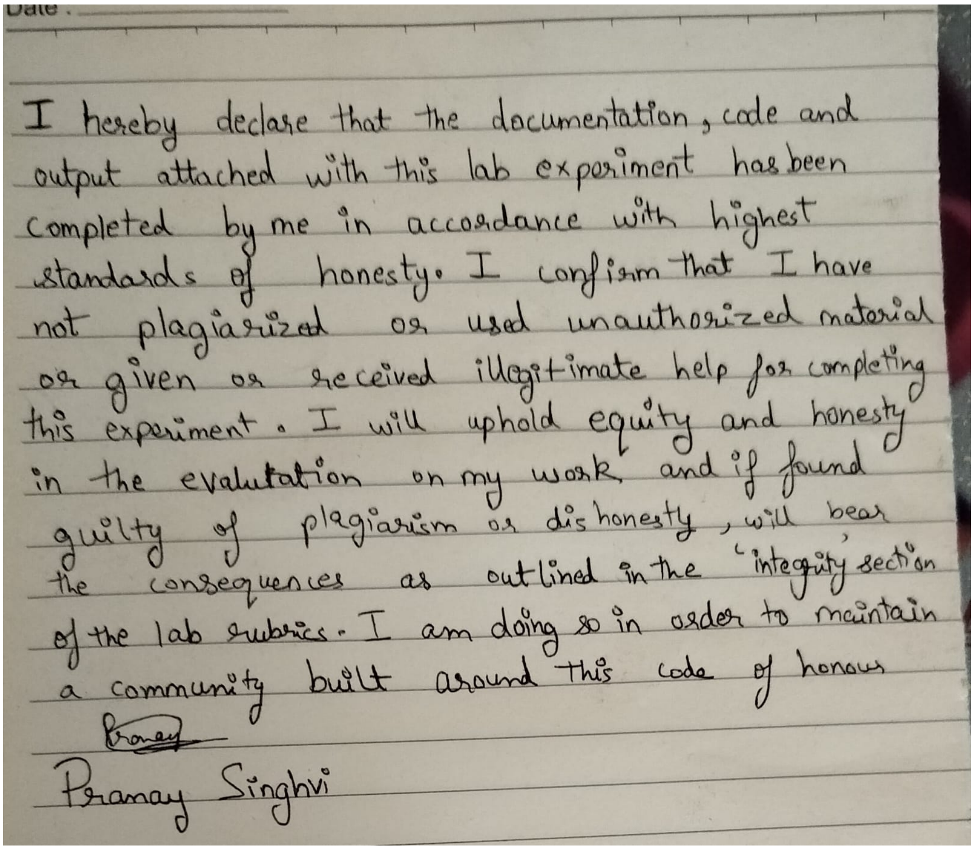
Name	Pranay Singhvi
UID No.	2021300126

Experiment 3

<table>
<tr><td>HONOR PLEDGE</td><td>
Date :

I hereby declare that the documentation, code and output attached with this lab experiment has been completed by me in accordance with highest standards of honesty. I confirm that I have not plagiarized or used unauthorized material or given or received illegitimate help for completing this experiment. I will uphold equity and honesty in the evalutation on my work and if found guilty of plagiarism or dishonesty, will bear the consequences as outlined in the 'integrity' section of the lab rubrics. I am doing so in order to maintain a community built around this code of honour
Pranay Singhvi</td></tr>
<tr><td>PROBLEM STATEMENT</td><td>Data Integration and Reshaping:
1. Merge two or more data frames based on a common key:
Create a new pandas dataframe with containing 20 records and 5 attributes. One attribute should compulsorily be a categorical variable, which is common with a categorical attribute from the CSV dataset that has been used earlier by you. The other 4 attributes can be generated on reasonable assumptions. Merge these 2 datasets on the common key

2. Concatenate multiple Data Frames vertically or horizontally:
Create 5 new rows of the same schema as the original csv dataframe. Use a new categorical value for the common key attribute. Concatenate this horizontally with the existing dataframe Similarly, execute a vertical concatenation with a mock dataframe.

3. Pivot a Data Frame from long to wide format or vice versa:
Add reasoning for using pivot. Explain with a relevant example how pivot operation is useful in data analysis

4. Stack and unstack columns or levels in the Data Frame:
Reason about the use and application of stacking and unstacking with the help of the current dataframe or another example.

5. Data Wrangling:</td></tr>
</table>

	Experiment with other techniques in data wrangling to convert and reshape your dataframe into its final state which can be used for analysis
THEORY	1. **Merging Data Frames** Data frames frequently contain information from distinct sources, each contributing unique insights. The process of merging serves as a bridge, facilitating the combination of these frames based on shared identifiers. Consider delving into the analysis of customer purchase history as an example. One data frame could encompass customer demographics, while another captures their transactional data. Through merging, one can delve into exploring the interplay between demographics and purchasing behavior, ultimately constructing a comprehensive and holistic understanding. **For this Dataset:** I combined the initial dataframe with a supplementary one that introduces five additional attributes: location, state, revenue, number of employees, and product_type. The merging process involved executing an inner join based on the shared key 'Location' to integrate the two dataframes. 2. **Concatenation of Data Frames** Sometimes, adding data means expanding dimensions. Concatenation empowers you to do just that, either vertically or horizontally. Vertical concatenation, stacking data frames like building blocks, increases the number of observations. Picture analyzing sales data across multiple stores. Stacking monthly sales data from each store creates a comprehensive timeline for analysis. Horizontal concatenation, joining data frames side-by-side, adds new features. Think about adding weather data to sales data. Concatenation horizontally creates a richer dataset for exploring sales-weather relationships. **For this Dataset:** I performed concatenation on the dataframes in both vertical and horizontal orientations. For the horizontal concatenation, I included an additional 5 rows adhering to the same schema as the original CSV dataframe, with "Location: Jodhpur" assigned as a new attribute value. On the other hand, vertical concatenation involved the utilization of a new Remarks column to facilitate the combination of the dataframes. 3. **Pivoting Data Frames** Pivot tables are data transformers, summarizing and reorganizing data from a wide format to a more condensed and insightful one. Imagine a vast sales dataset with rows for each transaction, listing product, customer, price, and quantity. A pivot table can group by product and customer, calculating quantities sold and average prices. This condensed view instantly reveals top-selling products for each customer, uncovering valuable buying patterns. **For this Dataset:** I transformed the dataframe from a long to a wide format by employing the pivot_table function. In this process, I designated 'City Location' as the index, 'Remarks' as the columns, and 'Amount in USD' as the values. 4. **Stacking/Unstacking Data Frames** Data often needs reshaping for specific analysis needs. Stacking

and unstacking are powerful tools for this. Stacking takes wide data (many columns) and condenses it into long data (fewer columns with repeated values). Imagine a dataset with city, year, and separate sales figures for each product. Stacking creates rows for each city-year combination, with columns for product and sales, making it easier to analyze trends across products and time. Unstacking reverses this, going from long to wide format. It might be needed for heatmaps or visualizations that require product-specific sales figures for each city-year.

For this Dataset:
I utilized the stack() function to reconfigure the dataframe from a wide to a long format, effectively stacking it. Additionally, I employed the unstack() function to revert the dataframe from a long back to a wide format, unstacking it for a different perspective.

5. **Data Wrangling**
Data wrangling is the often invisible but crucial first step in any data analysis journey. It's the art of cleaning, transforming, and organizing raw data into a usable format. Imagine a messy room filled with scattered information. Data wrangling is like decluttering and organizing that room, making it easy to navigate and find what you need. It removes inconsistencies, fixes errors, and formats data appropriately, ensuring the analysis is based on accurate and reliable information. This saves time and effort later, as you're working with clean, ready-to-use data, ultimately leading to more trustworthy and impactful insights.

For this Dataset:
Furthermore, I applied additional data wrangling functions, such as groupby(), mean(), and sort_values(), to conduct analysis on the dataframe. Specifically, I utilized these functions to determine insights like identifying the location with the highest average investment in the dataset.

1. Importing Libraries & Dataset

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# read csv file
df = pd.read_excel('startup.xlsx')
df.head()
```

Out[ ]:

	Unnamed: 0	Date dd/mm/yyyy	Startup Name	Industry Vertical	City Location	Investors Name	In
0	0	09/01/2020	BYJU'S	E-Tech	Bengaluru	Tiger Global Management	
1	1	13/01/2020	Shuttl	Transportation	Gurgaon	Susquehanna Growth Equity	
2	2	09/01/2020	Mamaearth	E-commerce	Bengaluru	Sequoia Capital India	
3	3	02/01/2020	https://www.wealthbucket.in/	FinTech	New Delhi	Vinod Khatumal	
4	4	02/01/2020	Fashor	Fashion and Apparel	Mumbai	Sprout Venture Partners	

2. Merging data frames based on a common key

In [ ]:

```
new_data = {
    'City  Location': np.random.choice(['Bangalore', 'Mumbai', 'Delhi', 'Chennai', 'H
    'Revenue': np.random.randint(1000000, 9999999, 20),
    'Employees': np.random.randint(100, 999, 20),
    'product_type': np.random.choice(['Software', 'Hardware'], size=20),
}
df2 = pd.DataFrame(new_data)
df2['State'] = df2['City  Location'].map({
    'Bangalore': 'Karnataka',
    'Mumbai': 'Maharashtra',
    'Delhi': 'NCR',
    'Chennai': 'Tamil Nadu',
    'Hyderabad': 'Telangana'
})
df2.head()
```

Out[ ]:

	City Location	Revenue	Employees	product_type	State
0	Hyderabad	5522961	782	Software	Telangana
1	Chennai	6902786	678	Hardware	Tamil Nadu
2	Hyderabad	5931917	728	Software	Telangana
3	Chennai	1940443	859	Software	Tamil Nadu
4	Chennai	7595803	838	Software	Tamil Nadu

In [ ]:

```
# perform inner join on df and df2
df3 = pd.merge(df, df2, on='City  Location', how='inner')
df3.head()
```

Out[ ]:

	Unnamed: 0	Date dd/mm/yyyy	Startup Name	Industry Vertical	City Location	Investors Name	InvestmentnType	Amount in USD
0	4	02/01/2020	Fashor	Fashion and Apparel	Mumbai	Sprout Venture Partners	Seed Round	14.403297
1	4	02/01/2020	Fashor	Fashion and Apparel	Mumbai	Sprout Venture Partners	Seed Round	14.403297
2	4	02/01/2020	Fashor	Fashion and Apparel	Mumbai	Sprout Venture Partners	Seed Round	14.403297
3	4	02/01/2020	Fashor	Fashion and Apparel	Mumbai	Sprout Venture Partners	Seed Round	14.403297
4	5	13/01/2020	Pando	Logistics	Chennai	Chiratae Ventures	Series A	16.012735

3. Concatenating multiple Data Frames vertically/horizontally

In [ ]:
```
# create 5 new rows for original dataframe with location value as Jamnagar
extra_data = {
    'Date dd/mm/yyyy': np.random.choice(['01/01/2017', '01/02/2017', '01/03/2017', '0
    'Startup Name': ['Udemy', 'Reliance', 'IDBI Bank', 'Blinkit', 'Pepsico'],
    'Industry Vertical': ['EduTech', 'Telecom', 'Banking', 'IT', 'Food & Beverage'],
    'City  Location': np.repeat('Jodhpur', 5), # changed attribute value
    'Investors Name': ['Softbank', 'Alibaba', 'Tencent', 'Sequoia', 'Accel Partners']
    'InvestmentnType': np.random.choice(['Seed Funding', 'Private Equity'], 5),
    'Amount in USD': np.random.randint(1000000, 9999999, 5),
}
df4 = pd.DataFrame(extra_data)
df4['Amount in USD'] = np.log(df4['Amount in USD'])
df4.head()
```

Out[ ]:

	Date dd/mm/yyyy	Startup Name	Industry Vertical	City Location	Investors Name	InvestmentnType	Amount in USD
0	01/02/2017	Udemy	EduTech	Jodhpur	Softbank	Private Equity	16.022146
1	01/02/2017	Reliance	Telecom	Jodhpur	Alibaba	Seed Funding	15.862235
2	01/05/2017	IDBI Bank	Banking	Jodhpur	Tencent	Private Equity	15.391707
3	01/01/2017	Blinkit	IT	Jodhpur	Sequoia	Seed Funding	15.452335
4	01/04/2017	Pepsico	Food & Beverage	Jodhpur	Accel Partners	Seed Funding	14.070341

In [ ]:
```
# concatenate df and df4, horizontally
df5 = pd.concat([df, df4], axis=0)
df5.tail()
```

Out[ ]:

	Unnamed: 0	Date dd/mm/yyyy	Startup Name	Industry Vertical	City Location	Investors Name	InvestmentnType	Amount in USD
0	NaN	01/04/2017	Udemy	EduTech	Jamnagar	Softbank	Seed Funding	14.531252
1	NaN	01/03/2017	Reliance	Telecom	Jamnagar	Alibaba	Private Equity	15.949679
2	NaN	01/02/2017	IDBI Bank	Banking	Jamnagar	Tencent	Private Equity	15.418290
3	NaN	01/03/2017	Blinkit	IT	Jamnagar	Sequoia	Private Equity	15.410821
4	NaN	01/02/2017	Pepsico	Food & Beverage	Jamnagar	Accel Partners	Seed Funding	16.027653

In [ ]:
```
# creating new df with remarks column with good, average or badn values
df6 = pd.DataFrame({
    'Remarks': np.random.choice(['Good', 'Average', 'Bad'], size=df5.shape[0])
})
df6.head()
```

Out[ ]:

	Remarks
0	Bad
1	Good
2	Average
3	Good
4	Good

In [ ]:

```
# concatenate df5 and df6, vertically
df5.reset_index(drop=True, inplace=True)
df7 = pd.concat([df5, df6], axis=1)
df7.head()
```

Out[ ]:

	Unnamed: 0	Date dd/mm/yyyy	Startup Name	Industry Vertical	City Location	Investors Name	In
0	0.0	09/01/2020	BYJU'S	E-Tech	Bengaluru	Tiger Global Management	
1	1.0	13/01/2020	Shuttl	Transportation	Gurgaon	Susquehanna Growth Equity	
2	2.0	09/01/2020	Mamaearth	E-commerce	Bengaluru	Sequoia Capital India	
3	3.0	02/01/2020	https://www.wealthbucket.in/	FinTech	New Delhi	Vinod Khatumal	
4	4.0	02/01/2020	Fashor	Fashion and Apparel	Mumbai	Sprout Venture Partners	

4. Pivoting a Data Frame from long to wide format or vice versa

In [ ]:
```
# pivoting df7 with Location as index
df8 = df7.pivot_table(index='City  Location', columns='Remarks', values='Amount in US
df8.head()
```

Out[ ]:

Remarks	Average	Bad	Good
City Location			
Agra	NaN	0.000000	NaN
Ahmedabad	13.011931	7.314568	11.575955
Amritsar	NaN	NaN	12.611538
Andheri	NaN	NaN	15.564710
Belgaum	NaN	13.122363	NaN

5. Stacking & unstacking columns/levels in the Data Frame

In [ ]:
```
# stacking df
df9 = df.stack()
df9.head()
```

Out[ ]:
```
0  Unnamed: 0                 0
   Date dd/mm/yyyy   09/01/2020
   Startup Name          BYJU'S
   Industry Vertical     E-Tech
   City  Location     Bengaluru
dtype: object
```

In [ ]:
```
# unstacking df
df10 = df9.unstack()
df10.head()
```

Out[ ]:

	Unnamed: 0	Date dd/mm/yyyy	Startup Name	Industry Vertical	City Location	Investors Name	In
0	0	09/01/2020	BYJU'S	E-Tech	Bengaluru	Tiger Global Management	
1	1	13/01/2020	Shuttl	Transportation	Gurgaon	Susquehanna Growth Equity	
2	2	09/01/2020	Mamaearth	E-commerce	Bengaluru	Sequoia Capital India	
3	3	02/01/2020	https://www.wealthbucket.in/	FinTech	New Delhi	Vinod Khatumal	
4	4	02/01/2020	Fashor	Fashion and Apparel	Mumbai	Sprout Venture Partners	

6. Data Wrangling

In [ ]:

```
df11 = df.copy()
# reversing log transformation on Amount in USD column
df11['Amount in USD'] = np.exp(df11['Amount in USD'])
# displaying the mean of Amount in USD for each location in descending order
df11 = df11.groupby('City  Location')['Amount in USD'].mean().sort_values(ascending=F
df11['Amount in USD'] = df11['Amount in USD'].round(2).astype(int)
df11.head()
```

Out[ ]:

	City Location	Amount in USD
0	Menlo Park	450000000
1	California	300000000
2	Tulangan	200000000
3	Kormangala	142000000
4	Santa Monica	110000000

CONCLUSION	In this experiment, we acquired the skills to merge two data frames by leveraging a common key and explored the techniques of concatenating them, whether horizontally or vertically. Additionally, we delved into the processes of pivoting a dataframe and stacking/unstacking it to reshape its structure. Finally, we incorporated supplementary data wrangling functions such as groupby, sort_values, and others to conduct in-depth analysis on the dataframe.