

DAY 2 LAB EXPERIMENTS

NAME : LARAVIND

Reg No : 192424080

Date : 17/12/2025

EXP_6 To Calculate the total Cost of Customer Purchase including Discount and Taxes

The screenshot shows a Jupyter Notebook interface with a dark theme. On the left, there's a sidebar with a 'Files' section containing a 'customer.csv' file. The main area has three code cells:

```
import pandas as pd
import numpy as np
```

```
df=pd.read_csv("customer.csv")
print(df.head())
```

	Order ID	Customer Name	Category	Sub Category	City	\
0	001	Harish	Oil & Masala	Masalas	Vellore	
1	002	Sudha	Beverages	Health Drinks	Krishnagiri	
2	003	Hussain	Food Grains	Atta & Flour	Percambalur	
3	004	Jackson	Fruits & Vegetables	Fresh Vegetables	Dharmapuri	
4	005	Ridhesh	Food Grains	Organic Staples	Ooty	

```
Order Date Region Sales Discount Profit State
0 11-08-2017 North 1254 0.12 401.28 Tamil Nadu
1 11-08-2017 South 749 0.18 149.88 Tamil Nadu
2 06-12-2017 West 2368 0.21 165.20 Tamil Nadu
3 10-11-2016 South 896 0.25 89.68 Tamil Nadu
4 10-11-2016 South 2355 0.26 918.45 Tamil Nadu
```

```
discount_rate=10
tax_rate=5
total_cost=(df["Sales"]).sum()
discount_amount=(discount_rate/100)*total_cost
tax_amount=(tax_rate/100)*(total_cost-discount_amount)
total_amount=total_cost-discount_amount+tax_amount
```

```
print("Total Cost is:",total_cost)
print("Discount amount is:",discount_amount)
print("Tax amount is:",tax_amount)
print("Total amount is:",total_amount)
```

EXP_7 Data Analyst for e commerce company and perform some operations like calculate total number of orders, average

```
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Files
[87] ✓ On EXP_7 Data Analyst for e commerce company and perform some operations like calculate total number of orders,average
import pandas as pd
import numpy as np

df=pd.read_csv("e_commerce.csv",encoding="latin1")
print(df.head())
... InvoiceNo StockCode Description Quantity \
0 536365 85123A WHITE HANGING HEART T-LIGHT HOLDER 6
1 536365 71053 WHITE METAL LANTERN 6
2 536365 844068 CREAM CUPID HEARTS COAT HANGER 8
3 536365 848296 KNITTED UNION FLAG HOT WATER BOTTLE 6
4 536365 848296 RED WOOLLY HOTTE WHIE HEART. 6

InvoiceDate UnitPrice CustomerID Country
0 12/1/2010 8:26 2.55 17850.0 United Kingdom
1 12/1/2010 8:26 3.39 17850.0 United Kingdom
2 12/1/2010 8:26 2.75 17850.0 United Kingdom
3 12/1/2010 8:26 3.39 17850.0 United Kingdom
4 12/1/2010 8:26 3.39 17850.0 United Kingdom

df["InvoiceDate"] = pd.to_datetime(df["InvoiceDate"])

[88] ✓ On
total_orders=df.groupby("CustomerID").size()
print("Total number of orders by each customer is :")
print(total_orders)

Total number of orders by each customer is :
CustomerID
12346.0      2
12347.0    182
Disk 86.55 GB available
```

```
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Files
[89] ✓ On Total number of orders by each customer is :
CustomerID
12346.0      2
12347.0    182
12348.0      31
12349.0      73
12350.0      17
...
18280.0      10
18281.0       7
18282.0      13
18283.0     756
18284.0      78
Length: 4372, dtype: int64

[90] ✓ On
average_quantity=df.groupby("StockCode")["Quantity"].mean()
print("Average order quantity for each product is :")
print(average_quantity)

... Average order quantity for each product is :
StockCode
10002        14.205479
10080        20.625000
10120         6.433333
10123C       3.250000
10123G       38.000000

gift_0001_20      2.000000
gift_0001_30      4.625000
gift_0001_40      1.000000
gift_0001_50      1.000000
...
Name: Quantity, Length: 4070, dtype: float64

[91] ✓ On
earliest_date = df["InvoiceDate"].min()
```

```

File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all

Files
sample_data
customer.csv

[1]
1 11-09-2017 South 749 0.18 149.80 Tamil Nadu
2 06-12-2017 West 2369 0.21 165.20 Tamil Nadu
3 10-11-2016 South 896 0.25 89.60 Tamil Nadu
4 10-11-2016 South 2355 0.26 918.45 Tamil Nadu

[2]
discount_rate=10
tax_rate=5
total_cost=df["Sales"].sum()
discount_amount=(discount_rate/100)*total_cost
tax_amount=(tax_rate/100)*(total_cost-discount_amount)
total_amount=total_cost-discount_amount+tax_amount

[3]
print("Total Cost is:",total_cost)
print("Discount amount is:",discount_amount)
print("Tax amount is:",tax_amount)
print("Total amount is:",total_amount)

Total Cost is: 1495692
Discount amount is: 20000000000
Tax amount is: 673064.1900000001
Total amount is: 14134347.99

```

EXP_8 A Company that sells products online and the Data Scientist been tasked with Analyzing the Sales data for the past month.

```

File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all

Files
sample_data
customer.csv
e commerce.csv
online_retail.csv

[1]
import pandas as pd

[2]
df=pd.read_csv("online_retail.csv",encoding="latin1")
print(df.head())

index InvoiceNo StockCode Description Quantity \
0 0 536365 85123A WHITE HANGING HEART T-LIGHT CANDLE 6
1 1 536365 85123A WHITE METAL LANTERN 6
2 2 536365 844068 CREAM CUPID HEARTS COAT HANGER 8
3 3 536365 840096 KNITTED UNION FLAG HOT WATER BOTTLE 6
4 4 536365 840096 RED WOOLLY HOTTIE WHITE HEART. 6

InvoiceDate UnitPrice CustomerID Country
0 12/1/2010 8:26 2.55 17850.0 United Kingdom
1 12/1/2010 8:26 3.39 17850.0 United Kingdom
2 12/1/2010 8:26 2.55 17850.0 United Kingdom
3 12/1/2010 8:26 3.39 17850.0 United Kingdom
4 12/1/2010 8:26 3.39 17850.0 United Kingdom

[3]
df["InvoiceDate"] = pd.to_datetime(df["InvoiceDate"])
last_date=df["InvoiceDate"].max()
last_month_date=last_date-pd.Timedelta(days=30)
last_month_data=df[df["InvoiceDate"] >= last_month_date]

[4]
product_sales = last_month_data.groupby("Description")["Quantity"].sum()

[5]
top_5_products = product_sales.sort_values(ascending=False).head(5)
print("Top 5 products sold in the past month:")
print(top_5_products)

```

```

File Edit View Insert Runtime Tools Help
Commands + Code - Text Run all
Files
sample_data
customer.csv
e commerce.csv
online_retail.csv
online_retail.csv

[116] In [116]
[116] Out
[117] In [117]
[117] Out
[118] In [118]
[118] Out
[119] In [119]
[119] Out

```

InvoiceDate UnitPrice CustomerID Country

0	12/1/2010 8:26	2.55	536365	United Kingdom
1	12/1/2010 8:26	3.39	536365	United Kingdom
2	12/1/2010 8:26	2.75	536365	United Kingdom
3	12/1/2010 8:26	3.39	536365	United Kingdom
4	12/1/2010 8:26	3.39	536365	United Kingdom

df["InvoiceDate"] = pd.to_datetime(df["InvoiceDate"])
latest_date = df["InvoiceDate"].max()
last_month_date = latest_date - pd.Timedelta(days=30)
last_month_data = df[df["InvoiceDate"] >= last_month_date]

product_sales = last_month_data.groupby("Description")["Quantity"].sum()

top_5_products = product_sales.sort_values(ascending=False).head(5)
print("Top 5 products sold in the past month:")
print(top_5_products)

... Top 5 products sold in the past month:

Description	Quantity
WORLD WAR 2 GLIDERS ASSTD DESIGNS	8169
POPCORN HOLDER	6602
RABBIT NIGHT LIGHT	5719
JUMBO METAL METROSPOT	4693
60 CAKE CASES VINTAGE CHRISTMAS	3468

Name: Quantity, dtype: int64

EXP_9 Real Estate Agency and have been given a Dataset containing Information about Properties for sale.

```

File Edit View Insert Runtime Tools Help
Commands + Code - Text Run all
Files
sample_data
customer.csv
e commerce.csv
online_retail.csv
properties.csv
Disk 86.51 GB available

[76] In [76]
[76] Out
[77] In [77]
[77] Out
[78] In [78]
[78] Out
[79] In [79]
[79] Out

```

EXP_9 Real Estate Agency and have been given a Dataset containing Information about Properties for sale.

```

import pandas as pd

property_data = pd.read_csv("properties.csv", low_memory=False)
print(property_data.head())
print(property_data.columns)

area_type availability location size \
0 Super built-up Area 19-Dec Electronic City Phase II 2 BHK
1 Plot Area Ready To Move Chikka Tirupathi 4 Bedroom
2 Built-up Area Ready To Move Uttarahalli 3 BHK
3 Super built-up Area Ready To Move Lingadheranahalli 3 BHK
4 Super built-up Area Ready To Move Kothamur 2 BHK

society total_sqft bath balcony price
0 Coomee 1056 2.0 1.0 39.07
1 Theamap 2600 5.0 3.0 120.00
2 NaN 1440 2.0 3.0 62.00
3 Soliure 1521 3.0 1.0 95.00
4 NaN 3200 4.0 1.0 51.00
Index(['area_type', 'availability', 'location', 'size', 'society',
       'total_sqft', 'bath', 'balcony', 'price'],
      dtype='object')

property_data["bedrooms"] = (
    property_data["size"]
    .str.extract(r'(\d+)')
    .astype(float)
)

```

avg_price_per_location = property_data.groupby("location")["price"].mean()

A screenshot of a Jupyter Notebook interface. The left sidebar shows a file tree with 'sample_data' containing 'customer.csv', 'e commerce.csv', 'online_retail.csv', and 'properties.csv'. The main area displays the following code and its output:

```
avg_price_per_location = property_data.groupby("location")["price"].mean()
print("Average listing price of properties in each location:")
print(avg_price_per_location)

Average listing price of properties in each location:
location
Anekal          16.000000
Bannavadi       35.000000
Basavangudi      50.000000
Bhoganhalli     22.890000
Devarabeesana Halli 124.833333
...
t.c palya        169.000000
tc-palya         60.750000
vijaynagar       200.000000
white field,kadugodi 275.000000
whitefield       32.730000
Name: price, Length: 1305, dtype: float64
```

```
properties_more_than_4_bedrooms = property_data[property_data["bedrooms"] > 4].shape[0]
print("Number of properties with more than four bedrooms:")
print(properties_more_than_4_bedrooms)

Number of properties with more than four bedrooms:
846
```

```
largest_area_property = property_data.loc[
    property_data["total_sqft"].idxmax()
]
print("\nProperty with the largest area:")
print(largest_area_property)

Property with the largest area:
area_type      Built-up Area
availability   Ready To Move
location       BEML Layout
size           2 BHK
sector        Nall
total_sqft     999
bath          3.0
balcony        3.0
price          45.0
bedrooms       2.0
Name: 1369, dtype: object
```

A screenshot of a Jupyter Notebook interface. The left sidebar shows a file tree with 'sample_data' containing 'customer.csv', 'e commerce.csv', 'online_retail.csv', and 'properties.csv'. The main area displays the following code and its output:

```
properties_more_than_4_bedrooms = property_data[property_data["bedrooms"] > 4].shape[0]
print("Number of properties with more than four bedrooms:")
print(properties_more_than_4_bedrooms)

Number of properties with more than four bedrooms:
846
```

```
largest_area_property = property_data.loc[
    property_data["total_sqft"].idxmax()
]
print("\nProperty with the largest area:")
print(largest_area_property)

Property with the largest area:
area_type      Built-up Area
availability   Ready To Move
location       BEML Layout
size           2 BHK
sector        Nall
total_sqft     999
bath          3.0
balcony        3.0
price          45.0
bedrooms       2.0
Name: 1369, dtype: object
```

EXP_10 Data visualization project and need to create basic plots using Matplotlib.

A screenshot of a Jupyter Notebook interface. On the left, there's a sidebar with file navigation and a disk status bar. The main area shows a code cell with three lines of Python code:

```
import pandas as pd  
import matplotlib.pyplot as plt  
  
data = pd.read_csv("data_visualization_cars.csv")
```

Below the code cell is another cell containing the following code:

```
plt.figure()  
plt.plot(data["Month"], data["Sales"])  
plt.xlabel("Month")  
plt.ylabel("Sales")  
plt.title("Monthly Sales Line Plot")  
plt.show()
```

The output of this second cell is a line plot titled "Monthly Sales Line Plot" showing monthly sales data with significant fluctuations.



A screenshot of a Google Chrome browser window displaying a Jupyter Notebook cell. The cell contains a line of Python code to generate a bar plot:

```
plt.figure()  
plt.bar(data["Month"], data["Sales"])  
plt.xlabel("Month")  
plt.ylabel("Sales")  
plt.title("Monthly Sales Bar Plot")  
plt.show()
```

The output of this cell is a bar chart titled "Monthly Sales Bar Plot" showing monthly sales data as blue bars. The x-axis is labeled "Month" and the y-axis is labeled "Sales". The sales values are represented by the height of the bars, showing a similar seasonal pattern to the line plot.