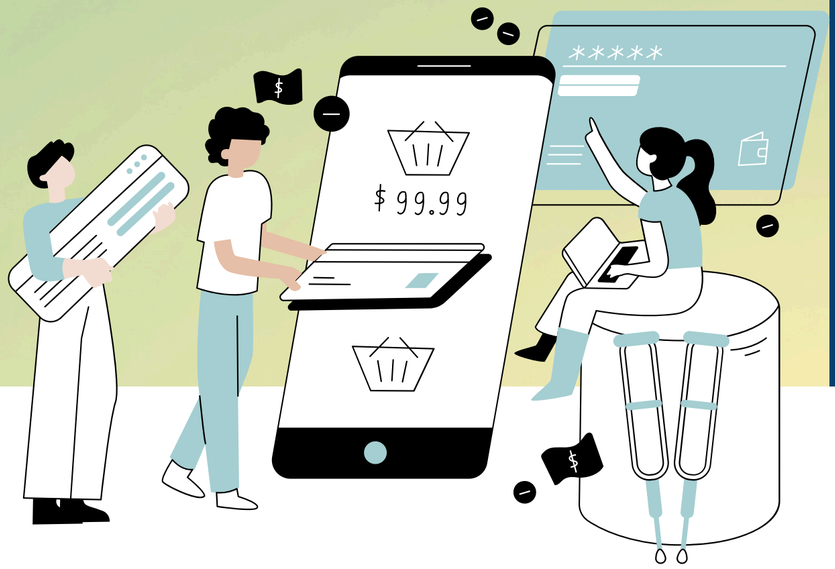
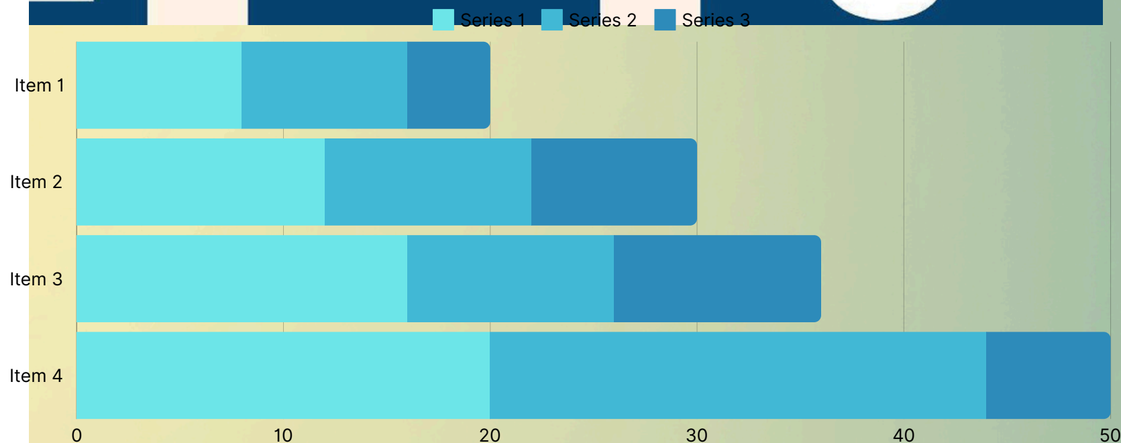


E-Commerce Data Analysis

by Amit Prasad



Amit Prasad

Data Analyst

INTRODUCTION

Undergraduate Student at Asansol Engineering College: I am currently pursuing my B.Tech degree at Asansol Engineering College, focusing on [mention your field or specialization]. My academic journey includes a strong foundation in [mention relevant subjects or interests], and I am passionate about applying my skills to real-world projects and challenges.

SPECIALIZATIONS

As a student specializing in data analysis, I am dedicated to exploring and interpreting data to extract meaningful insights. My coursework and projects have equipped me with skills in data manipulation, visualization, and the use of tools like Python, SQL, and statistical methods to uncover patterns and trends. I am enthusiastic about leveraging data to drive informed decisions and solve complex problems in various domains.

Contents

REAL ESTATE LISTING — JANUARY 2020

PROJECT OVERVIEW

THE PROJECT AIMED TO ANALYZE A SUBSTANTIAL DATASET USING PYTHON IN CONJUNCTION WITH MYSQL. THE DATASET, COMPRISING [DESCRIBE THE DATASET BRIEFLY], REQUIRED ROBUST DATA HANDLING CAPABILITIES AND EFFICIENT QUERY MANAGEMENT FOR DERIVING ACTIONABLE INSIGHTS.

Integration of Python and SQL

Data Import Process

Challenges with MySQL Direct Import: Directly importing large datasets into MySQL posed significant challenges:

- Performance Issues: Sluggish import speeds due to the dataset's size and complexity.
- Time-Consuming Process: Lengthy import times impacting project timelines and efficiency.

Using mysql.connector in Python:

To address these challenges, mysql.connector in Python was instrumental:

- Efficient CSV Import: Leveraged mysql.connector to efficiently batch-import CSV files into MySQL, optimizing data insertion processes.
- Preprocessing Capabilities: Python facilitated data preprocessing tasks before inserting data into MySQL, ensuring data quality and consistency.

REAL ESTATE LISTING — JANUARY 2020

PROJECT OBJECTIVE

Efficient Data Import

Implement a streamlined process to import large CSV files into MySQL, overcoming traditional performance bottlenecks.

Optimized Data Manipulation

Utilize Python's data manipulation libraries and MySQL's querying capabilities to preprocess and analyze data effectively.

Insightful Visualization

Develop insightful visualizations to uncover trends, patterns, and anomalies within the dataset.

PROBLEM STATEMENTS

Basic Queries

1. List all unique cities where customers are located.
2. Count the number of orders placed in 2017.
3. Find the total sales per category.
4. Calculate the percentage of orders that were paid in installments.
5. Count the number of customers from each state.

Intermediate Queries

1. Calculate the number of orders per month in 2018.
2. Find the average number of products per order, grouped by customer city.
3. Calculate the percentage of total revenue contributed by each product category.
4. Identify the correlation between product price and the number of times a product has been purchased.
5. Calculate the total revenue generated by each seller, and rank them by revenue.

1. LIST ALL UNIQUE CITIES WHERE CUSTOMERS ARE LOCATED.

```
query = """ select distinct upper(customer_city) from customers """

cur.execute(query)

data = cur.fetchall()

df= pd.DataFrame(data)

df.head()
```

	0
0	FRANCA
1	SAO BERNARDO DO CAMPO
2	SAO PAULO
3	MOGI DAS CRUZES
4	CAMPINAS

2. COUNT THE NUMBER OF ORDERS PLACED IN 2017.

```
query = """select count(order_id) from orders where year(order_purchase_timestamp) = 2017 """

cur.execute(query)

data = cur.fetchall()

"total order placed in 2017 are ",data[0][0]
```

```
('total order placed in 2017 are ', 45101)
```

3. FIND THE TOTAL SALES PER CATEGORY.

```
query = """  select upper(products.product_category),
round(sum(payments.payment_value),2) sales
from products join order_items
on products.product_id = order_items.product_id
join payments
on payments.order_id = order_items.order_id
group by product_category
"""  cur.execute(query)
data = cur.fetchall()
df= pd.DataFrame(data,columns = ["Category","Sales"])
df
```

	Category	Sales
0	PERFUMERY	506738.66
1	FURNITURE DECORATION	1430176.39
2	TELEPHONY	486882.05
3	BED TABLE BATH	1712553.67
4	AUTOMOTIVE	852294.33
...	...	...
69	CDS MUSIC DVDS	1199.43
70	LA CUISINE	2913.53
71	FASHION CHILDREN'S CLOTHING	785.67
72	PC GAMER	2174.43
73	INSURANCE AND SERVICES	324.51

74 rows × 2 columns

4. CALCULATE THE PERCENTAGE OF ORDERS THAT WERE PAID IN INSTALLMENTS.

```
query = """ select sum(case when
payment_installments >= 1 then 1

else 0 end)/count(*)*100 from payments

"""

cur.execute(query)

data = cur.fetchall()

" the pecentage of orders that were paid in
installments is ",data[0][0]
```

```
(' the pecentage of orders that were paid in installments is ',
 Decimal('99.9981'))
```

5. COUNT THE NUMBER OF CUSTOMERS FROM EACH STATE.

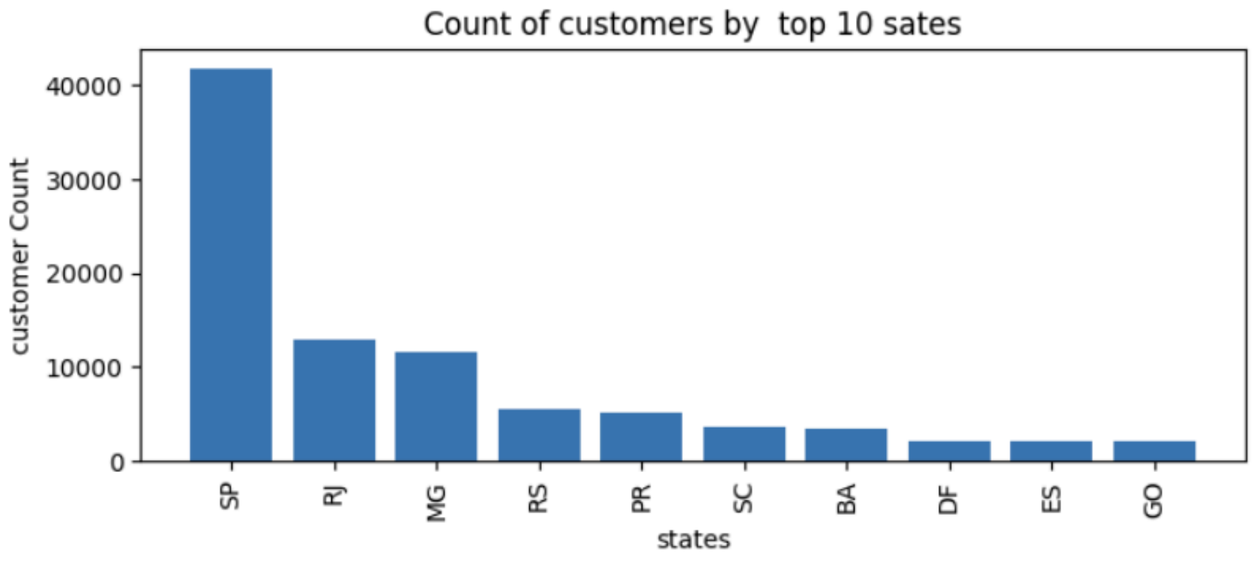
```
query = """ select customer_state , count(customer_id)
from customers group by customer_state
"""

cur.execute(query)

data = cur.fetchall()

df= pd.DataFrame(data,columns = ["state","customer_count"])
df = df.sort_values(by = "customer_count", ascending = False)
df = df.head(10)
plt.figure(figsize = (8,3) )
plt.bar(df["state"],df["customer_count"])
plt.xlabel('states')
plt.ylabel('customer Count')
plt.title('Count of customers by  top 10 sates')

plt.xticks(rotation =90)
plt.show()
```



CALCULATE THE NUMBER OF ORDERS PER MONTH IN 2018.

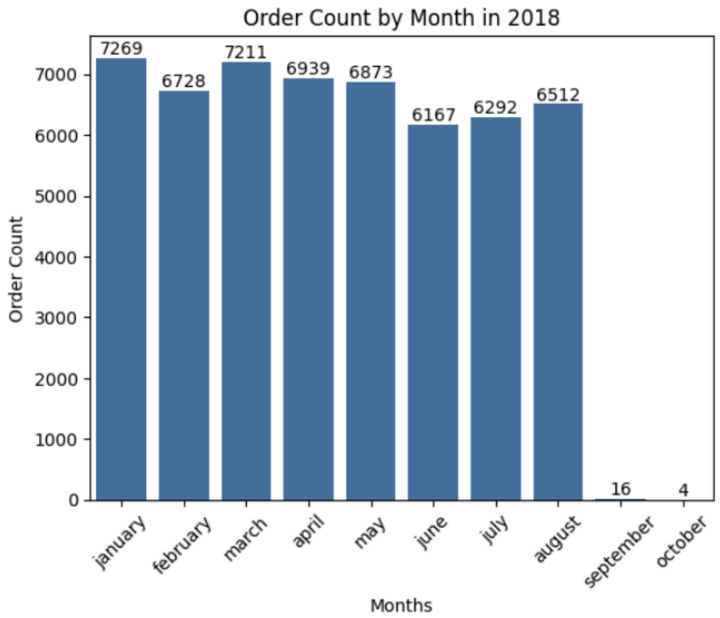
```
query = """
SELECT MONTHNAME(order_purchase_timestamp) AS months, COUNT(order_id) AS order_count
FROM orders
WHERE YEAR(order_purchase_timestamp) = 2018
GROUP BY months
"""

cur.execute(query)
data = cur.fetchall()

df = pd.DataFrame(data, columns=["months", "order_count"])
df["months"] = df["months"].str.lower()
order = ["january", "february", "march", "april", "may", "june", "july", "august", "september", "october"]



ax = sns.barplot(x="months", y="order_count", data=df, order=order)
plt.xticks(rotation=45)
plt.xlabel('Months')
plt.ylabel('Order Count')
plt.title('Order Count by Month in 2018')
ax.bar_label(ax.containers[0])
plt.show()
```



2. FIND THE AVERAGE NUMBER OF PRODUCTS PER ORDER, GROUPED BY CUSTOMER CITY.

```
query = """
with count_per_order as
(select orders.order_id,orders.customer_id, count(order_items.order_id) as oc
from orders join order_items
on orders.order_id = order_items.order_id
group by orders.order_id, orders.customer_id)

select customers.customer_city,round(avg(count_per_order.oc),2) average_order
from customers join count_per_order
on customers.customer_id = count_per_order.customer_id
group by customers.customer_city order by average_order desc
"""

cur.execute(query)
data = cur.fetchall()
df=pd.DataFrame(data, columns = ["customer city", "average orders per order"])
df.head(10)
```

customer city	average orders per order
padre carvalho	7.00
celso ramos	6.50
datas	6.00
candido godoi	6.00
matias olimpio	5.00
cidelandia	4.00
picarra	4.00
morro de sao paulo	4.00
teixeira soares	4.00
curralinho	4.00

3. CALCULATE THE PERCENTAGE OF TOTAL REVENUE CONTRIBUTED BY EACH PRODUCT CATEGORY.

```
query = """  select upper(products.product_category),
round((sum(payments.payment_value) / (select sum(payment_value) from
payments))*100,2) sales_percentage
from products join order_items
on products.product_id = order_items.product_id
join payments
on payments.order_id = order_items.order_id
group by product_category order by sales_percentage desc;



"""


cur.execute(query)


data = cur.fetchall()


df= pd.DataFrame(data,columns = ["Category","Sales"])
df
```

	Category	Sales
0	BED TABLE BATH	10.70
1	HEALTH BEAUTY	10.35
2	COMPUTER ACCESSORIES	9.90
3	FURNITURE DECORATION	8.93
4	WATCHES PRESENT	8.93
...	...	...
69	HOUSE COMFORT 2	0.01
70	CDS MUSIC DVDS	0.01
71	PC GAMER	0.01
72	FASHION CHILDREN'S CLOTHING	0.00
73	INSURANCE AND SERVICES	0.00

IDENTIFY THE CORRELATION BETWEEN PRODUCT PRICE AND THE NUMBER OF TIMES A PRODUCT HAS BEEN PURCHASED.

```
query = """  select upper(products.product_category),
count(order_items.product_id),
round(avg(order_items.price),2)
from products join order_items
on products.product_id = order_items.product_id
group  by products.product_category;

"""

cur.execute(query)

data = cur.fetchall()

df= pd.DataFrame(data,columns = ["Category","order_count","price"])
df
```

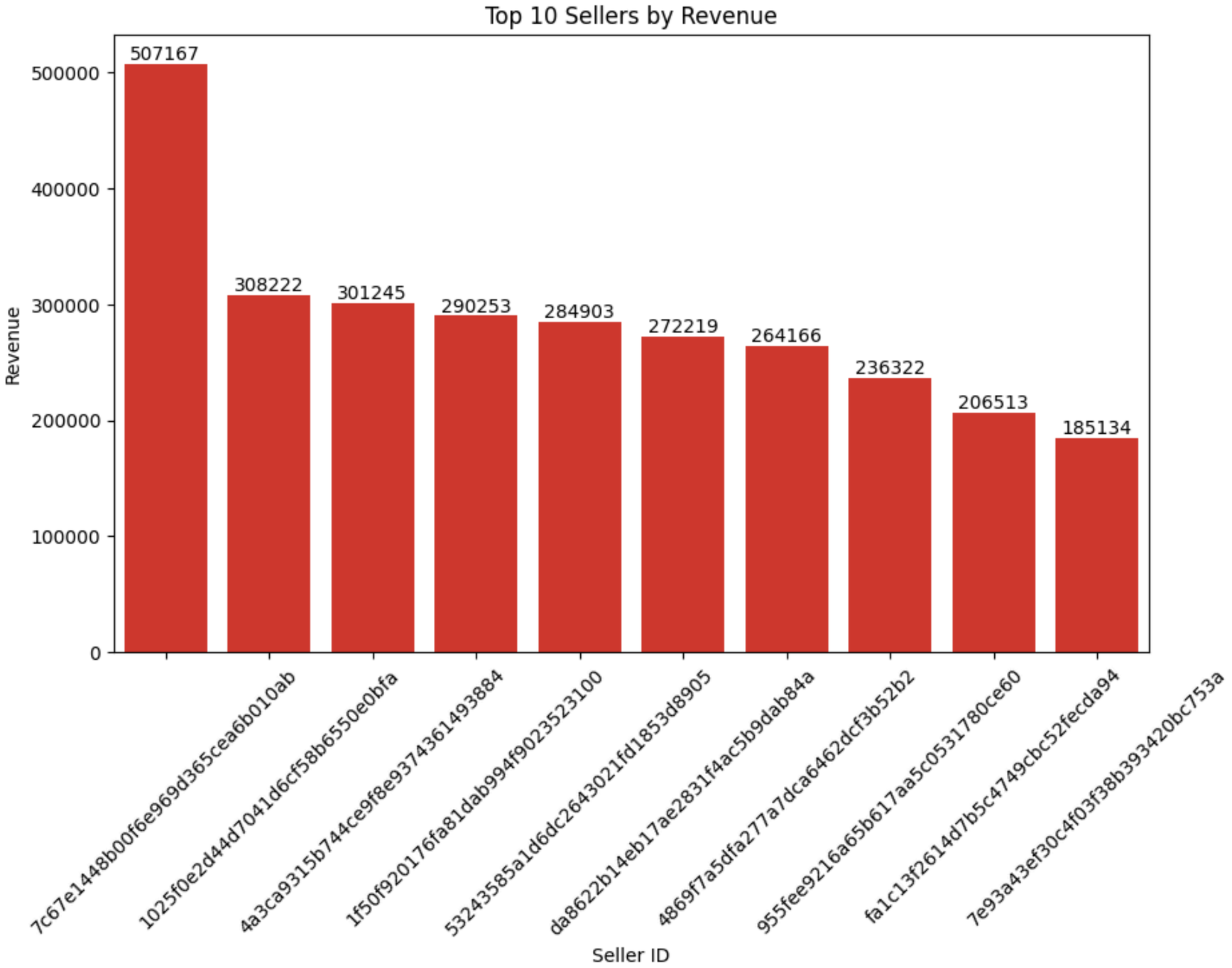
	Category	order_count	price
0	HEALTH BEAUTY	9670	130.16
1	SPORT LEISURE	8641	114.34
2	COOL STUFF	3796	167.36
3	COMPUTER ACCESSORIES	7827	116.51
4	WATCHES PRESENT	5991	201.14
...	...	...	...
69	FLOWERS	33	33.64
70	KITCHEN PORTABLE AND FOOD COACH	15	264.57
71	HOUSE COMFORT 2	30	25.34
72	CITTE AND UPHACK FURNITURE	38	114.95
73	CDS MUSIC DVDS	14	52.14

74 rows × 3 columns

CALCULATE THE TOTAL REVENUE GENERATED BY EACH SELLER, AND RANK THEM BY REVENUE.

```
query = """
SELECT *, dense_rank() OVER (ORDER BY revenue DESC) AS rn
FROM (
    SELECT order_items.seller_id, SUM(payments.payment_value) AS revenue
    FROM order_items
    JOIN payments ON order_items.order_id = payments.order_id
    GROUP BY order_items.seller_id
) AS a
"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns=["seller_id", "revenue", "rank"])
df = df.head(10)
plt.figure(figsize=(10, 6))
ax = sns.barplot(x="seller_id", y="revenue", data=df, color='red')

plt.xlabel('Seller ID')
plt.ylabel('Revenue')
plt.title('Top 10 Sellers by Revenue')
plt.xticks(rotation=45)
ax.bar_label(ax.containers[0])
plt.show()
```


Top 10 Sellers by Revenue
Revenue
Seller ID
0
100000
200000
300000
400000
500000
507167
308222
301245
290253
284903
272219
264166
236322
206513
185134
7c67e1448b00f6e969d365cea6b010ab
1025f0e2d44d7041d6cf58b6550e0bfa
4a3ca9315b744ce9f8e937436149388 4
1f50f920176fa81dab994f9023523100
53243585a1d6dc2643021fd1853d8905
da8622b14eb17ae2831f4ac5b9dab84a
4869f7a5dfa277a7dca6462dcf3b52b2
955fee9216a65b617aa5c0531780ce60
fa1c13f2614d7b5c4749cbc52fecda94
7e93a43ef30c4f03f38b393420bc753a

CONCLUSION

IN THIS ECOMMERCE DATA ANALYSIS PROJECT USING PYTHON AND SQL, I THOROUGHLY EXPLORED CUSTOMER DEMOGRAPHICS, SALES TRENDS, AND PRODUCT PERFORMANCE. BY ANSWERING A RANGE OF QUERIES FROM BASIC CUSTOMER INSIGHTS TO COMPLEX REVENUE ANALYSIS AND CORRELATION STUDIES, I UNCOVERED VALUABLE INSIGHTS INTO REGIONAL CUSTOMER BEHAVIOR, CATEGORY PERFORMANCE, AND SELLER RANKINGS. THIS PROJECT NOT ONLY ENHANCED MY TECHNICAL SKILLS BUT ALSO PROVIDED ACTIONABLE INSIGHTS CRUCIAL FOR OPTIMIZING MARKETING STRATEGIES, INVENTORY MANAGEMENT, AND OVERALL BUSINESS PERFORMANCE IN THE COMPETITIVE ECOMMERCE LANDSCAPE.

THANK YOU