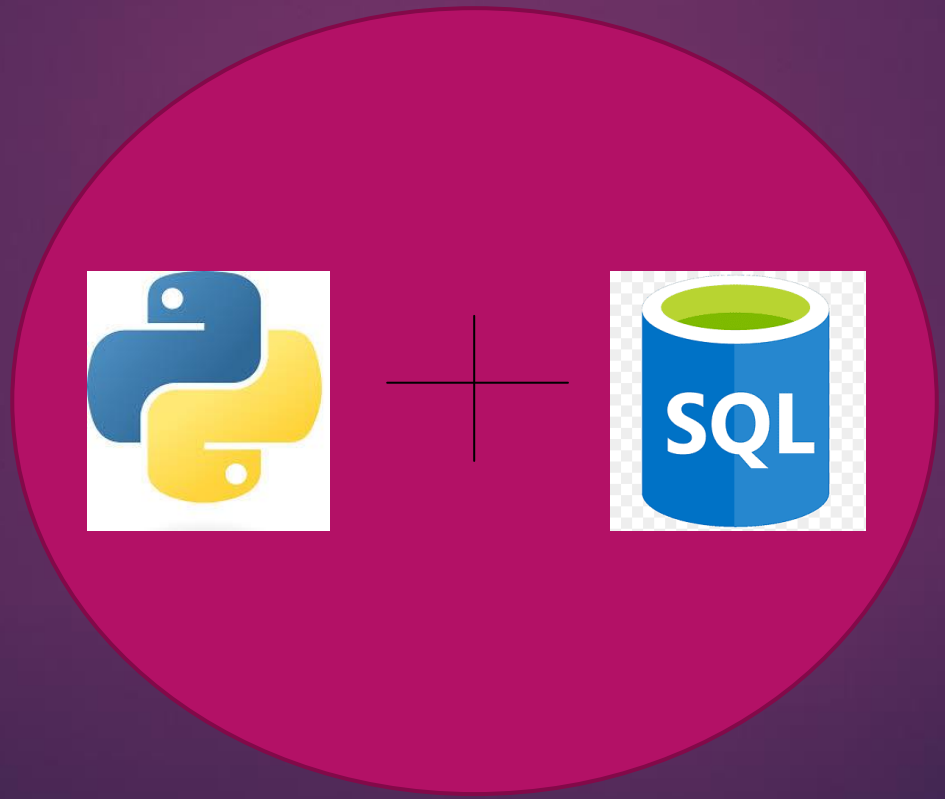
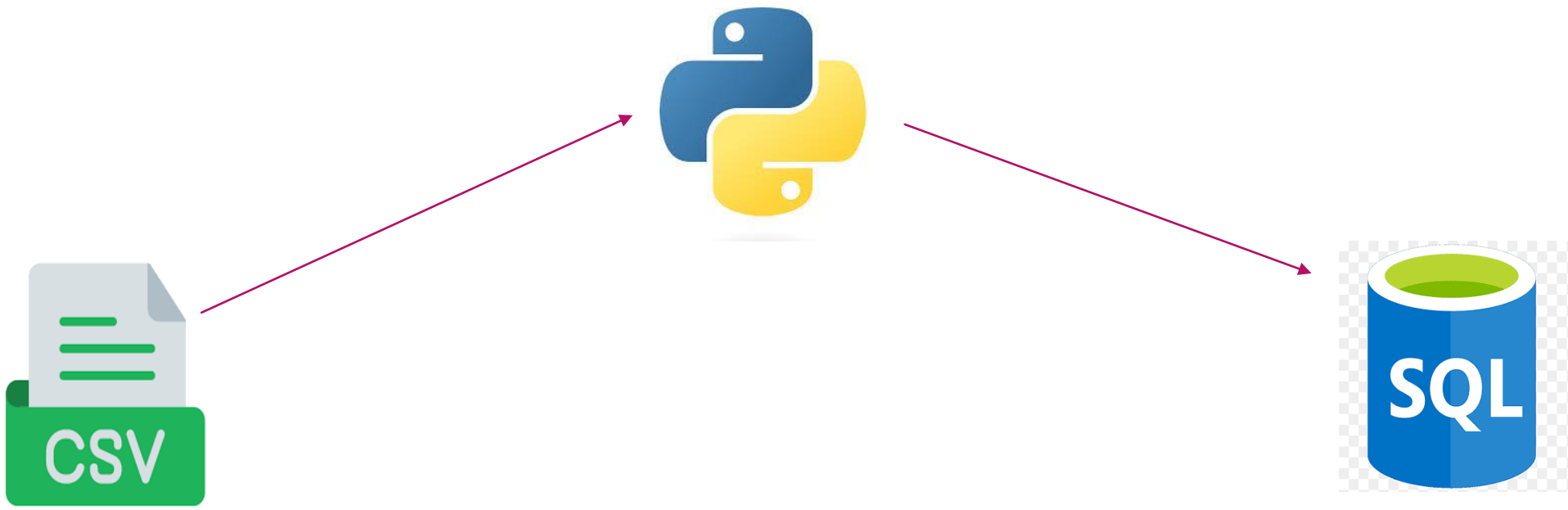
Ecommerce Project



Connectivity:

Retrieve data from SQL using python to create charts out of it.



Overview:

- **Target** is a globally recognized brand and a leading retailer in the United States, known for offering exceptional value, inspiration, innovation, and a unique shopping experience.

- This dataset focuses on Target's operations in Brazil, covering 100,000 orders placed between 2016 and 2018. It includes detailed information on order status, pricing, payment and shipping performance, customer locations, product attributes, and customer reviews.

Use Case:

- Analyzing this dataset offers valuable insights into Target's Brazilian operations, revealing details about order processing, pricing strategies, payment and shipping efficiency, customer demographics, product characteristics, and customer satisfaction. This comprehensive dataset is a valuable resource for understanding various business aspects and enhancing strategic decision-making.

1. List all unique cities where customers are located.

```
query = """ select distinct customer_city from customers """
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data)
df.head()
```

Output

0	franca
1	sao bernardo do campo
2	sao paulo
3	mogi das cruzes
4	campinas

2. Find the total sales per category.

```
query = """ select a1.product_category, round(sum(a3.payment_value),2) as total from products a1 join order_items a2
on a1.product_id = a2.product_id
join payments a3 on a2.order_id = a3.order_id
group by product_category """
cur.execute(query)
data = cur.fetchall()
data
df = pd.DataFrame(data, columns = ["category","sales"])
df
```

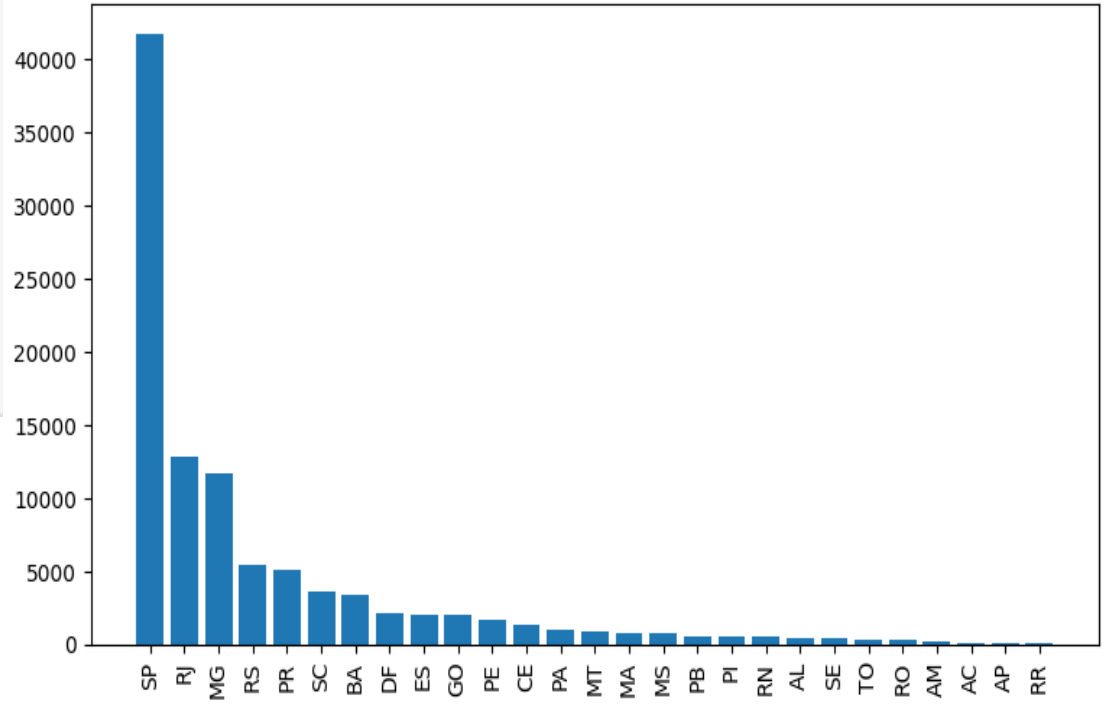
Output

	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

3. 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","count"])
df = df.sort_values(by="count", ascending=False)
plt.figure(figsize=(9,5))
plt.bar(df["state"],df["count"])
plt.xticks(rotation = 90)
plt.show()
```

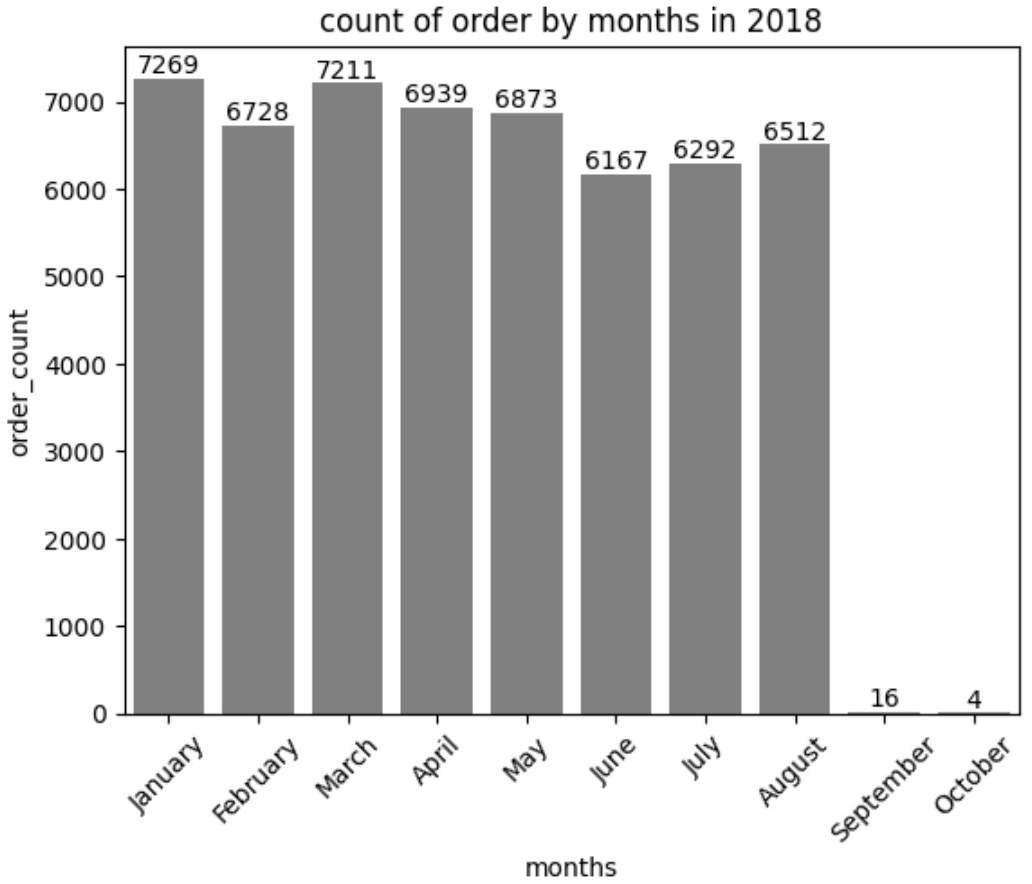
Output



4. Calculate the number of orders per month in 2018

```
query = """ select monthname(order_purchase_timestamp) as months, count(order_id) as 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"])
o = ["January","February","March","April","May","June","July","August","September","October"]
ax = sns.barplot(x=df["months"],y=df["order_count"],data=df,order=o,color = "grey")
plt.xticks(rotation=45)
ax.bar_label(ax.containers[0])
plt.title("count of order by months in 2018")
plt.show()
```

Output



5. Calculate the cumulative sales per month for each year

```
query = """ select years,months,sales, sum(sales) over(order by years,months) as cumulative_sales from
(select year(o.order_purchase_timestamp) as years,
month(o.order_purchase_timestamp) as months,
round(sum(p.payment_value),2) as sales from orders o join payments p
on o.order_id = p.order_id
group by years,months
order by years,months) as a """
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns=["year","month","sales","cumulative_sales"])
df
```

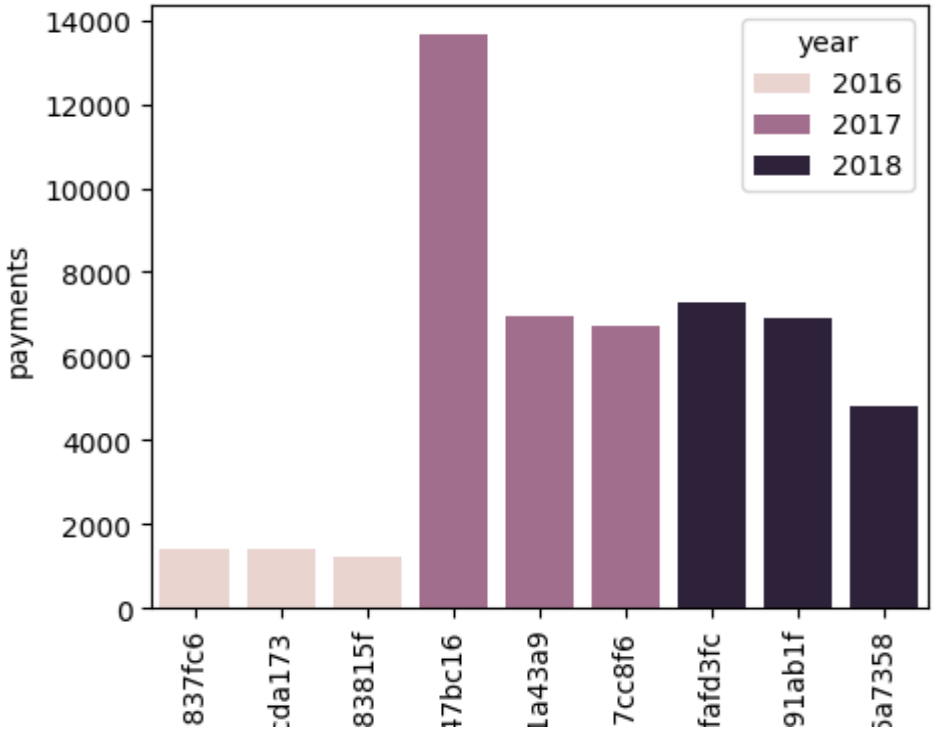
Output

	year	month	sales	cumulative_sales
0	2016	9	252.24	252.24
1	2016	10	59090.48	59342.72
2	2016	12	19.62	59362.34
3	2017	1	138488.04	197850.38
4	2017	2	291908.01	489758.39
5	2017	3	449863.60	939621.99
6	2017	4	417788.03	1357410.02
7	2017	5	592918.82	1950328.84
8	2017	6	511276.38	2461605.22
9	2017	7	592382.92	3053988.14
10	2017	8	674396.32	3728384.46

6. Identify the top 3 customers who spent the most money in each year

```
query = """ select years, customer_id, payments, d_rank from
(select year(o.order_purchase_timestamp) as years, o.customer_id, round(sum(p.payment_value),2) as payments,
dense_rank() over(partition by year(o.order_purchase_timestamp) order by sum(p.payment_value) desc) as d_rank from orders o join payments p
on o.order_id = p.order_id
group by years, customer_id) as a
where d_rank <=3 """
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns=["year","customer_id","payments","ranks"])
plt.figure(figsize=(8,8))
sns.barplot(x="customer_id",y="payments",data=df,hue="year")
plt.xticks(rotation=90)
plt.savefig("top 3 customers.jpg")
plt.show()
```

Output



Thank You

Please share your valuable feedback