7-Customer Demographics

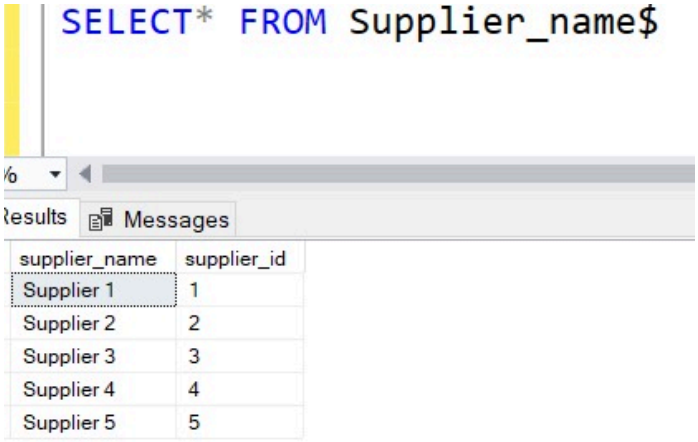
-Customer Demographics
-Customer Demographics ID



	customer_demographics	customer_demographics_id
1	Non-binary	1
2	Female	2
3	Unknown	3
4	Male	4

8-Supplier name :

-Supplier name
-Supplier ID



supplier_name	supplier_id
Supplier 1	1
Supplier 2	2
Supplier 3	3
Supplier 4	4
Supplier 5	5

9-Supplier

-Sku
-Number of products sold
-Revenue generated
-Order quantities
-Product type ID
-Customer demographics ID

```
SELECT* FROM Supplier$
```

sults | Messages

sku	supplier_id	location_id	supplier_lead_time	production_volumes
SKU0	3	1	29	215
SKU1	3	1	23	517
SKU10	5	2	18	830
SKU11	2	2	28	362
SKU12	4	2	3	563
SKU13	5	4	23	173
SKU14	1	2	25	558
SKU15	1	4	14	580
SKU16	1	4	3	399
SKU17	1	5	7	453
SKU18	2	2	18	374
SKU19	4	5	20	694
SKU2	1	1	12	971
SKU20	1	5	29	309
SKU21	5	5	19	791
SKU22	4	2	22	780
SKU23	5	2	11	568
SKU24	2	4	28	447
SKU25	4	2	19	934
SKU26	2	4	26	171
SKU27	5	1	25	291
SKU28	1	1	13	171
SKU29	1	2	16	329
SKU3	5	2	24	937
SKU30	4	2	27	806
SKU31	3	5	24	461
SKU32	3	2	30	737
SKU33	5	5	1	251
SKU34	1	5	4	452
SKU35	1	3	3	367
SKU36	2	3	26	671
SKU37	3	2	24	867

10-Supplier location :

-Location
-Location ID

```
SELECT* FROM Supplier_location$
```

Results | Messages

location	location_id
Mumbai	1
Kolkata	2
Delhi	3
Bangalore	4
Chennai	5

11-Manufacturing

- -Sku
  -N³««}tk® 6 ,
  -Inspection_result ID
  -Manufacturing lead ID()
  - ?^¡ ³p^g²³®t¡ r g£¯²¯
  - , kpkg² ®^²k

```
SELECT* FROM Manufacturing$
```

150 % ▾

Results | Messages

	sku	supplier_id	inspection_results_id	manufacturing_lead_time	manufacturing_costs	defect_rates
1	SKU0	3	1	29	46.2798792405083	0.226410360849925
2	SKU1	3	1	30	33.61676895373	4.85406802638871
3	SKU10	5	3	5	96.5273527853109	1.72731392835594
4	SKU11	2	1	11	27.5923630866637	0.0211698213729943
5	SKU12	4	2	3	32.321286213424	2.16125374755591
6	SKU13	5	1	10	97.8290501101733	1.63107423007154
7	SKU14	1	1	14	5.79143662986299	0.100682851565094
8	SKU15	1	3	7	97.1212817514743	2.26440576119855
9	SKU16	1	3	21	77.10634249785	1.01256308925805
10	SKU17	1	2	16	47.6796803683553	0.102020754918176
11	SKU18	2	1	17	27.1079808548439	2.23193911072926
12	SKU19	4	2	16	82.3733205879902	3.64645086541703
13	SKU2	1	1	27	30.6880193482842	4.58059261919923
14	SKU20	1	3	6	65.6862596084886	4.23141657353454
15	SKU21	5	1	4	61.7357289541609	0.0186075676310149
16	SKU22	4	2	28	50.1208396129773	2.59127547321112
17	SKU23	5	1	29	98.6099572427039	1.34229156272273
18	SKU24	2	1	3	40.3823597029248	3.69131029262873
19	SKU25	4	1	23	78.2803831184154	3.79723121711418
20	SKU26	2	3	4	15.9722297571818	2.11931973672492
21	SKU27	5	2	4	10.5282450700422	2.86466783788337
22	SKU28	1	2	7	59.4293818106916	0.815757079295672
23	SKU29	1	3	7	39.2928755860657	3.87809893658849
24	SKU3	5	2	18	35.624741397125	4.74664862064775
25	SKU30	4	1	30	51.6348934001093	0.965394705352393
26	SKU31	3	1	8	60.2511456615981	2.98900000665508
27	SKU32	3	3	7	29.6924671537498	1.94603611938611
28	SKU33	5	2	23	23.8534275128961	3.54104601225092
29	SKU34	1	3	10	10.7542728150293	0.646604559372055
30	SKU35	1	3	2	58.0047870447438	0.541154098060581
31	SKU36	2	2	19	45.5313642371621	3.80553337924335
32	SKU37	3	1	15	34.3432774650754	2.61028808484811

12-Tansportation :

-Sku
-Supplier ID
-Transportation modes ID
-Routes ID
-Transportation costs

```
SELECT* FROM Transportation$
```

% ▾

Results | Messages

sku	supplier_id	transportation_modes_id	routes_id	transportation_costs
SKU0	3	1	2	187.752075459204
SKU1	3	1	2	503.065579149669
SKU10	5	1	2	806.103177702923
SKU11	2	2	1	126.723033409407
SKU12	4	1	2	402.968789073771
SKU13	5	1	2	547.241005160968
SKU14	1	2	2	929.23528996089
SKU15	1	4	2	127.861800001625
SKU16	1	2	1	865.52577977124
SKU17	1	2	3	670.93439079241
SKU18	2	4	1	593.480258720652
SKU19	4	1	3	477.307631090903
SKU2	1	2	3	141.920281771519
SKU20	1	2	2	493.871215316206
SKU21	5	2	3	523.360914720158
SKU22	4	3	3	205.571995826947
SKU23	5	3	1	196.329446112413
SKU24	2	2	1	758.724772602938
SKU25	4	1	2	458.535945739209
SKU26	2	3	1	617.866916458377
SKU27	5	4	2	762.459182155684
SKU28	1	2	1	123.437027511827
SKU29	1	1	2	764.935375940708
SKU3	5	3	1	254.776159219287
SKU30	4	1	3	880.080988247161
SKU31	3	3	3	609.379206618427
SKU32	3	1	1	761.173909514878
SKU33	5	4	1	371.255295519871
SKU34	1	1	2	510.358000433524
SKU35	1	4	3	553.420471230356
SKU36	2	2	3	403.808974248181
SKU37	3	4	1	183.932968043594

13- Routes

-Routes
-Routes ID

```
SELECT* FROM routes$
```

0 %

Results | Messages

routes	routes_id
Route A	1
Route B	2
Route C	3

14-transportation modes

-transportation modes
-transportation modes ID

```
SELECT* FROM transportation_modes$
```

%

Results | Messages

transportation_modes	transportation_modes_id
Road	1
Air	2
Rail	3
Sea	4

15-inspection results

-inpection results
-inpection results ID

```
SELECT* FROM inspection_results$
```

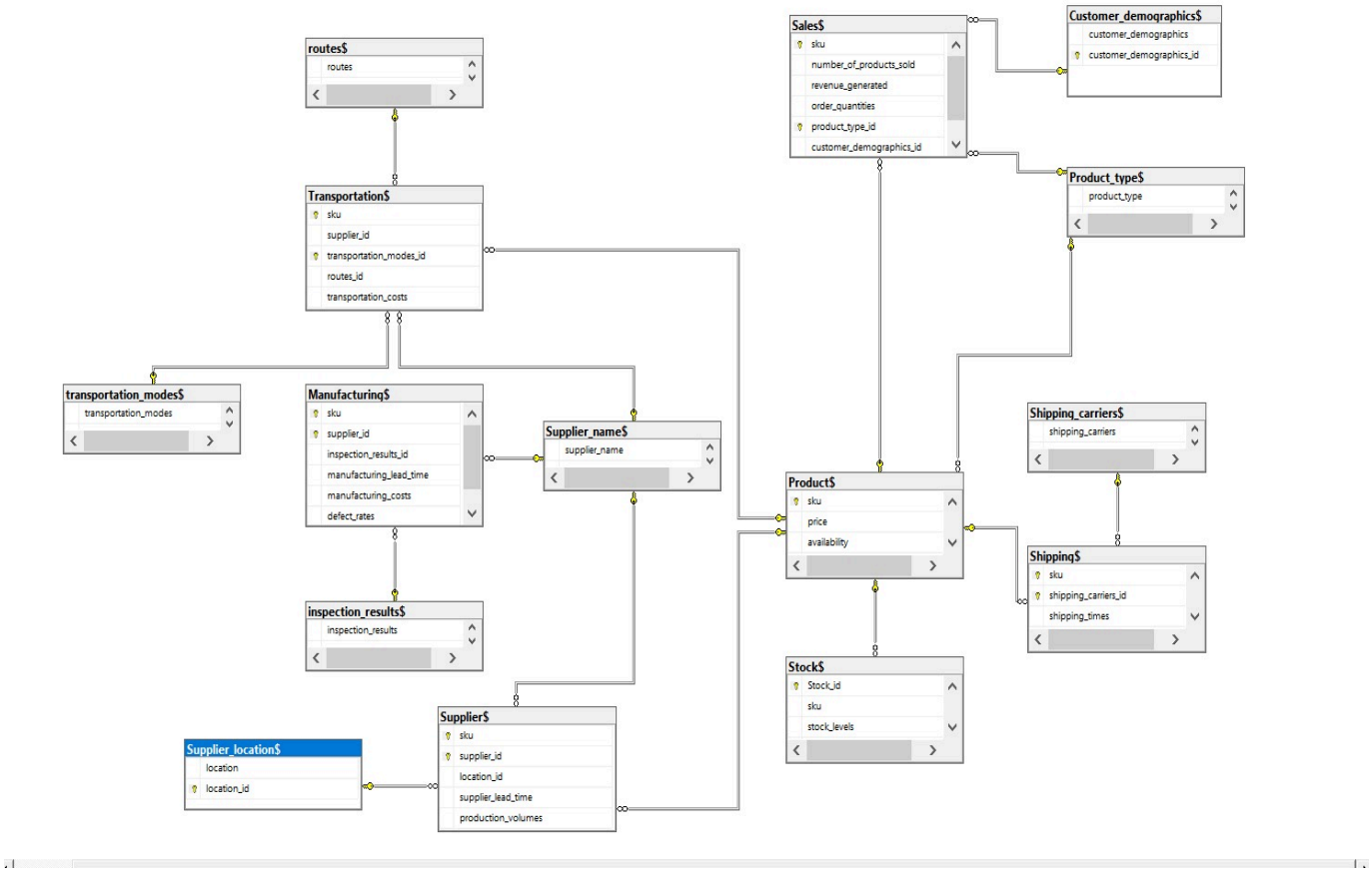
150 %

Results | Messages

	inspection_results	inspection_results_id
1	Pending	1
2	Fail	2
3	Pass	3

Data Model :

To build and implement the data model for the supply chain dataset, we used SQL Server Management Studio (SSMS). SSMS provided a comprehensive environment for designing, managing, and visualizing the database schema. The tool facilitated the creation of normalized tables, relationships, and constraints, ensuring a structured and efficient database model



Constraints :

1.-routes$

- Primary Key: routes_id

2.-Transportation$

- Primary Key: sku, supplier_id

oForeign Keys:

- transportation_modes_id → transportation_modes$ (transportation_modes_id)
- routes_id → routes$ (routes_id)

3.-transportation_modes$

- Primary Key: transportation_modes_id

4.-Manufacturing$

- Primary Key: sku, supplier_id
- Foreign Keys:
  - supplier_id → Supplier$ (supplier_id)

5.-Supplier_name$

- Primary Key: supplier_id

6.-Sales$

- Primary Key: sku
- Foreign Keys:
  - product_type_id → Product_type$ (product_type_id)
  - customer_demographics_id → Customer_demographics$ (customer_demographics_id)

7. Product$
    - Primary Key: sku
8. Shipping$
    - Primary Key: sku
    - Foreign Keys:
        - shipping_carriers_id → Shipping_carriers$ (shipping_carriers_id)
9. Stock$
    - Primary Key: Stock_id
    - Foreign Key:
        - sku → Product$ (sku)
10. Supplier_location$
    - Primary Key: location_id
11. Supplier$
    - Primary Key: supplier_id, sku
    - Foreign Key:
        - location_id → Supplier_location$ (location_id)
12. Shipping_carriers$
    - Primary Key: shipping_carriers_id
13. Customer_demographics$
    - Primary Key: customer_demographics_id
14. Product_type$
    - Primary Key: product_type_id
15. Inspection_results$
    - Primary Key: inspection_results_id

Analysis Questions

Throughout the project, several key analysis questions were formulated and answered using tools such as Python and SQL. These questions were designed to provide insights into the supply chain operations and help decision-makers optimize processes. The following are the main questions addressed:

SQL :

1. What is the total revenue generated per product type?

This question helps understand which product categories (e.g., skincare, haircare) are generating the most revenue.

We find that skincare category generated the most revenue.

```
SELECT pt.product_type, SUM(s.revenue_generated) AS total_revenue
FROM Sales$ s
JOIN Product_type$ pt ON s.product_type_id = pt.product_type_id
GROUP BY pt.product_type
ORDER BY total_revenue DESC;
```

150 %

Results | Messages

	product_type	total_revenue
1	skincare	241628.162133063
2	haircare	174455.390605462
3	cosmetics	161521.265999483

```
);
```

2.Which product SKU generated the highest revenue?

This will help identify the top-performing product in terms of revenue.
We find that Product SKU with the highest revenue is SKU51 with revenue 9866.465458

```
SELECT TOP 1 p.sku, SUM(s.revenue_generated) AS total_revenue
FROM Sales$ s
JOIN Product$ p ON s.sku = p.sku
GROUP BY p.sku
ORDER BY total_revenue DESC
```

3. Which product SKU has sold the highest number of units?

This will help identify the best-selling product in terms of units sold.

We find that Product SKU10 has sold the highest number of units (996).

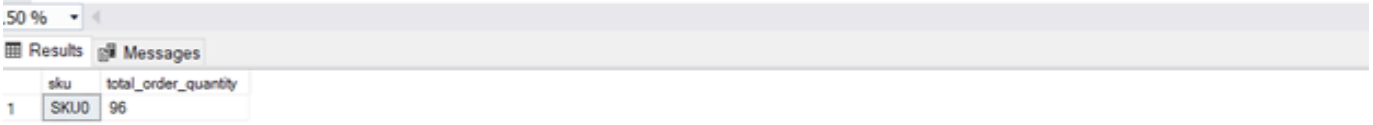
```
SELECT TOP 1 p.sku, SUM(s.number_of_products_sold) AS total_units_sold
FROM Sales$ s
JOIN Product$ p ON s.sku = p.sku
GROUP BY p.sku
ORDER BY total_units_sold DESC
```

4. Which product has the highest order quantity?

This helps determine which products are ordered in bulk by customers.

We find that Product SKU0 has the highest order quantity (96 ORDER)

```
SELECT TOP 1 p.sku, SUM(s.order_quantities) AS total_order_quantity
FROM Sales$ s
JOIN Product$ p ON s.sku = p.sku
GROUP BY p.sku
ORDER BY total_order_quantity DESC
```



Results Messages

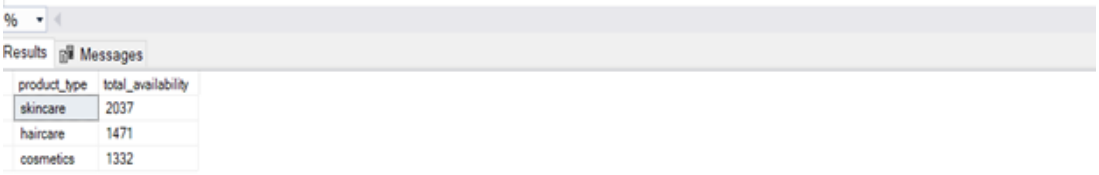
	sku	total_order_quantity
1	SKU0	96

5. Which product type has the highest availability in stock?

This question helps identify which product type has the most inventory available for sale.

We find that skincare has the highest availability in stock (203)

```
SELECT pt.product_type, SUM(p.availability) AS total_availability
FROM Product$ p
JOIN Product_type$ pt ON p.product_type_id = pt.product_type_id
GROUP BY pt.product_type
ORDER BY total_availability DESC;
```



Results Messages

product_type	total_availability
skincare	2037
haircare	1471
cosmetics	1332

6. Which customer demographic group buys the most products?

This question helps understand the customer demographics associated with the highest sales.(UNKNOWN)

```
SELECT cd.customer_demographics, SUM(s.number_of_products_sold) AS total_products_sold
FROM Sales$ s
JOIN Customer_demographics$ cd ON s.customer_demographics_id = cd.customer_demographics_id
GROUP BY cd.customer_demographics
ORDER BY total_products_sold DESC;
```

Results Messages

customer_demographics	total_products_sold
Unknown	15211
Female	12801
Non-binary	10580
Male	7507

7. What is the average shipping time for each carrier?

This helps understand the efficiency of each shipping carrier based on average shipping times.

Fastest shipping carrier: Carrier B

```
SELECT sc.shipping_carriers, AVG(sh.shipping_times) AS avg_shipping_time
FROM Shipping$ sh
JOIN Shipping_carriers$ sc ON sh.shipping_carriers_id = sc.shipping_carriers_id
GROUP BY sc.shipping_carriers
ORDER BY avg_shipping_time;
```

Results Messages

shipping_carriers	avg_shipping_time
Carrier B	5.30232558139535
Carrier C	6.03448275862069
Carrier A	6.14285714285714

10. Which supplier in each location has the highes t production volume?

This can help optimize procurement by identifying suppliers with the most capacity.

```
SELECT sn.supplier_name, sl.location, MAX(s.production_volumes) AS highest_production_volume
FROM Supplier$ s
JOIN Supplier_name$ sn ON s.supplier_id = sn.supplier_id
JOIN Supplier_location$ sl ON s.location_id = sl.location_id
GROUP BY sn.supplier_name, sl.location
ORDER BY highest_production_volume DESC;
```

Results Messages

	supplier_name	location	highest_production_volume
1	Supplier 2	Bangalore	985
2	Supplier 1	Mumbai	971
3	Supplier 1	Kolkata	964
4	Supplier 2	Chennai	963
5	Supplier 2	Mumbai	955
6	Supplier 5	Kolkata	937
7	Supplier 4	Kolkata	934
8	Supplier 4	Mumbai	929
9	Supplier 4	Delhi	919
10	Supplier 2	Delhi	918
11	Supplier 4	Chennai	892
12	Supplier 3	Chennai	870
13	Supplier 3	Kolkata	867
14	Supplier 5	Mumbai	858
15	Supplier 2	Kolkata	791
16	Supplier 5	Chennai	791
17	Supplier 1	Bangalore	775
18	Supplier 1	Chennai	759
19	Supplier 3	Bangalore	736
20	Supplier 3	Delhi	698
21	Supplier 5	Bangalore	673
22	Supplier 3	Mumbai	648
23	Supplier 1	Delhi	631
24	Supplier 4	Bangalore	564
25	Supplier 5	Delhi	177

11. What is the average manufacturing cost per product type ?

This question helps in understanding which product types are the most expensive to manufacture.

```
SELECT pt.product_type, AVG(m.manufacturing_costs) AS avg_manufacturing_cost
FROM Manufacturing$ m
JOIN Product$ p ON m.sku = p.sku
JOIN Product_type$ pt ON p.product_type_id = pt.product_type_id
GROUP BY pt.product_type;
```

50 %

Results Messages

	product_type	avg_manufacturing_cost
1	cosmetics	43.0527404964934
2	haircare	48.4579934206273
3	skincare	48.9931573734209

8. Which supplier has the highest defect rates?

This helps identify suppliers with quality issues based on their defect rates.(Supplier 5)

```
SELECT TOP 1 sn.supplier_name, m.defect_rates
FROM Manufacturing$ m
JOIN Supplier_name$ sn ON m.supplier_id = sn.supplier_id
ORDER BY m.defect_rates DESC
```

Results | Messages

supplier_name	defect_rates
Supplier 5	4.93925528862095

9.Which supplier has the longest lead time per location?

This will help identify which supplier, in each location, takes the longest to fulfill orders.

```
SELECT sn.supplier_name, sl.location, MAX(s.supplier_lead_time) AS longest_lead_time
FROM Supplier$ s
JOIN Supplier_name$ sn ON s.supplier_id = sn.supplier_id
JOIN Supplier_location$ sl ON s.location_id = sl.location_id
GROUP BY sn.supplier_name, sl.location
ORDER BY longest_lead_time DESC;
```

Results | Messages

	supplier_name	location	longest_lead_time
	Supplier 2	Kolkata	30
	Supplier 3	Kolkata	30
	Supplier 3	Mumbai	29
	Supplier 4	Mumbai	29
	Supplier 1	Chennai	29
	Supplier 2	Chennai	29
	Supplier 2	Bangalore	28
	Supplier 5	Chennai	28
	Supplier 3	Delhi	28
0	Supplier 5	Delhi	28
1	Supplier 2	Delhi	27
2	Supplier 3	Chennai	27
3	Supplier 4	Kolkata	27
4	Supplier 5	Bangalore	26
5	Supplier 1	Kolkata	26
6	Supplier 5	Mumbai	25
7	Supplier 5	Kolkata	24
8	Supplier 1	Mumbai	24
9	Supplier 1	Bangalore	22
0	Supplier 4	Bangalore	22
1	Supplier 4	Chennai	21
2	Supplier 2	Mumbai	21
3	Supplier 3	Bangalore	18
4	Supplier 1	Delhi	17
5	Supplier 4	Delhi	5

12.Which route has the highest transportation cost?

This question helps to identify the most expensive shipping route.(Route B)

```
SELECT r.routes, SUM(t.transportation_costs) AS total_cost
FROM Transportation$ t
JOIN routes$ r ON t.routes_id = r.routes_id
GROUP BY r.routes
ORDER BY total_cost DESC;
```

50 %

Results | Messages

	routes	total_cost
1	Route B	22039.3840256014
2	Route A	20875.7744943532
3	Route C	10009.4196958596

Answering Analytical Questions Using Python

Analysis Questions

Total revenue generated for each product type

```
total_revenue_per_product_type = data.groupby('product_type')['revenue_generated'].sum()
print("Total revenue generated for each product type:\n")
print(total_revenue_per_product_type)
```

```
Total revenue generated for each product type:

product_type
cosmetics    161521.265999
haircare     174455.390605
skincare     241628.162133
Name: revenue_generated, dtype: float64
```

Product SKU with the highest revenue

```
highest_revenue_sku = data.loc[data['revenue_generated'].idxmax(), ['sku', 'product_type', 'revenue_generated']]
print("Product SKU with the highest revenue:\n")
print(highest_revenue_sku)
```

```
Product SKU with the highest revenue:

sku                       SKU51
product_type            haircare
revenue_generated    9866.465458
Name: 51, dtype: object
```

Average Lead Time per Supplier and Location

```
avg_lead_time_per_supplier_location = data.groupby(['supplier_name', 'location'])['supplier_lead_time'].mean()

# Display the result
print("Average Lead Time per Supplier and Location:\n")
print(avg_lead_time_per_supplier_location)
```

```
Average Lead Time per Supplier and Location:

supplier_name  location
Supplier 1     Bangalore    16.000000
               Chennai      16.250000
               Delhi         6.500000
               Kolkata      19.375000
               Mumbai       12.166667
Supplier 2     Bangalore    16.600000
               Chennai      27.666667
               Delhi        18.833333
               Kolkata      25.333333
               Mumbai       10.600000
Supplier 3     Bangalore    10.333333
               Chennai      19.250000
               Delhi        23.000000
               Kolkata      22.666667
               Mumbai       26.666667
Supplier 4     Bangalore    16.000000
               Chennai      18.000000
               Delhi         3.000000
               Kolkata      15.666667
               Mumbai       17.500000
Supplier 5     Bangalore    22.333333
               Chennai      15.200000
               Delhi        28.000000
               Kolkata      18.600000
               Mumbai       15.250000
Name: supplier_lead_time, dtype: float64
```

Transportation Mode with the Highest Associated Cost

```
# Calculate the total transportation costs for each transportation mode
total_cost_per_transport_mode = data.groupby('transportation_modes')['transportation_costs'].sum()

# Find the transportation mode with the highest total cost
highest_cost_transport_mode = total_cost_per_transport_mode.idxmax()

# Display the results
print("Total cost per transportation mode:\n")
print(total_cost_per_transport_mode)
print(f"Transportation mode with the highest cost: {highest_cost_transport_mode}")
```

```
Total cost per transportation mode:

transportation_modes
Air     14604.527498
Rail    15168.931559
Road    16048.193639
Sea      7102.925520
Name: transportation_costs, dtype: float64
Transportation mode with the highest cost: Road
```

Supplier with the Highest Defect Rates and Their Manufacturing Lead Times

```
# Find the supplier with the highest defect rates and their manufacturing lead times
supplier_with_highest_defect = data.loc[data['defect_rates'].idxmax(), ['supplier_name', 'defect_rates', 'manufacturing_lead_time']]

# Display the result
print("the supplier with the highest defect rates and their manufacturing lead times:\n")
print(supplier_with_highest_defect)
```

```
the supplier with the highest defect rates and their manufacturing lead times:

supplier_name              Supplier 5
defect_rates                 4.939255
manufacturing_lead_time             7
Name: 42, dtype: object
```

Total Production Volumes by Supplier and Location

```
# Calculate total production volumes by supplier and location
total_production_per_supplier_location = data.groupby(['supplier_name', 'location'])['production_volumes'].sum()

# Display the result
print("total_production_per_supplier_location:\n")
print(total_production_per_supplier_location)
```

```
total_production_per_supplier_location:

supplier_name  location
Supplier 1     Bangalore    2349
               Chennai      1973
               Delhi        1740
               Kolkata      4021
               Mumbai       3462
Supplier 2     Bangalore    2236
               Chennai      2678
               Delhi        4225
               Kolkata      1527
               Mumbai       3439
Supplier 3     Bangalore    1549
               Chennai      2056
               Delhi        1094
               Kolkata      1918
               Mumbai       1380
Supplier 4     Bangalore     668
               Chennai      2854
               Delhi        1126
               Kolkata      4425
               Mumbai       2683
Supplier 5     Bangalore    1025
               Chennai      2423
               Delhi         177
               Kolkata      3560
               Mumbai       2196
Name: production_volumes, dtype: int64
```

Shipping Times for Each Carrier and the Fastest Carrier

```
# Calculate the average shipping times for each carrier
avg_shipping_time_per_carrier = data.groupby('shipping_carriers')['shipping_times'].mean()

# Find the carrier with the fastest average shipping time
fastest_shipping_carrier = avg_shipping_time_per_carrier.idxmin()

# Display the results
print("Average shipping time per carrier:\n")
print(avg_shipping_time_per_carrier)
print(f"Fastest shipping carrier: {fastest_shipping_carrier}")
```

```
Average shipping time per carrier:

shipping_carriers
Carrier A    6.142857
Carrier B    5.302326
Carrier C    6.034483
Name: shipping_times, dtype: float64
Fastest shipping carrier: Carrier B
```

Route with the Highest Transportation Costs

```
# Calculate the total transportation costs for each route
total_cost_per_route = data.groupby('routes')['transportation_costs'].sum()

# Find the route with the highest transportation cost
highest_cost_route = total_cost_per_route.idxmax()

# Display the results
print("Total cost per route:\n")
print(total_cost_per_route)
print(f"Route with the highest transportation cost: {highest_cost_route}")
```

```
Total cost per route:

routes
Route A    20875.774494
Route B    22039.384026
Route C    10009.419696
Name: transportation_costs, dtype: float64
Route with the highest transportation cost: Route B
```

Relationship Between Stock Levels and Number of Products Sold

```
# Calculate the correlation between stock levels and number of products sold
stock_vs_sales_correlation = data['stock_levels'].corr(data['number_of_products_sold'])

# Display the result
print(f"Correlation between stock levels and number of products sold: {stock_vs_sales_correlation}")
```

```
Correlation between stock levels and number of products sold: 0.022189481359586975
```

Average Manufacturing Cost per Product Type

```
# Calculate the average manufacturing cost per product type
avg_manufacturing_cost_per_product_type = data.groupby('product_type')['manufacturing_costs'].mean()

# Display the result
print("avg_manufacturing_cost_per_product_type:\n")
print(avg_manufacturing_cost_per_product_type)
```

```
avg_manufacturing_cost_per_product_type:

product_type
cosmetics    43.052740
haircare     48.457993
skincare     48.993157
Name: manufacturing_costs, dtype: float64
```

Data Visualization

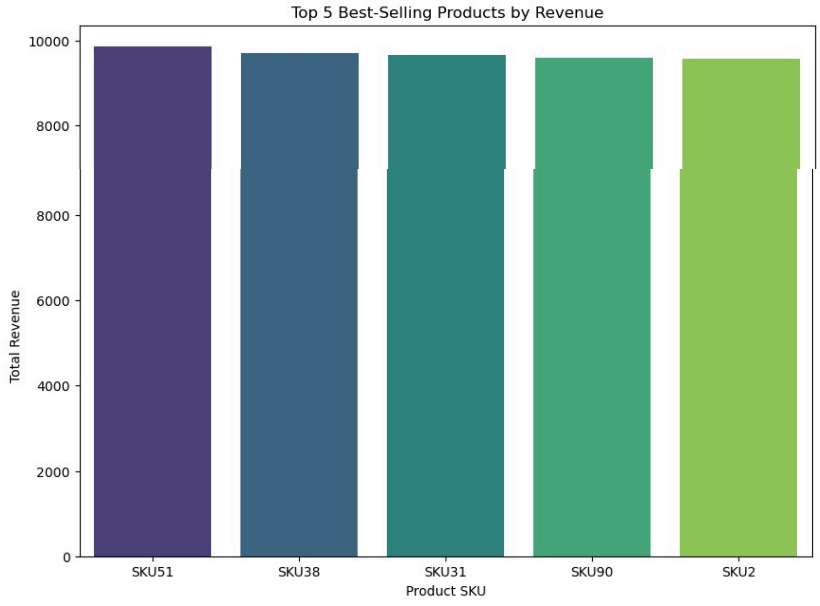
In this phase, we utilized Tableau and Python to create comprehensive visualizations that illustrate key insights derived from the supply chain dataset. The combination of these tools allowed for both interactive dashboard creation and custom visual analysis.

Top 5 Best selling products by revenue

Visualization

```
import matplotlib.pyplot as plt
import seaborn as sns
best_selling = data.groupby('sku')['revenue_generated'].sum().reset_index().sort_values(by='revenue_generated', ascending=False).head(5)

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='sku', y='revenue_generated', data=best_selling, palette='viridis')
plt.title('Top 5 Best-Selling Products by Revenue')
plt.ylabel('Total Revenue')
plt.xlabel('Product SKU')
plt.show()
```

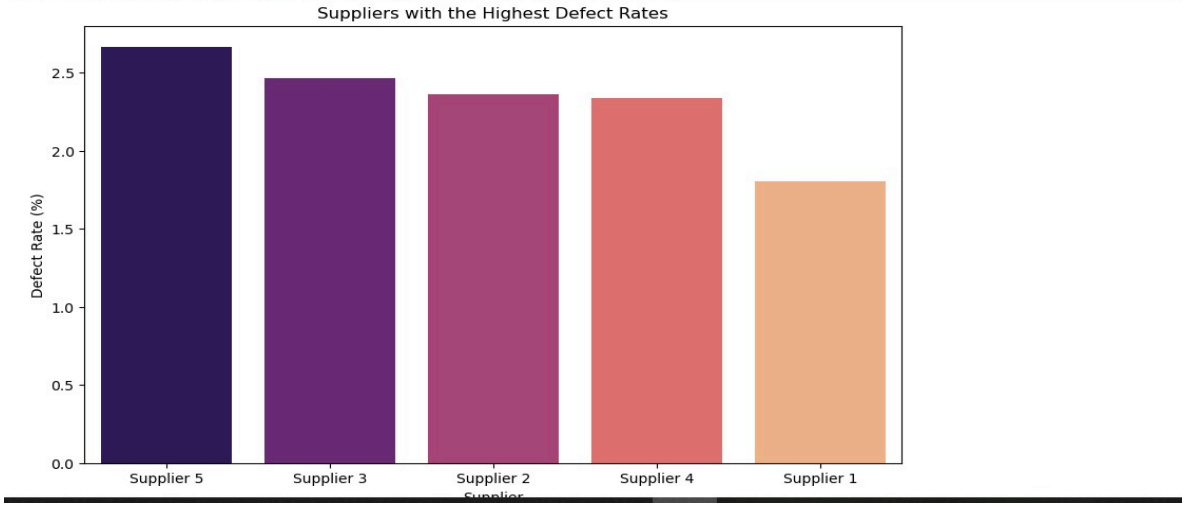
```
C:\Users\STORM\AppData\Local\Temp\ipykernel_9268\3668861183.py:7: FutureWarning:

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` for the same effect.

  sns.barplot(x='sku', y='revenue_generated', data=best_selling, palette='viridis')
```



Suppliers With the Highest defect rates

```
# Group by supplier and calculate average defect rates
defect_rates = data.groupby('supplier_name')['defect_rates'].mean().reset_index().sort_values(by='defect_rates', ascending=False).head(5)
# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='supplier_name', y='defect_rates', data=defect_rates, palette='magma')
plt.title('Suppliers with the Highest Defect Rates')
plt.ylabel('Defect Rate (%)')
plt.xlabel('Supplier')
plt.show()
```

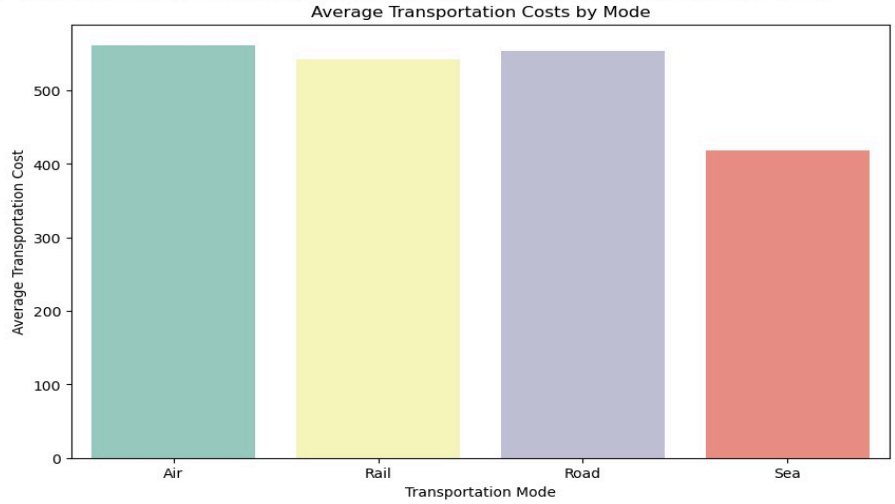
```
C:\Users\STORM\AppData\Local\Temp\ipykernel_9268\833242848.py:6: FutureWarning:

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` for the same effect.

  sns.barplot(x='supplier_name', y='defect_rates', data=defect_rates, palette='magma')
```



Average Transportation Costs by mode

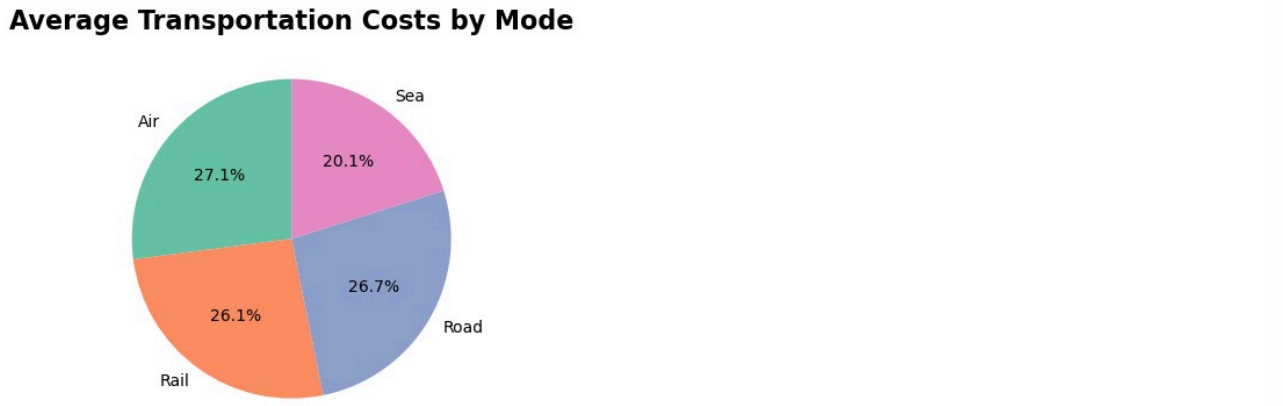
```
transportation_costs = data.groupby('transportation_modes')['transportation_costs'].mean().reset_index()
# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='transportation_modes', y='transportation_costs', data=transportation_costs, palette='Set3')
plt.title('Average Transportation Costs by Mode')
plt.ylabel('Average Transportation Cost')
plt.xlabel('Transportation Mode')
plt.show()
```

```
C:\Users\STORM\AppData\Local\Temp\ipykernel_9268\195855962.py:6: 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` for the same effect.

  sns.barplot(x='transportation_modes', y='transportation_costs', data=transportation_costs, palette='Set3')
```



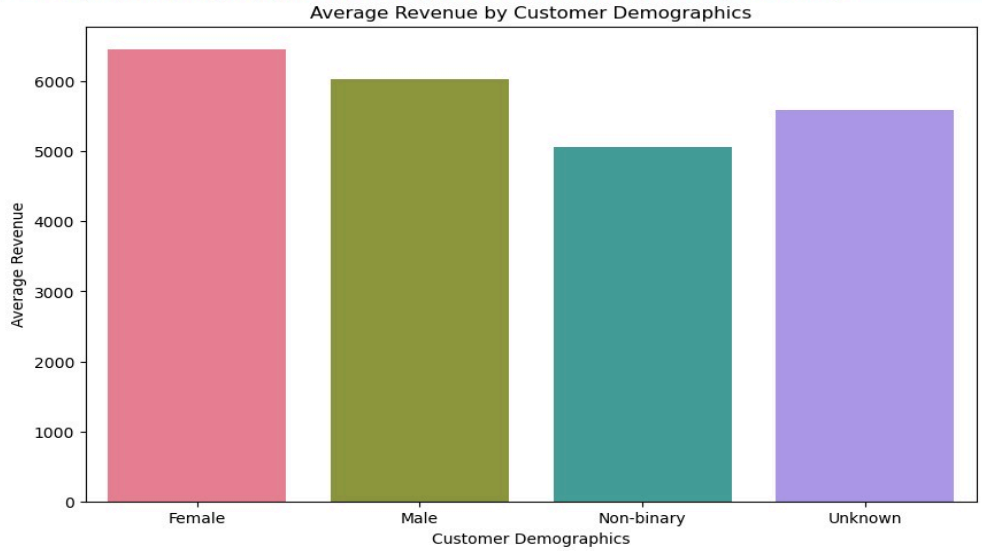
Average Transportation Costs by Mode

```
# Group by Transportation Mode and calculate the average transportation cost
avg_transport_costs = data.groupby('transportation_modes')['transportation_costs'].mean().reset_index()

# Plot a pie chart
plt.figure(figsize=(4, 4))
plt.pie(avg_transport_costs['transportation_costs'], labels=avg_transport_costs['transportation_modes'],
        autopct='%1.1f%%', startangle=90, colors=sns.color_palette('Set2', len(avg_transport_costs)))

# Ensure pie is drawn as a circle
plt.title('Average Transportation Costs by Mode', fontsize=16, fontweight='bold')
plt.tight_layout()
plt.show()
```



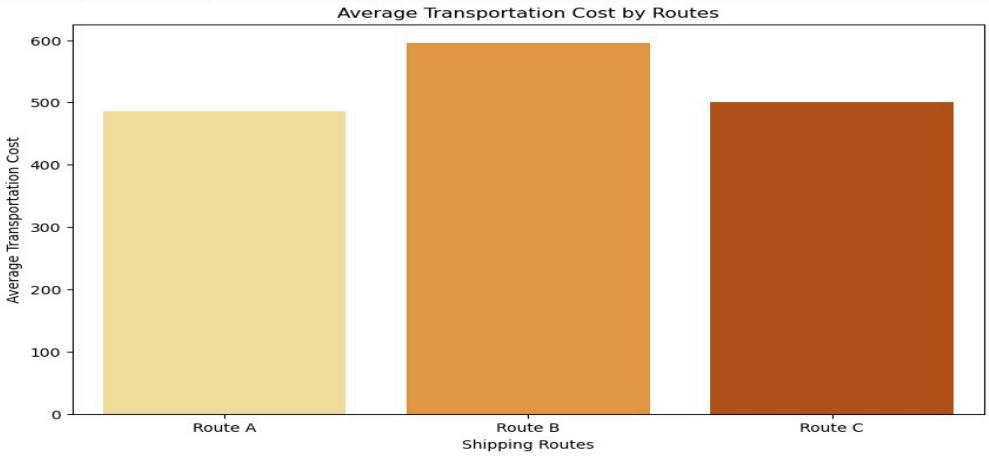

```
customer_revenue = data.groupby('customer_demographics')['revenue_generated'].mean().reset_index()

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='customer_demographics', y='revenue_generated', data=customer_revenue, palette='husl')
plt.title('Average Revenue by Customer Demographics')
plt.ylabel('Average Revenue')
plt.xlabel('Customer Demographics')
plt.show()
```

```
C:\Users\STORM\AppData\Local\Temp\ipykernel_9268\3822175976.py:6: 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` for the same effect.

  sns.barplot(x='customer_demographics', y='revenue_generated', data=customer_revenue, palette='husl')
```



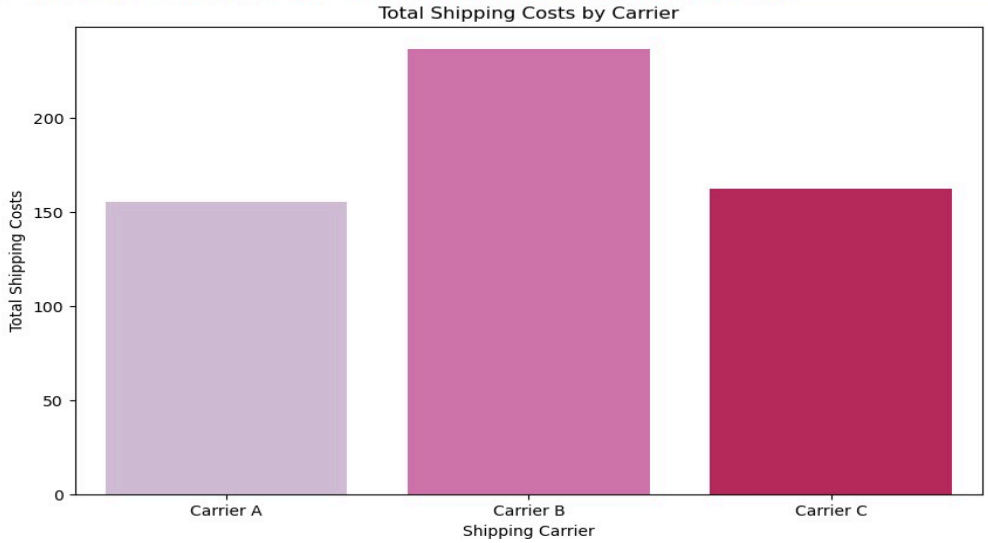
```
route_costs = data.groupby('routes')['transportation_costs'].mean().reset_index()
# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='routes', y='transportation_costs', data=route_costs, palette='YlOrBr')
plt.title('Average Transportation Cost by Routes')
plt.ylabel('Average Transportation Cost')
plt.xlabel('Shipping Routes')
plt.show()
```

```
C:\Users\STORM\AppData\Local\Temp\ipykernel_9268\2196699143.py:6: 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` for the same effect.

  sns.barplot(x='routes', y='transportation_costs', data=route_costs, palette='YlOrBr')
```



[177]:
```
# Group by shipping carrier and sum the shipping costs
shipping_costs = data.groupby('shipping_carriers')['shipping_costs'].sum().reset_index()
# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='shipping_carriers', y='shipping_costs', data=shipping_costs, palette='PuRd')
plt.title('Total Shipping Costs by Carrier')
plt.ylabel('Total Shipping Costs')
plt.xlabel('Shipping Carrier')
plt.show()
```

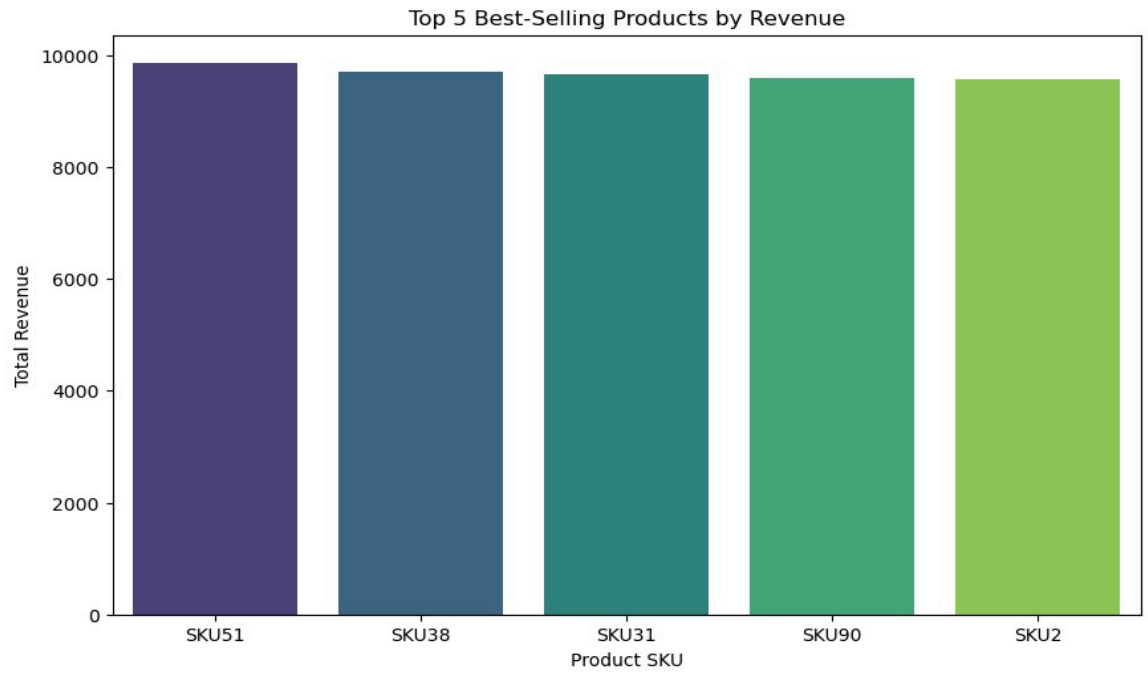
```
C:\Users\STORM\AppData\Local\Temp\ipykernel_9268\250031278.py:6: FutureWarning:

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` for the same effect.

  sns.barplot(x='shipping_carriers', y='shipping_costs', data=shipping_costs, palette='PuRd')
```



Visualization

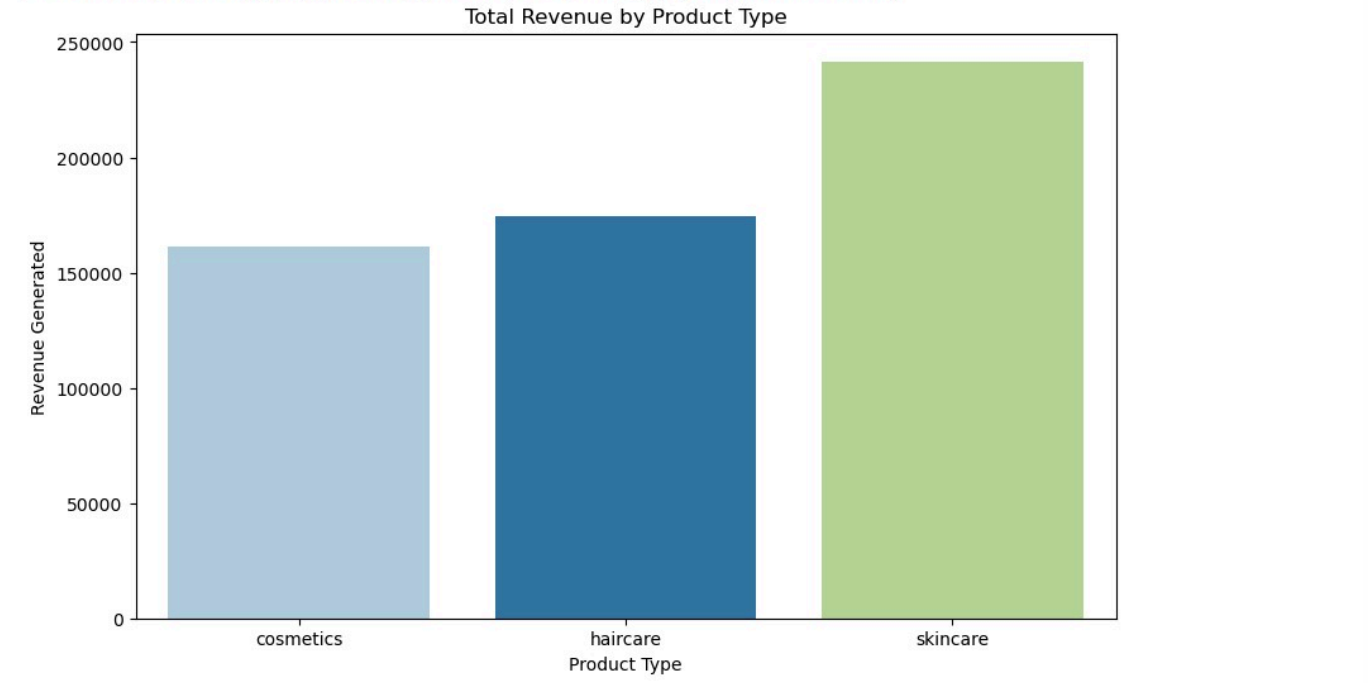
```
import matplotlib.pyplot as plt
import seaborn as sns
best_selling = data.groupby('sku')['revenue_generated'].sum().reset_index().sort_values(by='revenue_generated', ascending=False).head(5)

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='sku', y='revenue_generated', data=best_selling, palette='viridis')
plt.title('Top 5 Best-Selling Products by Revenue')
plt.ylabel('Total Revenue')
plt.xlabel('Product SKU')
plt.show()
```

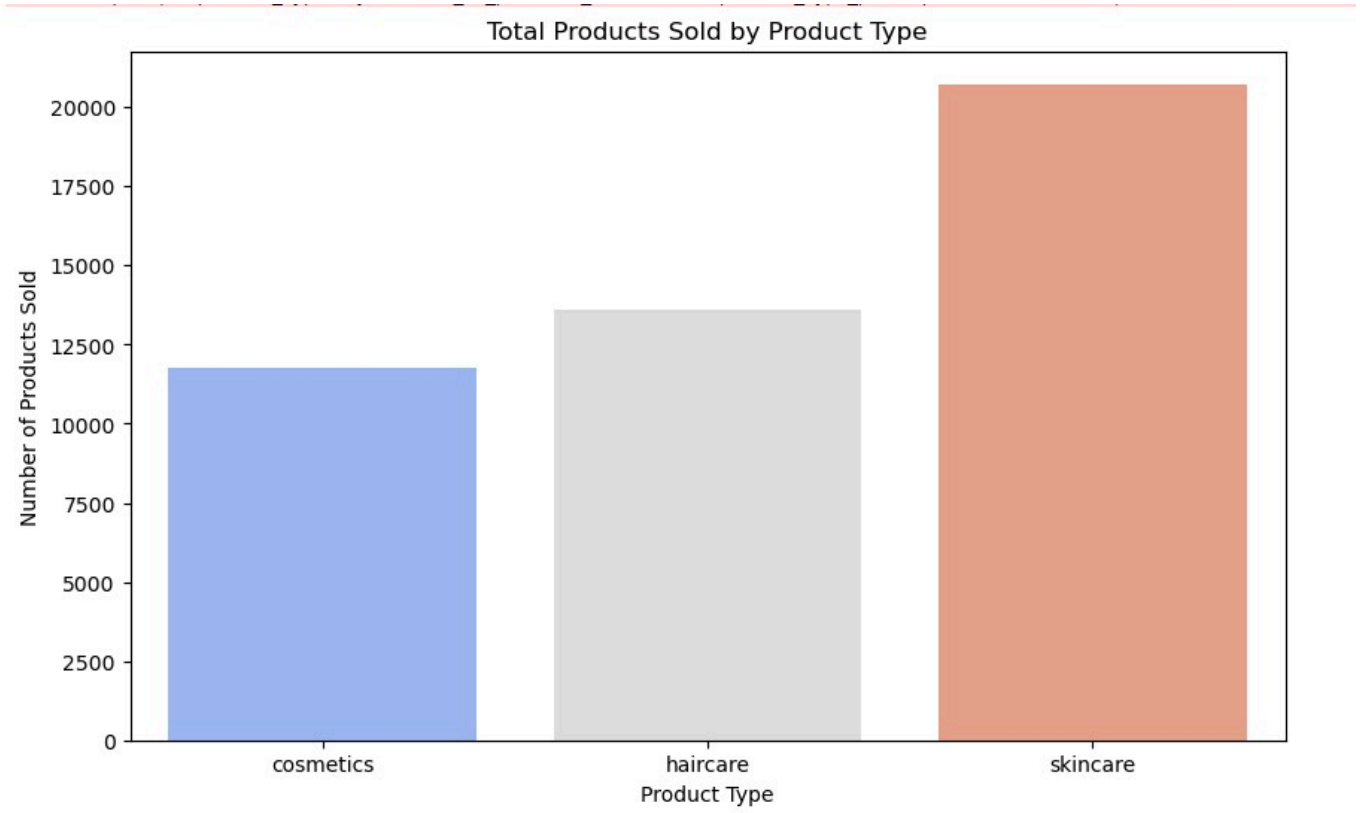
```
C:\Users\STORM\AppData\Local\Temp\ipykernel_9268\3668861183.py:7: FutureWarning:

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` for the
same effect.

  sns.barplot(x='sku', y='revenue_generated', data=best_selling, palette='viridis')
```



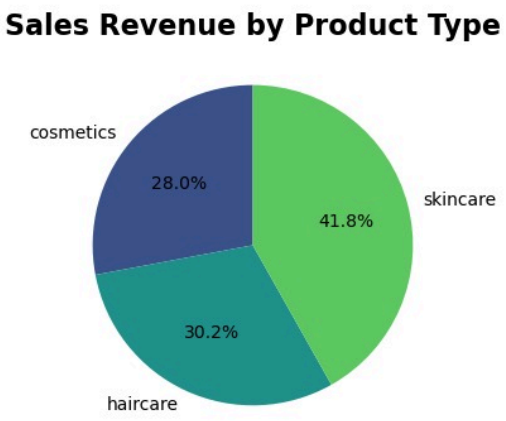
```python
# Group by product type and calculate total sales and revenue
product_type_perf = data.groupby('product_type').agg({'number_of_products_sold': 'sum',
                                                      'revenue_generated': 'sum'}).reset_index()

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='product_type', y='revenue_generated', data=product_type_perf, palette='Paired')
plt.title('Total Revenue by Product Type')
plt.ylabel('Revenue Generated')
plt.xlabel('Product Type')
plt.show()

# For sales volume:
plt.figure(figsize=(10, 6))
sns.barplot(x='product_type', y='number_of_products_sold', data=product_type_perf, palette='coolwarm')
plt.title('Total Products Sold by Product Type')
plt.ylabel('Number of Products Sold')
plt.xlabel('Product Type')
plt.show()
```

```
C:\Users\STORM\AppData\Local\Temp\ipykernel_9268\2840786767.py:7: 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` for the same effect.

  sns.barplot(x='product_type', y='revenue_generated', data=product_type_perf, palette='Paired')
```

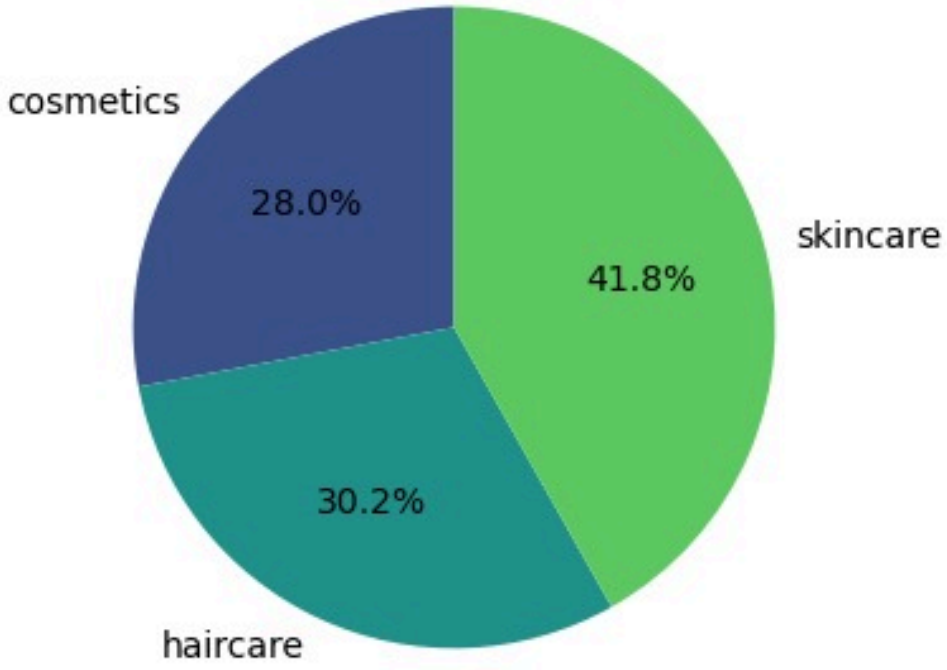




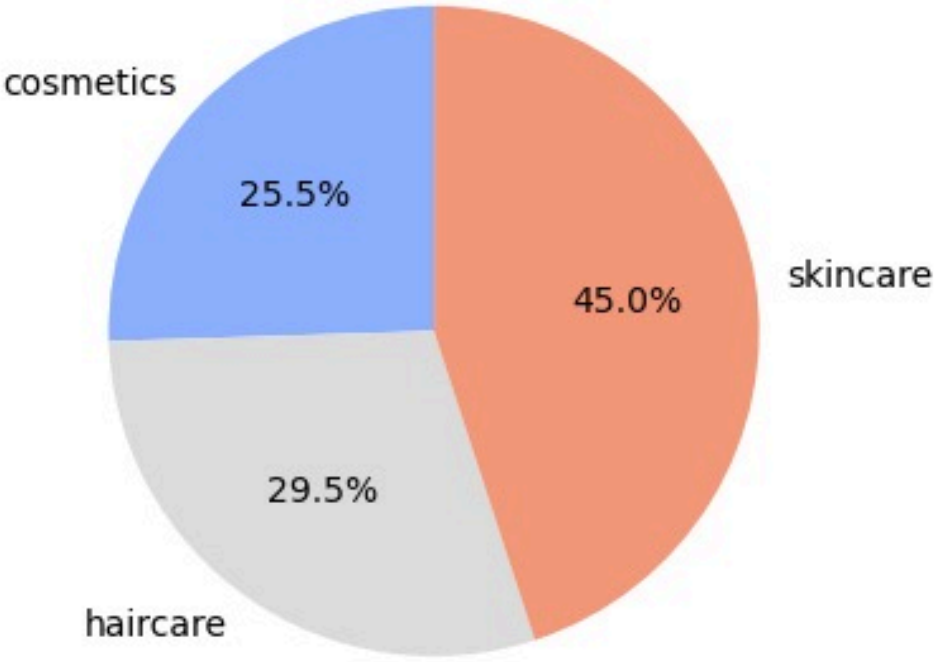
0]:
```
# Group by product type and calculate total sales and revenue
product_type_perf = data.groupby('product_type').agg({'number_of_products_sold': 'sum',
                                                      'revenue_generated': 'sum'}).reset_index()

# Pie chart plot
plt.figure(figsize=(4, 4))
plt.pie(revenue_by_product_type['revenue_generated'], labels=revenue_by_product_type['product_type'],
        autopct='%1.1f%%', startangle=90, colors=sns.color_palette('viridis', len(revenue_by_product_type)))

# Equal aspect ratio ensures the pie chart is drawn as a circle
plt.title('Sales Revenue by Product Type', fontsize=16, fontweight='bold')
plt.tight_layout()
plt.show()

# Plot a pie chart
plt.figure(figsize=(4, 4))
plt.pie(products_sold_by_type['number_of_products_sold'], labels=products_sold_by_type['product_type'],
        autopct='%1.1f%%', startangle=90, colors=sns.color_palette('coolwarm', len(products_sold_by_type)))

# Ensure pie is drawn as a circle
plt.title('Total Products Sold by Product Type', fontsize=16, fontweight='bold')
plt.tight_layout()
plt.show()
```



Sales Revenue by Product Type

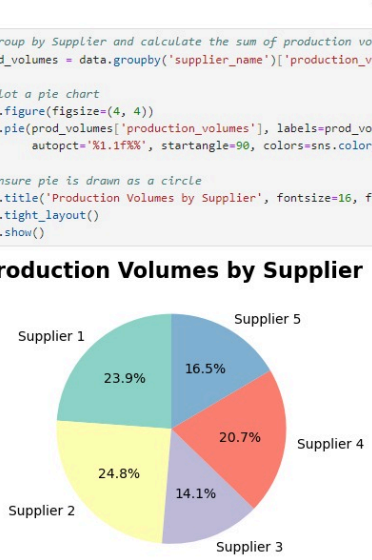


Total Products Sold by Product Type

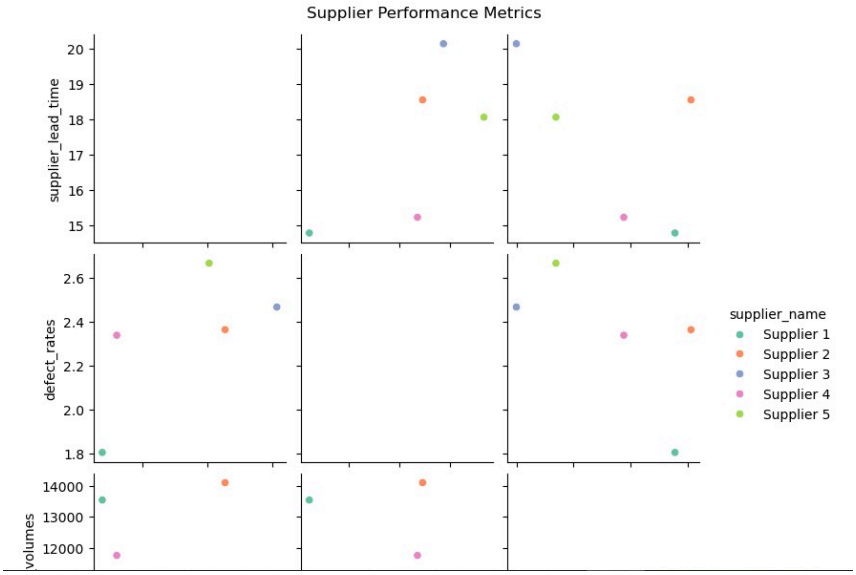


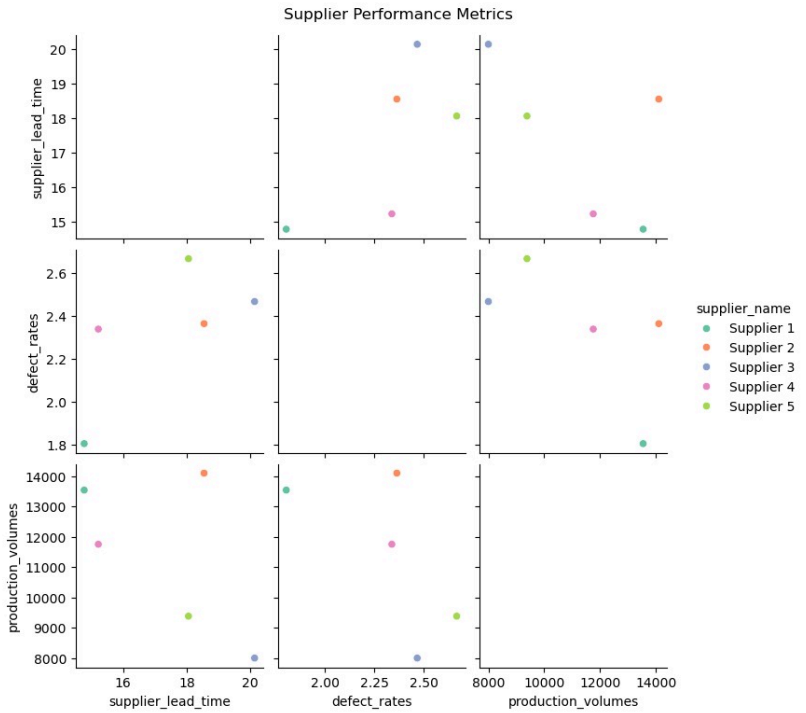
Supplier

```
# Group by Supplier and calculate the sum of production volumes
prod_volumes = data.groupby('supplier_name')['production_volumes'].sum().reset_index()

# Plot a pie chart
plt.figure(figsize=(4, 4))
plt.pie(prod_volumes['production_volumes'], labels=prod_volumes['supplier_name'],
        autopct='%1.1f%%', startangle=90, colors=sns.color_palette('Set3', len(prod_volumes)))

# Ensure pie is drawn as a circle
plt.title('Production Volumes by Supplier', fontsize=16, fontweight='bold')
plt.tight_layout()
plt.show()
```




```
# Group by supplier and calculate performance metrics
supplier_performance = data.groupby('supplier_name').agg({'supplier_lead_time': 'mean',
                                                          'defect_rates': 'mean',
                                                          'production_volumes': 'sum'}).reset_index()

# Plot a pairplot to visualize relationships
sns.pairplot(supplier_performance, hue='supplier_name', palette='Set2')
plt.suptitle('Supplier Performance Metrics', y=1.02)
plt.show()
```




Supplier Performance Metrics
supplier_lead_time
defect_rates
production_volumes
supplier_name
Supplier 1
Supplier 2
Supplier 3
Supplier 4
Supplier 5

Interactive Dashboard Visualization