📊 Exploratory Data Analysis (EDA) on Restaurant Dataset

📌 1. Introduction & Setup

📌 2. Exploring the DataFrame

- `df.shape`
- `df.head()`
- `df.tail()`
- `df.sample()`
- `df.info()`
- `df.describe()`

📌 3. Working with Columns

- `df['Column name']`
- `df[['col 1','col 2']]`
- `df['column name'].numerical_function_name()`
- `df['column name'].value_counts()`
- `df['column_name'].unique()`
- `df['new_column name']=--`

📌 4. Working with Rows

- `df[df['column_name']=="condition"]`
- `&`
- `|`
- `== or .str.contains`
- `df.iloc[3]`
- `df.iloc[slicing]`

Let's solve some questions using real dataset

```
!gdown "https://drive.google.com/uc?export=download&id=1qgMZdOpZ_KJAgqc46WW5pvG7xgkJ2jQ6" -O restaurants_all.csv
import pandas as pd # explain pd here
df1 = pd.read_csv('restaurants_all.csv')
df1.shape
```

```
Downloading...
From (original): https://drive.google.com/uc?export=download&id=1qgMZdOpZ_KJAgqc46WW5pvG7xgkJ2jQ6
From (redirected): https://drive.google.com/uc?export=download&id=1qgMZdOpZ_KJAgqc46WW5pvG7xgkJ2jQ6&confirm=t&uuid=cbb8f
To: /content/restaurants_all.csv
100% 190M/190M [00:02<00:00, 86.2MB/s]
(224520, 17)
```

```
df1.head()
```

	zomato_url	name	city	area	rating	rating_count	telephone	cusine	cost_for_two	a
0	https://www.zomato.com/ncr/sainik-food-pandav-...	Sainik Food	Delhi NCR	Pandav Nagar	3.2	21.0	011 22486474 +91 9717806814	North Indian	300.0	
1	https://www.zomato.com/mumbai/kunals-creamery-...	Kunal's Creamery & Eatery	Mumbai	Ambernath	3.6	51.0	+91 9561356690 +91 9637537499	Street Food, Chinese, Fast Food	500.0	S
2	https://www.zomato.com/ncr/brij-palace-restaur...	Brij Palace Restaurant	Delhi NCR	Jasola	NaN	NaN	+91 9891828106	North Indian	250.0	
3	https://www.zomato.com/ncr/sahib-hotel-paharga...	Sahib Hotel	Delhi NCR	Paharganj	NaN	NaN	+91 9670005455	North Indian	300.0	
4	https://www.zomato.com/kolkata/chunkys-shibpur...	Chunky's	Kolkata	Shibpur	3.0	78.0	+91 8442828284	Italian, Pizza, Continental	500.0	

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 224520 entries, 0 to 224519
Data columns (total 17 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   zomato_url         224520 non-null  object
 1   name               224520 non-null  object
 2   city               224520 non-null  object
 3   area               224520 non-null  object
 4   rating             144735 non-null  float64
 5   rating_count       142397 non-null  float64
 6   telephone          222930 non-null  object
 7   cusine             223190 non-null  object
 8   cost_for_two       220872 non-null  float64
 9   address            222734 non-null  object
 10  timings            221556 non-null  object
 11  online_order       224520 non-null  bool
 12  table_reservation  224520 non-null  bool
 13  delivery_only      224520 non-null  bool
 14  famous_food        52526 non-null   object
 15  longitude          224511 non-null  float64
 16  latitude           224511 non-null  float64
dtypes: bool(3), float64(5), object(9)
memory usage: 24.6+ MB
```

How many number of records do we have?

```
df1.shape[0]
```

```
224520
```

Calculate the number of restaurants in Mumbai

```
df1[df1['city']=='Mumbai'].shape[0]
```

```
25692
```

On an average which is most costly for dine-in, Mumbai or Delhi NCR?

```
df1[df1["city"] == "Mumbai"]["cost_for_two"].mean(), df1[df1["city"] == "Delhi NCR"]["cost_for_two"].mean()
```

```
(496.29071085231936, 453.5377245508982)
```

How many unique restaurants do we have?

```
df1["name"].unique().shape[0]
```

```
146659
```

Let's get started with today's class

```
import pandas as pd
!gdown  "https://drive.google.com/uc?export=download&id=1in_bJYfJ5Ahy8y9RcBhdqp0aKHguU7EX" -O dummy_restaurants.csv
df = pd.read_csv('dummy_restaurants.csv')
```

```
Downloading...
From: https://drive.google.com/uc?export=download&id=1in_bJYfJ5Ahy8y9RcBhdqp0aKHguU7EX
To: /content/dummy_restaurants.csv
100% 1.50k/1.50k [00:00<00:00, 5.49MB/s]
```

```
df
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
0	Gabbar's Bar & Kitchen	Kolkata	4.3	1500	North Indian, Chinese, Mexican, Italian	False	True
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.6	100	North Indian	True	False
2	YellowKrust	Delhi NCR	3.8	400	Bakery, Desserts	False	False
3	Hungry Minister	Delhi NCR	3.7	400	Continental, Mexican, Fast Food, Chinese	True	False
4	Night Food Xprs	Delhi NCR	3.3	500	North Indian, Chinese	True	False
5	HOP House of Proteins	Delhi NCR	4.8	600	Continental, North Indian, Healthy Food, Bever...	True	False
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
7	Swad Restaurant	Delhi NCR	3.8	750	North Indian, Chinese, Mughlai	True	False
8	Chai Peeni Hai	Delhi NCR	3.7	200	Tea, Fast Food	True	False
9	Baskin Robbins	Mumbai	3.8	200	Ice Cream, Desserts, Beverages	False	False
10	Spirit	Mumbai	NaN	850	Chinese, North Indian, Mughlai	NaN	NaN
11	Purple Box	Mumbai	3.9	200	Fast Food, Beverages, Desserts	True	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False
13	Talli Pangs	Delhi NCR	3.7	300	Fast Food, North Indian	False	False
14	Just Baked	Kolkata	3.3	250	Bakery, Fast Food	False	False

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   name               20 non-null     object
 1   city               20 non-null     object
 2   rating             18 non-null     float64
 3   cost_for_two       20 non-null     int64
 4   cusine             20 non-null     object
 5   online_order       18 non-null     object
 6   table_reservation  18 non-null     object
dtypes: float64(1), int64(1), object(5)
memory usage: 1.2+ KB
```

📌 5. Dealing with Missing Values

We observed there are some null(missing) values in dataset

Why is it important to deal with missing values

Missing values can distort statistical calculations, and affect sorting, merging, and visualizations. Many functions may fail or behave unexpectedly when NaNs are present.

```
import pandas as pd
import numpy as np

# Sample dataset with missing values
data = {'Name': ['Alice', 'Bob', 'Charlie', 'David'],
        'Age': [25, np.nan, 30, 40],
        'City': ['NY', 'LA', np.nan, 'SF']}

df = pd.DataFrame(data)
print(df)
```

```
      Name   Age City
0    Alice  25.0   NY
1      Bob   NaN   LA
2  Charlie  30.0  NaN
3    David  40.0   SF
```

```
a=5/0
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
<ipython-input-8-7ab028d636d0> in <cell line: 0>()
----> 1 a=5/0

ZeroDivisionError: division by zero
```

When you try to divide a number by zero in Python, it doesn't know how to handle this scenario mathematically, as division by zero is undefined and it gives an error

```
data = {'Value': [10, 20, 0, 0]}
df1 = pd.DataFrame(data)

# Perform division by zero (0 / 0)
df1['Result'] = df1['Value'] / df1['Value']  # This will result in NaN for 0 / 0

print(df1)
```

```
   Value  Result
0     10     1.0
1     20     1.0
2      0     NaN
3      0     NaN
```

In pandas, NaN (Not a Number) is used to represent missing, undefined, or invalid data in a DataFrame.It is a special floating-point value that allows for seamless handling of missing values in large datasets without causing errors, enabling flexible data analysis and manipulation.

🔷 Identifying Missing Data

First, we check whether the missing values exist using .isna().

```
# Check for missing values
df.isna()  # Returns True where data is missing
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False
7	False	False	False	False	False	False	False
8	False	False	False	False	False	False	False
9	False	False	False	False	False	False	False
10	False	False	True	False	False	True	True
11	False	False	False	False	False	False	False
12	False	False	False	False	False	False	False
13	False	False	False	False	False	False	False
14	False	False	False	False	False	False	False
15	False	False	False	False	False	False	False
16	False	False	False	False	False	True	True
17	False	False	True	False	False	False	False
18	False	False	False	False	False	False	False
19	False	False	False	False	False	False	False

```
# Check for missing values for a column
df['rating'].isna()
```

	rating
0	False
1	False
2	False
3	False
4	False
5	False
6	False
7	False
8	False
9	False
10	True
11	False
12	False
13	False
14	False
15	False
16	False
17	True
18	False
19	False

dtype: bool

```
# Find the number of missing values in each column., will count() work ??
df['rating'].isna().count() # No, because it gives count of all values in column
```

```
20
```

```
df['rating'].isna().sum() # Gives no. of missing values for rating column
```

2

now , we know missing values are ther , how to handle them ?

🔷 Handling Missing Data -

There are two main approaches: 🔷 A. Removing Missing Values 🔷 B. Filling Missing Values

Fill missing values using .fillna()

```
df['rating'].fillna(df['rating'].mean())  #fillna() is used to replace NaN values in a DataFrame or Series with a specified
```

	rating
0	4.30
1	3.60
2	3.80
3	3.70
4	3.30
5	4.80
6	3.40
7	3.80
8	3.70
9	3.80
10	3.85
11	3.90
12	3.40
13	3.70
14	3.30
15	3.40
16	4.80
17	3.85
18	4.30
19	4.30

dtype: float64

Let's check if changes are reflected in the original dataset.

```
df['rating'].isna().sum() #still not reflected , what to do now ?
```

2

```
df['rating']=df['rating'].fillna(df['rating'].mean()) #changes are made now
df
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
0	Gabbar's Bar & Kitchen	Kolkata	4.30	1500	North Indian, Chinese, Mexican, Italian	False	True
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.60	100	North Indian	True	False
2	YellowKrust	Delhi NCR	3.80	400	Bakery, Desserts	False	False
3	Hungry Minister	Delhi NCR	3.70	400	Continental, Mexican, Fast Food, Chinese	True	False
4	Night Food Xprs	Delhi NCR	3.30	500	North Indian, Chinese	True	False
5	HOP House of Proteins	Delhi NCR	4.80	600	Continental, North Indian, Healthy Food, Bever...	True	False
6	Daana Paani Family Restaurant	Kolkata	3.40	300	North Indian, Chinese	True	False
7	Swad Restaurant	Delhi NCR	3.80	750	North Indian, Chinese, Mughlai	True	False
8	Chai Peeni Hai	Delhi NCR	3.70	200	Tea, Fast Food	True	False
9	Baskin Robbins	Mumbai	3.80	200	Ice Cream, Desserts, Beverages	False	False
10	Spirit	Mumbai	3.85	850	Chinese, North Indian, Mughlai	NaN	NaN
11	Purple Box	Mumbai	3.90	200	Fast Food, Beverages, Desserts	True	False
12	Mio Amore	Kolkata	3.40	150	Bakery	False	False
13	Talli Pangs	Delhi NCR	3.70	300	Fast Food, North Indian	False	False
14	Just Baked	Kolkata	3.30	250	Bakery, Fast Food	False	False

```
df['rating'].isna().sum()
```

```
0
```

We also have another approach by using parameter "inplace" but we prefer direct assignment

```
df['rating'].fillna(df['rating'].mean(), inplace=True)
df
```

```
<ipython-input-17-fdf8d2a0367f>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[c
```

```
df['rating'].fillna(df['rating'].mean(), inplace=True)
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
0	Gabbar's Bar & Kitchen	Kolkata	4.30	1500	North Indian, Chinese, Mexican, Italian	False	True
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.60	100	North Indian	True	False
2	YellowKrust	Delhi NCR	3.80	400	Bakery, Desserts	False	False
3	Hungry Minister	Delhi NCR	3.70	400	Continental, Mexican, Fast Food, Chinese	True	False
4	Night Food Xprs	Delhi NCR	3.30	500	North Indian, Chinese	True	False
5	HOP House of Proteins	Delhi NCR	4.80	600	Continental, North Indian, Healthy Food, Bever...	True	False
6	Daana Paani Family Restaurant	Kolkata	3.40	300	North Indian, Chinese	True	False
7	Swad Restaurant	Delhi NCR	3.80	750	North Indian, Chinese, Mughlai	True	False
8	Chai Peeni Hai	Delhi NCR	3.70	200	Tea, Fast Food	True	False
9	Baskin Robbins	Mumbai	3.80	200	Ice Cream, Desserts, Beverages	False	False
10	Spirit	Mumbai	3.85	850	Chinese, North Indian, Mughlai	NaN	NaN
11	Purple Box	Mumbai	3.90	200	Fast Food, Beverages, Desserts	True	False
12	Mio Amore	Kolkata	3.40	150	Bakery	False	False
13	Talli Pangs	Delhi NCR	3.70	300	Fast Food, North Indian	False	False
14	Just Baked	Kolkata	3.30	250	Bakery, Fast Food	False	False
15	Monginis	Mumbai	3.40	200	Bakery, Fast Food	True	False

```
df.fillna({'rating': df['rating'].mean()},inplace=True)
df
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
0	Gabbar's Bar & Kitchen	Kolkata	4.30	1500	North Indian, Chinese, Mexican, Italian	False	True
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.60	100	North Indian	True	False
2	YellowKrust	Delhi NCR	3.80	400	Bakery, Desserts	False	False
3	Hungry Minister	Delhi NCR	3.70	400	Continental, Mexican, Fast Food, Chinese	True	False
4	Night Food Xprs	Delhi NCR	3.30	500	North Indian, Chinese	True	False
5	HOP House of Proteins	Delhi NCR	4.80	600	Continental, North Indian, Healthy Food, Bever...	True	False
6	Daana Paani Family Restaurant	Kolkata	3.40	300	North Indian, Chinese	True	False
7	Swad Restaurant	Delhi NCR	3.80	750	North Indian, Chinese, Mughlai	True	False
8	Chai Peeni Hai	Delhi NCR	3.70	200	Tea, Fast Food	True	False
9	Baskin Robbins	Mumbai	3.80	200	Ice Cream, Desserts, Beverages	False	False
10	Spirit	Mumbai	3.85	850	Chinese, North Indian, Mughlai	NaN	NaN
11	Purple Box	Mumbai	3.90	200	Fast Food, Beverages, Desserts	True	False
12	Mio Amore	Kolkata	3.40	150	Bakery	False	False
13	Talli Pangs	Delhi NCR	3.70	300	Fast Food, North Indian	False	False
14	Just Baked	Kolkata	3.30	250	Bakery, Fast Food	False	False

The inplace=True argument in Pandas functions modifies the DataFrame directly, making the changes without returning a new object

Dropping rows with missing values

```
df.dropna(subset=['rating'])
  # Dropping rows with missing values (use cautiously), row with 10th and 17th index is dropped i.e where we had missing val
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
0	Gabbar's Bar & Kitchen	Kolkata	4.3	1500	North Indian, Chinese, Mexican, Italian	False	True
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.6	100	North Indian	True	False
2	YellowKrust	Delhi NCR	3.8	400	Bakery, Desserts	False	False
3	Hungry Minister	Delhi NCR	3.7	400	Continental, Mexican, Fast Food, Chinese	True	False
4	Night Food Xprs	Delhi NCR	3.3	500	North Indian, Chinese	True	False
5	HOP House of Proteins	Delhi NCR	4.8	600	Continental, North Indian, Healthy Food, Bever...	True	False
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
7	Swad Restaurant	Delhi NCR	3.8	750	North Indian, Chinese, Mughlai	True	False
8	Chai Peeni Hai	Delhi NCR	3.7	200	Tea, Fast Food	True	False
9	Baskin Robbins	Mumbai	3.8	200	Ice Cream, Desserts, Beverages	False	False
11	Purple Box	Mumbai	3.9	200	Fast Food, Beverages, Desserts	True	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False
13	Talli Pangs	Delhi NCR	3.7	300	Fast Food, North Indian	False	False

📌 6. Sorting & Retrieving Top N

```
df
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
0	Gabbar's Bar & Kitchen	Kolkata	4.3	1500	North Indian, Chinese, Mexican, Italian	False	True
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.6	100	North Indian	True	False
2	YellowKrust	Delhi NCR	3.8	400	Bakery, Desserts	False	False
3	Hungry Minister	Delhi NCR	3.7	400	Continental, Mexican, Fast Food, Chinese	True	False
4	Night Food Xprs	Delhi NCR	3.3	500	North Indian, Chinese	True	False
5	HOP House of Proteins	Delhi NCR	4.8	600	Continental, North Indian, Healthy Food, Bever...	True	False
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
7	Swad Restaurant	Delhi NCR	3.8	750	North Indian, Chinese, Mughlai	True	False
8	Chai Peeni Hai	Delhi NCR	3.7	200	Tea, Fast Food	True	False
9	Baskin Robbins	Mumbai	3.8	200	Ice Cream, Desserts, Beverages	False	False
10	Spirit	Mumbai	NaN	850	Chinese, North Indian, Mughlai	NaN	NaN
11	Purple Box	Mumbai	3.9	200	Fast Food, Beverages, Desserts	True	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False
13	Talli Pangs	Delhi NCR	3.7	300	Fast Food, North Indian	False	False
14	Just Baked	Kolkata	3.3	250	Bakery, Fast Food	False	False

If you notice, the rating column is not sorted.

How can we perform sorting in Pandas?

```
df.sort_values(['rating']) #sorts by default in ascending order
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
4	Night Food Xprs	Delhi NCR	3.3	500	North Indian, Chinese	True	False
14	Just Baked	Kolkata	3.3	250	Bakery, Fast Food	False	False
15	Monginis	Mumbai	3.4	200	Bakery, Fast Food	True	False
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.6	100	North Indian	True	False
8	Chai Peeni Hai	Delhi NCR	3.7	200	Tea, Fast Food	True	False
3	Hungry Minister	Delhi NCR	3.7	400	Continental, Mexican, Fast Food, Chinese	True	False
13	Talli Pangs	Delhi NCR	3.7	300	Fast Food, North Indian	False	False
2	YellowKrust	Delhi NCR	3.8	400	Bakery, Desserts	False	False
7	Swad Restaurant	Delhi NCR	3.8	750	North Indian, Chinese, Mughlai	True	False
9	Baskin Robbins	Mumbai	3.8	200	Ice Cream, Desserts, Beverages	False	False
11	Purple Box	Mumbai	3.9	200	Fast Food, Beverages, Desserts	True	False
0	Gabbar's Bar & Kitchen	Kolkata	4.3	1500	North Indian, Chinese, Mexican, Italian	False	True
18	PizzaExpress	Mumbai	4.3	1900	Italian, Pizza	True	True

Rows get sorted based on values in rating column.

By default, values are sorted in ascending order.

How can we sort the rows in descending order?

```
df.sort_values(['rating'], ascending=False)
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
5	HOP House of Proteins	Delhi NCR	4.8	600	Continental, North Indian, Healthy Food, Bever...	True	False
16	The Bohri Kitchen	Mumbai	4.8	600	Mughlai, North Indian, Biryani	NaN	NaN
0	Gabbar's Bar & Kitchen	Kolkata	4.3	1500	North Indian, Chinese, Mexican, Italian	False	True
18	PizzaExpress	Mumbai	4.3	1900	Italian, Pizza	True	True
19	Artusi Ristorante in Piazza Horizon	Delhi NCR	4.3	3500	Italian	False	True
11	Purple Box	Mumbai	3.9	200	Fast Food, Beverages, Desserts	True	False
7	Swad Restaurant	Delhi NCR	3.8	750	North Indian, Chinese, Mughlai	True	False
2	YellowKrust	Delhi NCR	3.8	400	Bakery, Desserts	False	False
9	Baskin Robbins	Mumbai	3.8	200	Ice Cream, Desserts, Beverages	False	False
8	Chai Peeni Hai	Delhi NCR	3.7	200	Tea, Fast Food	True	False
13	Talli Pangs	Delhi NCR	3.7	300	Fast Food, North Indian	False	False
3	Hungry Minister	Delhi NCR	3.7	400	Continental, Mexican, Fast Food, Chinese	True	False
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.6	100	North Indian	True	False
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False

Can we perform sorting on multiple columns? Yes!

```
df.sort_values(['rating', 'cost_for_two'])
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
14	Just Baked	Kolkata	3.3	250	Bakery, Fast Food	False	False
4	Night Food Xprs	Delhi NCR	3.3	500	North Indian, Chinese	True	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False
15	Monginis	Mumbai	3.4	200	Bakery, Fast Food	True	False
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.6	100	North Indian	True	False
8	Chai Peeni Hai	Delhi NCR	3.7	200	Tea, Fast Food	True	False
13	Talli Pangs	Delhi NCR	3.7	300	Fast Food, North Indian	False	False
3	Hungry Minister	Delhi NCR	3.7	400	Continental, Mexican, Fast Food, Chinese	True	False
9	Baskin Robbins	Mumbai	3.8	200	Ice Cream, Desserts, Beverages	False	False
2	YellowKrust	Delhi NCR	3.8	400	Bakery, Desserts	False	False
7	Swad Restaurant	Delhi NCR	3.8	750	North Indian, Chinese, Mughlai	True	False
11	Purple Box	Mumbai	3.9	200	Fast Food, Beverages, Desserts	True	False
0	Gabbar's Bar & Kitchen	Kolkata	4.3	1500	North Indian, Chinese, Mexican, Italian	False	True
18	PizzaExpress	Mumbai	4.3	1900	Italian, Pizza	True	True

What exactly happened here?

Rows were first sorted based on 'rating'

Then, rows with same values of 'rating' were sorted based on 'cost_for_two'

How can we have different sorting orders for different columns in multi-level sorting?

```
# Sorting: Ascending by 'rating' and Descending by 'cost_for_two'
df.sort_values(['rating', 'cost_for_two'], ascending=[True, False])
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
4	Night Food Xprs	Delhi NCR	3.3	500	North Indian, Chinese	True	False
14	Just Baked	Kolkata	3.3	250	Bakery, Fast Food	False	False
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
15	Monginis	Mumbai	3.4	200	Bakery, Fast Food	True	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.6	100	North Indian	True	False
3	Hungry Minister	Delhi NCR	3.7	400	Continental, Mexican, Fast Food, Chinese	True	False
13	Talli Pangs	Delhi NCR	3.7	300	Fast Food, North Indian	False	False
8	Chai Peeni Hai	Delhi NCR	3.7	200	Tea, Fast Food	True	False
7	Swad Restaurant	Delhi NCR	3.8	750	North Indian, Chinese, Mughlai	True	False
2	YellowKrust	Delhi NCR	3.8	400	Bakery, Desserts	False	False
9	Baskin Robbins	Mumbai	3.8	200	Ice Cream, Desserts, Beverages	False	False
11	Purple Box	Mumbai	3.9	200	Fast Food, Beverages, Desserts	True	False
19	Artusi Ristorante in Piazza Horizon	Delhi NCR	4.3	3500	Italian	False	True
18	PizzaExpress	Mumbai	4.3	1900	Italian, Pizza	True	True

Sorting in categorical column?

Pandas sorts alphabetically (A → Z or Z → A).

```
# Sorts the column city in ascending order
df.sort_values('city', ascending=True)
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
19	Artusi Ristorante in Piazza Horizon	Delhi NCR	4.3	3500	Italian	False	True
1	Sardaar Ji De Chole Bhature	Delhi NCR	3.6	100	North Indian	True	False
2	YellowKrust	Delhi NCR	3.8	400	Bakery, Desserts	False	False
3	Hungry Minister	Delhi NCR	3.7	400	Continental, Mexican, Fast Food, Chinese	True	False
4	Night Food Xprs	Delhi NCR	3.3	500	North Indian, Chinese	True	False
5	HOP House of Proteins	Delhi NCR	4.8	600	Continental, North Indian, Healthy Food, Bever...	True	False
7	Swad Restaurant	Delhi NCR	3.8	750	North Indian, Chinese, Mughlai	True	False
8	Chai Peeni Hai	Delhi NCR	3.7	200	Tea, Fast Food	True	False
13	Talli Pangs	Delhi NCR	3.7	300	Fast Food, North Indian	False	False
14	Just Baked	Kolkata	3.3	250	Bakery, Fast Food	False	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False
0	Gabbar's Bar & Kitchen	Kolkata	4.3	1500	North Indian, Chinese, Mexican, Italian	False	True
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
10	Spirit	Mumbai	NaN	850	Chinese, North Indian, Mughlai	NaN	NaN

🔷 Retrieving Top N

Top 5 expensive restaurants

```
df.sort_values('cost_for_two', ascending=False).head(5)
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
19	Artusi Ristorante in Piazza Horizon	Delhi NCR	4.3	3500	Italian	False	True
18	PizzaExpress	Mumbai	4.3	1900	Italian, Pizza	True	True
0	Gabbar's Bar & Kitchen	Kolkata	4.3	1500	North Indian, Chinese, Mexican, Italian	False	True
10	Spirit	Mumbai	NaN	850	Chinese, North Indian, Mughlai	NaN	NaN

Try coding the top 5 least rated restaurant

```
# The top 5 least rated restaurant
df.sort_values(by='rating', ascending=True).head(5)
```

	name	city	rating	cost_for_two	cusine	online_order	table_reservation
4	Night Food Xprs	Delhi NCR	3.3	500	North Indian, Chinese	True	False
14	Just Baked	Kolkata	3.3	250	Bakery, Fast Food	False	False
15	Monginis	Mumbai	3.4	200	Bakery, Fast Food	True	False
6	Daana Paani Family Restaurant	Kolkata	3.4	300	North Indian, Chinese	True	False
12	Mio Amore	Kolkata	3.4	150	Bakery	False	False

What if I want to calculate average cost for two for all the cities in my dataset, how will I do?

Can one by one find for each city or run a loop ?

```
df[df["city"] == "Delhi NCR"]["cost_for_two"].mean()
```

```
750.0
```

```
df[df["city"] == "Mumbai"]["cost_for_two"].mean()
```

```
678.5714285714286
```

```
# Filtering rows where 'city' is 'Delhi'
delhi_data = df[df['city'] == 'Delhi NCR']

# Calculating the average 'cost_for_two' for Delhi
average_cost = delhi_data['cost_for_two'].mean()

average_cost
```

```
750.0
```

For every city we have to calculate individually

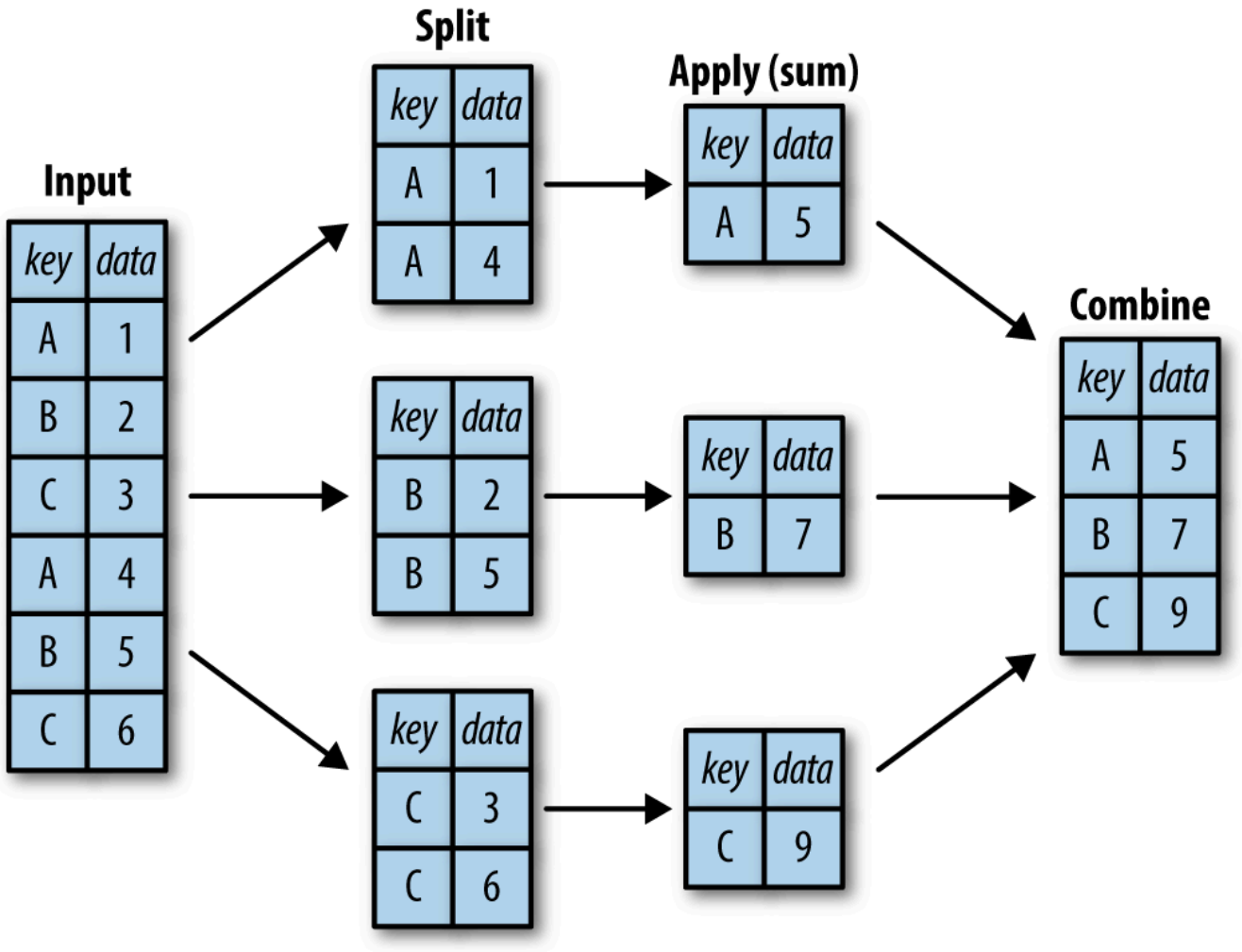
Do we have any simpler solution?

```
# yes, by using Groupby functions
```

📌 7. Basic Groupby Operations

-- The groupby() function in Pandas is used to group data based on a column and apply aggregation functions like sum(), mean(), count(), etc.

In simple terms, we could understood it through - Split, Apply, Combine



Split: Breaking up and grouping a DataFrame depending on the value of the specified key.

Apply: Computing some function, usually an aggregate, transformation, or filtering, within the individual groups.

Combine: Merging the results of these operations into an output array.

🔷 Groupby Syntax

df.groupby('column to group')['column to aggregate'].aggregate function

```
df.groupby('city')['cost_for_two'].mean()  # Average cost per city
```

city	cost_for_two
Delhi NCR	750.000000
Kolkata	550.000000
Mumbai	678.571429

dtype: float64

```
#How many restaurants are present in each city?
df.groupby('city')['name'].count()
```

city	name
Delhi NCR	9
Kolkata	4
Mumbai	7

dtype: int64

```
#What is the maximum cost_for_two in each city?
df.groupby('city')['cost_for_two'].max()
```

city	cost_for_two
Delhi NCR	3500
Kolkata	1500
Mumbai	1900

dtype: int64

```
# Which city has highest price variation cost for dining?
df.groupby('city')['cost_for_two'].std()
```

city	cost_for_two
Delhi NCR	1050.000000
Kolkata	636.396103
Mumbai	609.547061

dtype: float64

What if i want to know whether one column influence other? How did we find the relationship between two variables in descriptive statistics?

📌 8. Correlation

Statistical measure that expresses the extent to which two variables are linearly related

```
df['cost_for_two'].corr(df['table_reservation'])
```

```
0.8746765931639169
```

0.8746 (Strong positive correlation)

This suggests that restaurants that offer table reservations tend to be more expensive.

```
# does table reservation influence ratings ?
df['table_reservation'].corr(df['rating'])
```

```
0.5760536088158114
```

```
# What if we find correlation between non numeric and numeric columns?
df['city'].corr(df['table_reservation'])
# it gives an error
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-18-6bf42eca9005> in <cell line: 0>()
      1 # What if we find correlation between non numeric and numeric columns?
----> 2 df['city'].corr(df['table_reservation'])
      3 # it gives an error

                              1 frames
/usr/local/lib/python3.11/dist-packages/pandas/core/base.py in to_numpy(self, dtype, copy, na_value, **kwargs)
    660             values[np.asanyarray(isna(self))] = na_value
    661
```

📌 Summary & Insights

🔎 Key Takeaways:

1 Introduction to Pandas – Pandas is a Python library for data manipulation & analysis, providing structures like Series (1D) and DataFrame (2D).

2 Loading Data – Data can be imported from files such as CSV.

3 Viewing Data – Methods like .head(), .tail(), and .info() help in quickly inspecting the dataset.

4 Selecting Data – Columns can be accessed individually, and rows can be selected using indexing (iloc).

5 Filtering Data – Rows can be filtered using conditions, including multiple criteria with logical operators (&, |).

6 Sorting Data – Data can be arranged in ascending or descending order using .sort_values().

7 Handling Missing Values – Missing values can be detected with .isna() and handled by filling (.fillna()) or dropping (.dropna()).

8 Grouping and Aggregation – Data can be grouped using .groupby() to calculate mean, sum, count, etc.

Let's do some analysis on real dataset :)

```
import pandas as pd
!gdown "https://drive.google.com/uc?export=download&id=1qgMZdOpZ_KJAgqc46WW5pvG7xgkJ2jQ6" -O restaurants_all.csv
df = pd.read_csv('restaurants_all.csv')
```

```
Downloading...
From (original): https://drive.google.com/uc?export=download&id=1qgMZdOpZ_KJAgqc46WW5pvG7xgkJ2jQ6
From (redirected): https://drive.google.com/uc?export=download&id=1qgMZdOpZ_KJAgqc46WW5pvG7xgkJ2jQ6&confirm=t&uuid=99858
To: /content/restaurants_all.csv
100% 190M/190M [00:03<00:00, 49.4MB/s]
```