Data Cleaning

```
import pandas as pd
import numpy as np

# Sample dataset with missing values and duplicates
data1 = pd.DataFrame({
    'ID': [1, 2, 3, 4, 4],
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'David'],
    'Age': [25, 30, np.nan, 40, 40],
    'City': ['New York', 'Los Angeles', 'Chicago', 'Houston', 'Houston']
})

print("Original Data:")
print(data1)

# Remove duplicate rows
data1 = data1.drop_duplicates()

# Fill missing values in 'Age' with the column mean
data1['Age'].fillna(data1['Age'].mean(), inplace=True)

print("\nCleaned Data:")
print(data1)
```

```
Original Data:
   ID     Name   Age         City
0   1    Alice  25.0     New York
1   2      Bob  30.0  Los Angeles
2   3  Charlie   NaN      Chicago
3   4    David  40.0      Houston
4   4    David  40.0      Houston

Cleaned Data:
   ID     Name        Age         City
0   1    Alice  25.000000     New York
1   2      Bob  30.000000  Los Angeles
2   3  Charlie  31.666667      Chicago
3   4    David  40.000000      Houston
<ipython-input-1-8f580b932d8f>:19: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained as
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

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


  data1['Age'].fillna(data1['Age'].mean(), inplace=True)
```

Data Integration

```
# Second dataset to merge
data2 = pd.DataFrame({
    'ID': [1, 2, 3, 5],
    'Income': [50000, 60000, 55000, 45000],
    'Gender': ['F', 'M', 'M', 'F']
})

# Merge datasets on 'ID'
merged_data = pd.merge(data1, data2, on='ID', how='left')

print("Merged Data:")
print(merged_data)
```

```
Merged Data:
   ID     Name        Age         City   Income Gender
0   1    Alice  25.000000     New York  50000.0      F
1   2      Bob  30.000000  Los Angeles  60000.0      M
2   3  Charlie  31.666667      Chicago  55000.0      M
3   4    David  40.000000      Houston      NaN    NaN
```

Data Transformation

```
# Encode categorical variable 'Gender' using one-hot encoding
merged_data = pd.get_dummies(merged_data, columns=['Gender'], drop_first=True)

# Normalize 'Income' column using Min-Max scaling
merged_data['Income_Normalized'] = (
    (merged_data['Income'] - merged_data['Income'].min()) /
    (merged_data['Income'].max() - merged_data['Income'].min())
)

print("Transformed Data:")
print(merged_data)
```

```
Transformed Data:
   ID     Name        Age         City   Income  Gender_M  Income_Normalized
0   1    Alice  25.000000     New York  50000.0     False                0.0
1   2      Bob  30.000000  Los Angeles  60000.0      True                1.0
2   3  Charlie  31.666667      Chicago  55000.0      True                0.5
3   4    David  40.000000      Houston      NaN     False                NaN
```

Data Reduction

```
# Drop columns that are not useful for analysis
reduced_data = merged_data.drop(columns=['City', 'Income'])

print("Reduced Final Data:")
print(reduced_data)
```

```
Reduced Final Data:
   ID     Name        Age  Gender_M  Income_Normalized
0   1    Alice  25.000000     False                0.0
1   2      Bob  30.000000      True                1.0
2   3  Charlie  31.666667      True                0.5
3   4    David  40.000000     False                NaN
```

Upload the Excel File to Google Colab

```
from google.colab import files
uploaded = files.upload()
```

Choose Files student_data.xlsx

- **student_data.xlsx**(application/vnd.openxmlformats-officedocument.spreadsheetml.sheet) - 13912 bytes, last modified: 4/6/2025 - 100% done

```
import pandas as pd

# Load the Excel file into a DataFrame
df = pd.read_excel("student_data.xlsx")

# Display the data
print(df)
```

```
     StudentID      Name   Age Gender Grade                            Email  \
0         1000     Jerry   NaN  Other     C                  zhill@gmail.com
1         1001     Erica   NaN    NaN     D                 yayala@walker.com
2         1002    Ashley   NaN      F     C                 brian89@gmail.com
3         1003   Michael   NaN  Other     A          hickmanrebecca@yahoo.com
4         1004    Eileen  20.0  Other     B  michaelbaker@fernandez-davis.net
..         ...       ...   ...    ...   ...                               ...
100       1022  Danielle  18.0      F   NaN          delacruzbarry@montes.net
101       1080      Jack   NaN    NaN   NaN          ywatkins@kirk-peters.com
102       1053    Amanda  24.0      F     D             juliebailey@yahoo.com
103       1061    Justin   NaN      F     F                bryanking@quinn.com
104       1072   Gregory   NaN      F     F      kimberlymcconnell@hotmail.com

            City EnrollmentDate
0    Los Angeles            NaT
1    Los Angeles     2024-03-15
2    Los Angeles     2025-03-20
3        Phoenix     2024-01-28
4       New York     2024-07-12
..           ...            ...
100          NaN            NaT
101          NaN     2023-11-26
102  Los Angeles            NaT
103      Chicago            NaT
104  Los Angeles            NaT

[105 rows x 8 columns]
```

```
df.head()
```

```
   StudentID     Name   Age  Gender  Grade                          Email          City  EnrollmentDate
0       1000    Jerry   NaN   Other      C                zhill@gmail.com   Los Angeles             NaT
1       1001    Erica   NaN     NaN      D              yayala@walker.com   Los Angeles      2024-03-15
2       1002   Ashley   NaN       F      C             brian89@gmail.com    Los Angeles      2025-03-20
3       1003  Michael   NaN   Other      A      hickmanrebecca@yahoo.com       Phoenix      2024-01-28
4       1004   Eileen  20.0   Other      B  michaelbaker@fernandez-davis.net   New York      2024-07-12
```

Next steps: Generate code with df | View recommended plots | New interactive sheet

Data Cleaning

```
# Make a copy of the dataset for cleaning
cleaned_df = df.copy()

# Remove duplicate rows
cleaned_df = cleaned_df.drop_duplicates()

# Fill missing values
cleaned_df['Age'] = cleaned_df['Age'].fillna(cleaned_df['Age'].mean())
cleaned_df['Gender'] = cleaned_df['Gender'].fillna(cleaned_df['Gender'].mode()[0])
cleaned_df['Grade'] = cleaned_df['Grade'].fillna(cleaned_df['Grade'].mode()[0])
cleaned_df['City'] = cleaned_df['City'].fillna(cleaned_df['City'].mode()[0])
cleaned_df['EnrollmentDate'] = cleaned_df['EnrollmentDate'].fillna(pd.Timestamp('2023-01-01'))

# View cleaned data
cleaned_df.head()
```

```
   StudentID     Name        Age  Gender  Grade                            Email         City  EnrollmentDate
0       1000    Jerry  21.860465   Other      C                  zhill@gmail.com  Los Angeles      2023-01-01
1       1001    Erica  21.860465       F      D                yayala@walker.com  Los Angeles      2024-03-15
2       1002   Ashley  21.860465       F      C               brian89@gmail.com  Los Angeles      2025-03-20
3       1003  Michael  21.860465   Other      A         hickmanrebecca@yahoo.com      Phoenix      2024-01-28
4       1004   Eileen  20.000000   Other      B  michaelbaker@fernandez-davis.net     New York      2024-07-12
```

Next steps: Generate code with cleaned_df | View recommended plots | New interactive sheet

Data Integration

```
# Create simulated additional data for merging
import random

additional_data = pd.DataFrame({
    'StudentID': cleaned_df['StudentID'].sample(frac=0.8).values,
    'Club': np.random.choice(['Science', 'Art', 'Sports', 'None'], size=int(0.8 * len(cleaned_df)))
})

# Merge on StudentID
merged_df = pd.merge(cleaned_df, additional_data, on='StudentID', how='left')

# View merged data
merged_df.head()
```

```
   StudentID     Name        Age  Gender  Grade                            Email         City  EnrollmentDate  Club
0       1000    Jerry  21.860465   Other      C                  zhill@gmail.com  Los Angeles      2023-01-01  None
1       1001    Erica  21.860465       F      D                yayala@walker.com  Los Angeles      2024-03-15   NaN
2       1002   Ashley  21.860465       F      C               brian89@gmail.com  Los Angeles      2025-03-20   NaN
3       1003  Michael  21.860465   Other      A         hickmanrebecca@yahoo.com      Phoenix      2024-01-28   NaN
4       1004   Eileen  20.000000   Other      B  michaelbaker@fernandez-davis.net     New York      2024-07-12   Art
```

Next steps: Generate code with merged_df | View recommended plots | New interactive sheet

Data Transformation

```
# Make a copy for transformation
transformed_df = merged_df.copy()

# One-hot encode categorical columns
transformed_df = pd.get_dummies(transformed_df, columns=['Gender', 'Grade', 'Club'], drop_first=True)

# Normalize Age (Min-Max Scaling)
transformed_df['Age_Normalized'] = (
    (transformed_df['Age'] - transformed_df['Age'].min()) /
    (transformed_df['Age'].max() - transformed_df['Age'].min())
)

# Convert EnrollmentDate to datetime
transformed_df['EnrollmentDate'] = pd.to_datetime(transformed_df['EnrollmentDate'])

# View transformed data
transformed_df.head()
```

	StudentID	Name	Age	Email	City	EnrollmentDate	Gender_M	Gender_Other	Grade_B	Grade_C	Grade_D
0	1000	Jerry	21.860465	zhill@gmail.com	Los Angeles	2023-01-01	False	True	False	True	False
1	1001	Erica	21.860465	yayala@walker.com	Los Angeles	2024-03-15	False	False	False	False	True
2	1002	Ashley	21.860465	brian89@gmail.com	Los Angeles	2025-03-20	False	False	False	True	False
3	1003	Michael	21.860465	hickmanrebecca@yahoo.com	Phoenix	2024-01-28	False	True	False	False	False
4	1004	Eileen	20.000000	michaelbaker@fernandez-davis.net	New York	2024-07-12	False	True	True	False	False

Next steps: Generate code with transformed_df View recommended plots New interactive sheet

Data Reduction

```
# Drop irrelevant or less useful columns
reduced_df = transformed_df.drop(columns=['Email', 'City', 'EnrollmentDate'])

# Final preprocessed dataset
reduced_df.head()
```

	StudentID	Name	Age	Gender_M	Gender_Other	Grade_B	Grade_C	Grade_D	Grade_F	Club_None	Club_Science	Club_Sports	Ag
0	1000	Jerry	21.860465	False	True	False	True	False	False	True	False	False	
1	1001	Erica	21.860465	False	False	False	False	True	False	False	False	False	
2	1002	Ashley	21.860465	False	False	False	True	False	False	False	False	False	
3	1003	Michael	21.860465	False	True	False	False	False	False	False	False	False	
4	1004	Eileen	20.000000	False	True	True	False	False	False	False	False	False	

Next steps: Generate code with reduced_df View recommended plots New interactive sheet