ract-transform-and-load-processes

March 20, 2025

[8]:
```python
# Importing required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as pl
#input data
data={
'ID': [1, 2, 3, 4, 5, 6],
'Name': ['Poornima', 'Siya', 'Preeti', 'Aryan', 'Purva', 'Shreya'],
'Age': [25, 30, 35, 40, 22, 29],
'Country': ['India', 'UK', 'USA', 'Australia', 'Canada', 'Russia'],
'Sales': [200, 450, 300, 800, 150, 400]
}
#create dataframe
df = pd.DataFrame(data)
print("Original Dataset:")
print(df)
```

```
Original Dataset:
   ID      Name  Age    Country  Sales
0   1  Poornima   25      India    200
1   2      Siya   30         UK    450
2   3    Preeti   35        USA    300
3   4     Aryan   40  Australia    800
4   5     Purva   22     Canada    150
5   6    Shreya   29     Russia    400
```

[9]:
```python
#Data Transforamtions
#character map
#description: Transform text data by changing the case of characters.
#Here, we will convert the Name column to uppercase.
df['Name_Upper']=df['Name'].str.upper()
print("\nCharacter Map (Uppercase Names):")
print(df[['ID','Name','Name_Upper']])
```

```

Character Map (Uppercase Names):
   ID      Name Name_Upper
0   1  Poornima   POORNIMA
```

```
1   2      Siya       SIYA
2   3    Preeti     PREETI
3   4     Aryan      ARYAN
4   5     Purva      PURVA
5   6    Shreya     SHREYA
```

[10]:
```
#multicast: create two copies of the dataset
df_copy1 = df.copy()
df_copy2 = df.copy()
#transformations on each copy
df_copy1['Sales'] *= 1.1 #increase sales by 10%
df_copy2['Age'] += 5
print("\nMulticast (Modified copies):")
print("Copy 1 (Sales Increased):")
print(df_copy1)
print("\nCopy 2 (Age Increased):")
print(df_copy2)
```

```

Multicast (Modified copies):
Copy 1 (Sales Increased):
   ID      Name  Age    Country  Sales Name_Upper
0   1  Poornima   25      India  220.0   POORNIMA
1   2      Siya   30         UK  495.0       SIYA
2   3    Preeti   35        USA  330.0     PREETI
3   4     Aryan   40  Australia  880.0      ARYAN
4   5     Purva   22     Canada  165.0      PURVA
5   6    Shreya   29     Russia  440.0     SHREYA

Copy 2 (Age Increased):
   ID      Name  Age    Country  Sales Name_Upper
0   1  Poornima   30      India    200   POORNIMA
1   2      Siya   35         UK    450       SIYA
2   3    Preeti   40        USA    300     PREETI
3   4     Aryan   45  Australia    800      ARYAN
4   5     Purva   27     Canada    150      PURVA
5   6    Shreya   34     Russia    400     SHREYA
```

[11]:
```
#3. Conditional split
#description: split data based on a condition.
#Here, we will separate rows with Sales > 300.
#Conditional split: Sales > 300
high_sales = df[df['Sales'] > 300]
low_sales = df[df['Sales'] <= 300]
print("\nHigh sales :")
print(high_sales)
print("\nLow sales :")
print(low_sales)
```

```
High sales :
   ID    Name  Age    Country  Sales Name_Upper
1   2    Siya   30         UK    450       SIYA
3   4   Aryan   40  Australia    800      ARYAN
5   6  Shreya   29     Russia    400     SHREYA

Low sales :
   ID      Name  Age Country  Sales Name_Upper
0   1  Poornima   25   India    200   POORNIMA
2   3    Preeti   35     USA    300     PREETI
4   5     Purva   22  Canada    150      PURVA
```

[12]:
```
#4.Aggreagation
#Description: Aggreagate data, e.g.,calculate Total Sales by Country
agg_df = df.groupby('Country')['Sales'].sum().reset_index()
print("\nAggregation (Total Sales by Country):")
print(agg_df)
```

```
Aggregation (Total Sales by Country):
     Country  Sales
0  Australia    800
1     Canada    150
2      India    200
3     Russia    400
4         UK    450
5        USA    300
```

[13]:
```
#5. Sort
#Description: Sort the dataset by Sales in descending order.
#Sort: Sort by Sales in descending order
sorted_df = df.sort_values(by='Sales', ascending=False)
print("\nSort (Descending Sales):")
print(sorted_df)
```

```
Sort (Descending Sales):
   ID      Name  Age    Country  Sales Name_Upper
3   4     Aryan   40  Australia    800      ARYAN
1   2      Siya   30         UK    450       SIYA
5   6    Shreya   29     Russia    400     SHREYA
2   3    Preeti   35        USA    300     PREETI
0   1  Poornima   25      India    200   POORNIMA
4   5     Purva   22     Canada    150      PURVA
```

[14]:
```
#6.Derived Column: Categorize sales as 'High' or 'Low'.
#create a new column by deriving information from existing data.
#Derived Column: Categorize sales as 'High' or 'Low'.
df['Sales_Category'] = df['Sales'].apply(lambda x: 'High' if x > 300 else'Low')
print("\nDerived Column (Sales Category):")
print(df[['ID','Name','Sales','Sales_Category']])
```

```
Derived Column (Sales Category):
   ID      Name  Sales Sales_Category
0   1  Poornima    200            Low
1   2      Siya    450           High
2   3    Preeti    300            Low
3   4     Aryan    800           High
4   5     Purva    150            Low
5   6    Shreya    400           High
```