Amit Divekar | Assignment 1: Set B

DataFrame Operations and Merging - Set B

In [1]:

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

Q1. Consider a following record in DataFrame IPL

In [2]:

```python
data = {
    'player': ['Hardik Pandya', 'K L Rahul', 'Andre Russel', 'Jasprit Bumrah
    'team': ['Mumbai Indians', 'Kings Eleven', 'Kolkata Knight Riders', 'Mum
    'category': ['Batsman', 'Batsman', 'Batsman', 'Bowler', 'Batsman', 'Bats
    'bidprice': [13, 12, 7, 10, 17, 15],
    'runs': [1000, 2400, 900, 200, 3600, 3700]
}

IPL = pd.DataFrame(data)

print("Original DataFrame")
print(IPL)
```

```
Original DataFrame
          player                   team category  bidprice  runs
0  Hardik Pandya         Mumbai Indians  Batsman        13  1000
1      K L Rahul           Kings Eleven  Batsman        12  2400
2   Andre Russel  Kolkata Knight Riders  Batsman         7   900
3 Jasprit Bumrah         Mumbai Indians   Bowler        10   200
4    Virat Kohli                    RCB  Batsman        17  3600
5   Rohit Sharma         Mumbai Indians  Batsman        15  3700
```

a) Retrieve first 2 rows

In [3]:

```python
print("\na) First 2 rows")
print(IPL.head(2))
```

```
a) First 2 rows
          player            team category  bidprice  runs
0  Hardik Pandya  Mumbai Indians  Batsman        13  1000
1      K L Rahul    Kings Eleven  Batsman        12  2400
```

b) Retrieve last 3 rows

In [4]:

```python
print("\nb) Last 3 rows")
print(IPL.tail(3))
```

```
b) Last 3 rows
           player            team category  bidprice  runs
3  Jasprit Bumrah  Mumbai Indians   Bowler        10   200
4     Virat Kohli             RCB  Batsman        17  3600
5    Rohit Sharma  Mumbai Indians  Batsman        15  3700
```

c) Add null values in DataFrame

In [5]:

```
IPL.loc[2, 'runs'] = np.nan
IPL.loc[4, 'bidprice'] = np.nan

print("\nc) After adding null values")
print(IPL)
```

```
c) After adding null values
           player                   team category  bidprice    runs
0   Hardik Pandya         Mumbai Indians  Batsman      13.0  1000.0
1       K L Rahul           Kings Eleven  Batsman      12.0  2400.0
2    Andre Russel  Kolkata Knight Riders  Batsman       7.0     NaN
3  Jasprit Bumrah         Mumbai Indians   Bowler      10.0   200.0
4     Virat Kohli                    RCB  Batsman       NaN  3600.0
5    Rohit Sharma         Mumbai Indians  Batsman      15.0  3700.0
```

d) Find most expensive player

In [6]:

```
print("\nd) Most expensive player")
print(IPL.loc[IPL['bidprice'].idxmax()])
```

```
d) Most expensive player
player        Rohit Sharma
team        Mumbai Indians
category           Batsman
bidprice              15.0
runs                3700.0
Name: 5, dtype: object
```

e) Print total players per team

In [7]:

```
print("\ne) Total players per team")
print(IPL['team'].value_counts())
```

```
e) Total players per team
team
Mumbai Indians           3
Kings Eleven             1
Kolkata Knight Riders    1
RCB                      1
Name: count, dtype: int64
```

f) Find average runs of each player

In [8]:

```
print("\nf) Average runs of each player")
print(IPL.groupby('player')['runs'].mean())
```

```
f) Average runs of each player
player
Andre Russel         NaN
Hardik Pandya     1000.0
Jasprit Bumrah     200.0
K L Rahul         2400.0
Rohit Sharma      3700.0
Virat Kohli       3600.0
Name: runs, dtype: float64
```

g) Drop rows with missing data

In [9]:

```
IPL_clean = IPL.dropna()

print("\ng) After dropping missing values")
print(IPL_clean)
```

```
g) After dropping missing values
           player            team category  bidprice    runs
0   Hardik Pandya  Mumbai Indians  Batsman      13.0  1000.0
1       K L Rahul    Kings Eleven  Batsman      12.0  2400.0
3  Jasprit Bumrah  Mumbai Indians   Bowler      10.0   200.0
5    Rohit Sharma  Mumbai Indians  Batsman      15.0  3700.0
```

Q2. Create a following DataFrame named as "data"

In [10]:

```
data = pd.DataFrame(
    {
        'company': ['Apsara', 'Natraj', 'Cello', 'Parkar', 'Apsara'],
        'count': [15, 20, 25, 35, 20],
        'price': [250, 200, 600, 900, 300]
    },
    index=['Pencil', 'Pencil', 'Pen', 'Pen', 'Eraser']
)

print("Original DataFrame")
print(data)
```

```
Original DataFrame
       company  count  price
Pencil  Apsara     15    250
Pencil  Natraj     20    200
Pen      Cello     25    600
Pen     Parkar     35    900
Eraser  Apsara     20    300
```

a) Find all rows with the label "Pencil". Extract all columns

In [11]:

```
print("\na) Rows with label Pencil")
print(data.loc['Pencil'])
```

```
a) Rows with label Pencil
       company  count  price
Pencil  Apsara     15    250
Pencil  Natraj     20    200
```

b) Change the Eraser count as 25 instead of 20

In [12]:

```
data.loc['Eraser', 'count'] = 25

print("\nb) After changing eraser count")
print(data)
```

```
b) After changing eraser count
       company  count  price
Pencil  Apsara     15    250
Pencil  Natraj     20    200
Pen      Cello     25    600
Pen     Parkar     35    900
Eraser  Apsara     25    300
```

c) List only the columns Company and Price

In [13]:

```
print("\nc) Company and Price columns")
print(data[['company', 'price']])
```

```
c) Company and Price columns
       company  price
Pencil  Apsara    250
Pencil  Natraj    200
Pen      Cello    600
Pen     Parkar    900
Eraser  Apsara    300
```

d) List only rows with labels 'Pencil' and 'Pen'

In [14]:

```
print("\nd) Rows with Pencil and Pen")
print(data.loc[['Pencil', 'Pen']])
```

```
d) Rows with Pencil and Pen
       company  count  price
Pencil  Apsara     15    250
Pencil  Natraj     20    200
Pen      Cello     25    600
Pen     Parkar     35    900
```

e) Delete column Count from the above DataFrame

In [15]:

```
data_new = data.drop(columns=['count'])

print("\ne) After deleting Count column")
print(data_new)
```

```
e) After deleting Count column
       company  price
Pencil  Apsara    250
Pencil  Natraj    200
Pen      Cello    600
Pen     Parkar    900
Eraser  Apsara    300
```

Q3. Write a python program to join the two DataFrames with matching records from both sides where available.

In [16]:

```
student_data1 = pd.DataFrame({
    'Id': ['S2', 'S3', 'S4', 'S5', 'S5'],
    'Name': ['Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin', 'Kw
    'Marks': [210, 190, 222, 199, 199]
})

student_data2 = pd.DataFrame({
    'Id': ['S4', 'S5', 'S6', 'S7', 'S8'],
    'Name': ['Scarlette Fisher', 'Carla Williamson', 'Dante Morse', 'Kaiser
    'Marks': [201, 200, 198, 219, 201]
})

print("Student Data 1")
print(student_data1)

print("\nStudent Data 2")
print(student_data2)
```

```
Student Data 1
   Id          Name  Marks
0  S2  Ryder Storey    210
1  S3  Bryce Jensen    190
2  S4     Ed Bernal    222
3  S5   Kwame Morin    199
4  S5   Kwame Morin    199

Student Data 2
   Id              Name  Marks
0  S4  Scarlette Fisher    201
1  S5  Carla Williamson    200
2  S6       Dante Morse    198
3  S7    Kaiser William    219
4  S8   Madeeha Preston    201
```

In [17]:

```
result = pd.merge(student_data1, student_data2, on='Id', how='inner')

print("\nMatching records from both DataFrames")
print(result)
```

```
Matching records from both DataFrames
   Id       Name_x  Marks_x            Name_y  Marks_y
0  S4    Ed Bernal      222  Scarlette Fisher      201
1  S5  Kwame Morin      199  Carla Williamson      200
2  S5  Kwame Morin      199  Carla Williamson      200
```