Lab – 26
Student Name: Navneet P
Student ID: AF0411619

Topic: Pandas IO and Data Cleaning

Data cleaning: Our data often comes from multiple resources and is not clean. It may contain missing values, duplicates, wrong or undesired formats, etc. Therefore, it is necessary to prepare your data before it is fed to your model. This preparation of the data by identifying and resolving the potential errors, inaccuracies etc.

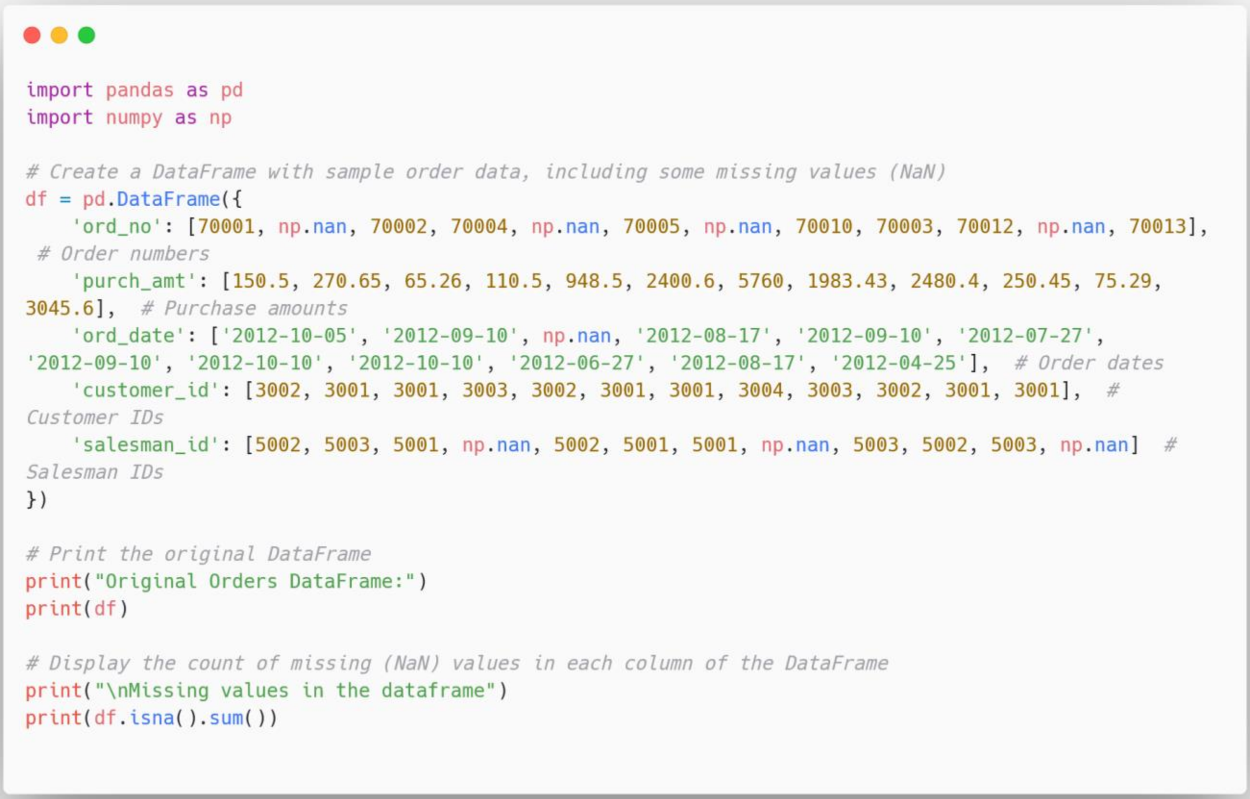
Functions used in this assignment:

- ✓ **numpy.nan**: numpy.nan represents "Not a Number" (NaN) in NumPy. It is used to represent missing or undefined numerical data in arrays. Operations involving nan generally result in nan, making it useful for handling missing data.

- ✓ **isna()**: isna() is a Pandas function that detects missing values (NaN) in a DataFrame or Series. It returns a DataFrame/Series of the same shape, with True for missing values and False for non-missing values. It helps in identifying where data is missing.

- ✓ **dropna()**: dropna() is a Pandas function that removes rows or columns with missing values (NaN) from a DataFrame or Series. By default, it drops any row that contains a NaN, but it can be customized to target specific columns or rows with missing data. It is useful for cleaning datasets before analysis.

Q1. Count the number of missing values in each column and display them.
Solution:



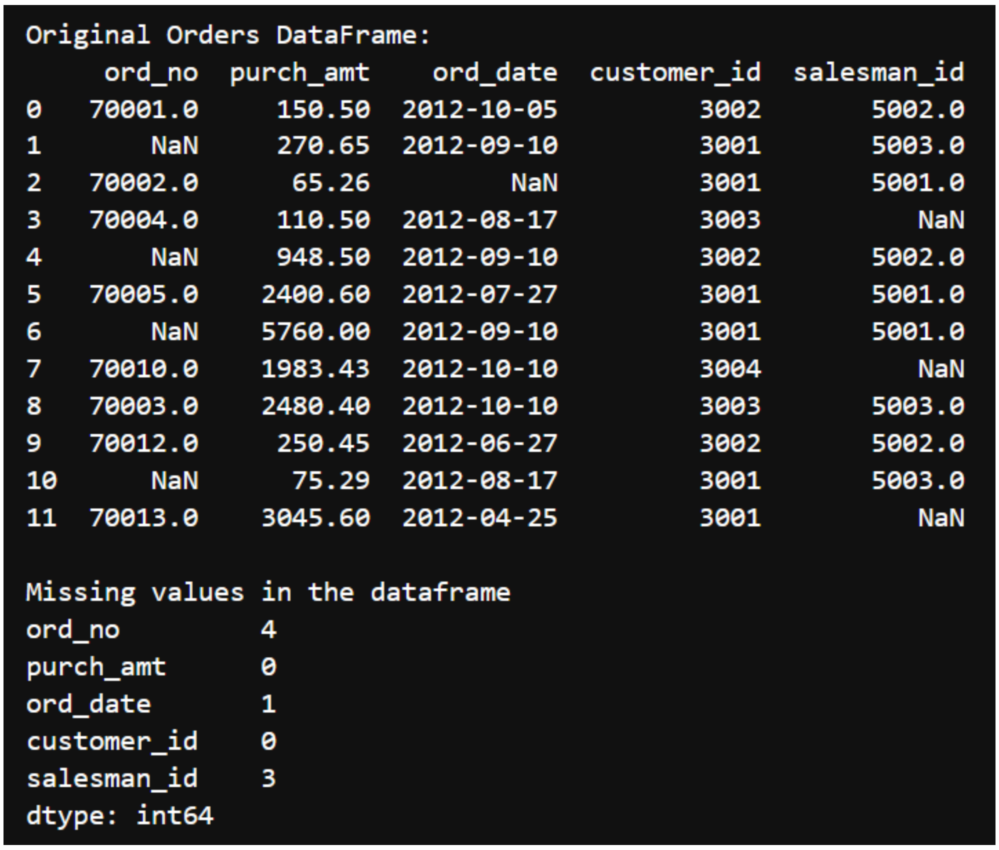
```python
import pandas as pd
import numpy as np

# Create a DataFrame with sample order data, including some missing values (NaN)
df = pd.DataFrame({
    'ord_no': [70001, np.nan, 70002, 70004, np.nan, 70005, np.nan, 70010, 70003, 70012, np.nan, 70013],
 # Order numbers
    'purch_amt': [150.5, 270.65, 65.26, 110.5, 948.5, 2400.6, 5760, 1983.43, 2480.4, 250.45, 75.29,
3045.6],  # Purchase amounts
    'ord_date': ['2012-10-05', '2012-09-10', np.nan, '2012-08-17', '2012-09-10', '2012-07-27',
'2012-09-10', '2012-10-10', '2012-10-10', '2012-06-27', '2012-08-17', '2012-04-25'],  # Order dates
    'customer_id': [3002, 3001, 3001, 3003, 3002, 3001, 3001, 3004, 3003, 3002, 3001, 3001],  #
Customer IDs
    'salesman_id': [5002, 5003, 5001, np.nan, 5002, 5001, 5001, np.nan, 5003, 5002, 5003, np.nan]  #
Salesman IDs
})

# Print the original DataFrame
print("Original Orders DataFrame:")
print(df)

# Display the count of missing (NaN) values in each column of the DataFrame
print("\nMissing values in the dataframe")
print(df.isna().sum())
```

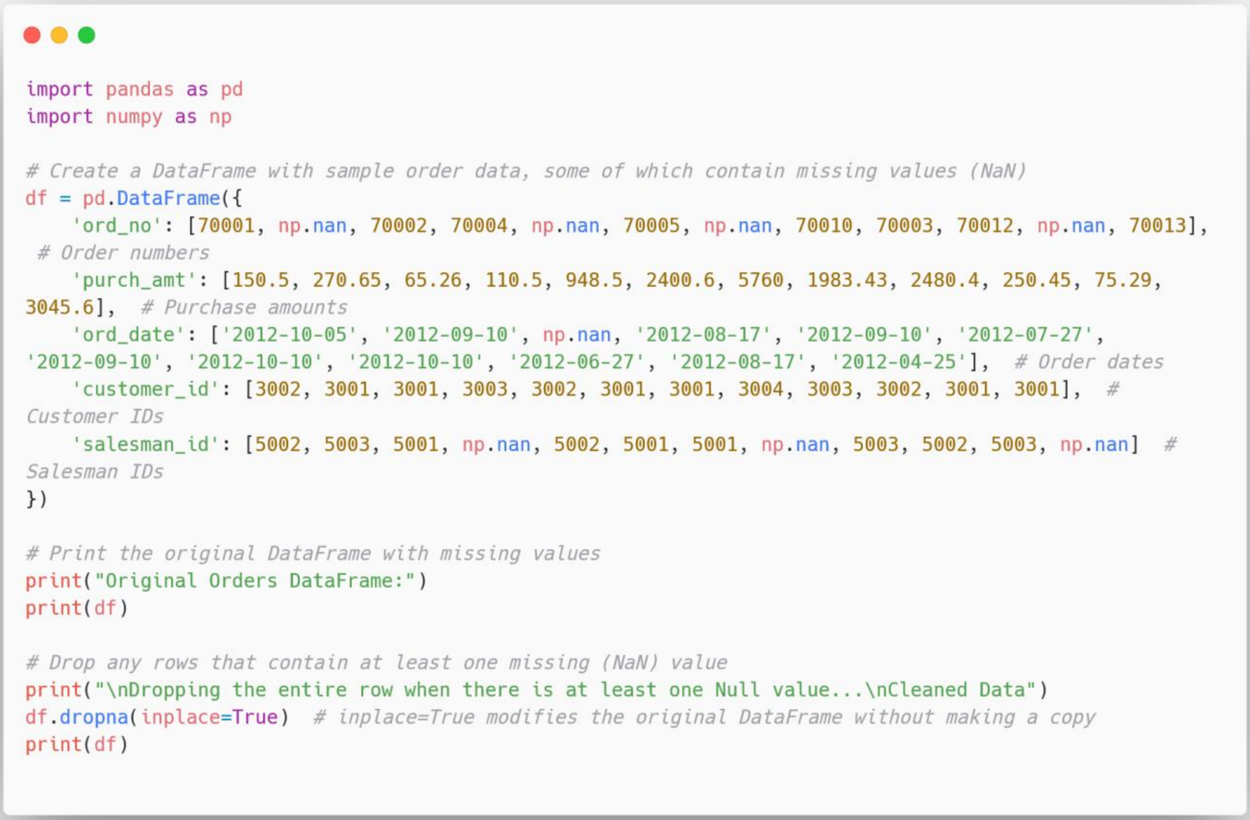
Output:



```
Original Orders DataFrame:
     ord_no  purch_amt    ord_date  customer_id  salesman_id
0   70001.0     150.50  2012-10-05         3002       5002.0
1       NaN     270.65  2012-09-10         3001       5003.0
2   70002.0      65.26         NaN         3001       5001.0
3   70004.0     110.50  2012-08-17         3003          NaN
4       NaN     948.50  2012-09-10         3002       5002.0
5   70005.0    2400.60  2012-07-27         3001       5001.0
6       NaN    5760.00  2012-09-10         3001       5001.0
7   70010.0    1983.43  2012-10-10         3004          NaN
8   70003.0    2480.40  2012-10-10         3003       5003.0
9   70012.0     250.45  2012-06-27         3002       5002.0
10      NaN      75.29  2012-08-17         3001       5003.0
11  70013.0    3045.60  2012-04-25         3001          NaN

Missing values in the dataframe
ord_no         4
purch_amt      0
ord_date       1
customer_id    0
salesman_id    3
dtype: int64
```

Q2. Drop those rows from the table which has at least one empty value.
Solution:



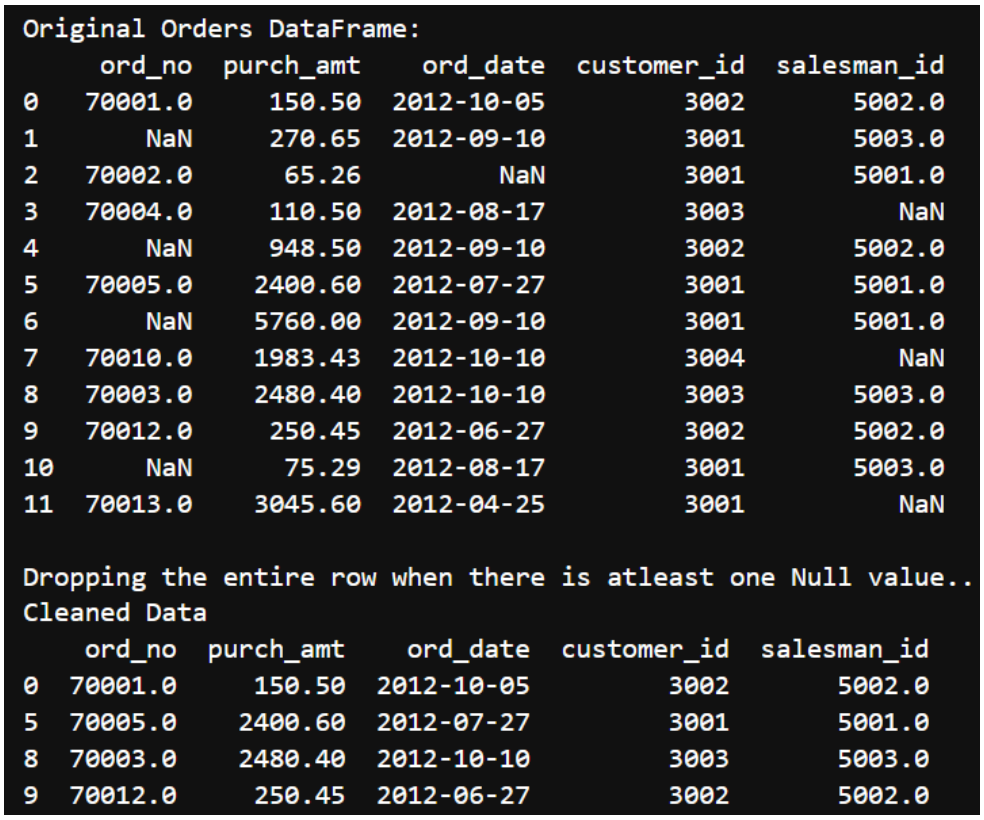
```
import pandas as pd
import numpy as np

# Create a DataFrame with sample order data, some of which contain missing values (NaN)
df = pd.DataFrame({
    'ord_no': [70001, np.nan, 70002, 70004, np.nan, 70005, np.nan, 70010, 70003, 70012, np.nan, 70013],
 # Order numbers
    'purch_amt': [150.5, 270.65, 65.26, 110.5, 948.5, 2400.6, 5760, 1983.43, 2480.4, 250.45, 75.29,
3045.6],  # Purchase amounts
    'ord_date': ['2012-10-05', '2012-09-10', np.nan, '2012-08-17', '2012-09-10', '2012-07-27',
'2012-09-10', '2012-10-10', '2012-10-10', '2012-06-27', '2012-08-17', '2012-04-25'],  # Order dates
    'customer_id': [3002, 3001, 3001, 3003, 3002, 3001, 3001, 3004, 3003, 3002, 3001, 3001],  #
Customer IDs
    'salesman_id': [5002, 5003, 5001, np.nan, 5002, 5001, 5001, np.nan, 5003, 5002, 5003, np.nan]  #
Salesman IDs
})

# Print the original DataFrame with missing values
print("Original Orders DataFrame:")
print(df)

# Drop any rows that contain at least one missing (NaN) value
print("\nDropping the entire row when there is at least one Null value...\nCleaned Data")
df.dropna(inplace=True)  # inplace=True modifies the original DataFrame without making a copy
print(df)
```

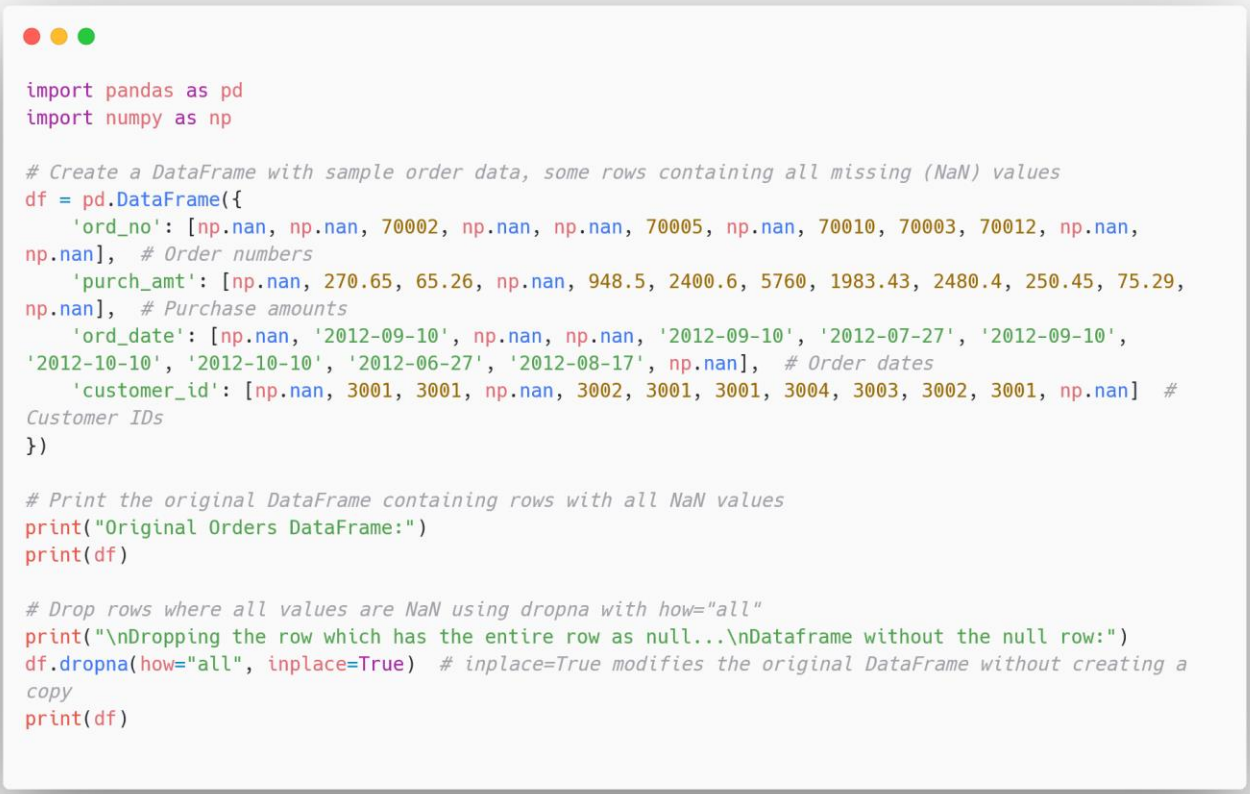
Output:



```
Original Orders DataFrame:
     ord_no  purch_amt    ord_date  customer_id  salesman_id
0   70001.0     150.50  2012-10-05         3002       5002.0
1       NaN     270.65  2012-09-10         3001       5003.0
2   70002.0      65.26         NaN         3001       5001.0
3   70004.0     110.50  2012-08-17         3003          NaN
4       NaN     948.50  2012-09-10         3002       5002.0
5   70005.0    2400.60  2012-07-27         3001       5001.0
6       NaN    5760.00  2012-09-10         3001       5001.0
7   70010.0    1983.43  2012-10-10         3004          NaN
8   70003.0    2480.40  2012-10-10         3003       5003.0
9   70012.0     250.45  2012-06-27         3002       5002.0
10      NaN      75.29  2012-08-17         3001       5003.0
11  70013.0    3045.60  2012-04-25         3001          NaN

Dropping the entire row when there is atleast one Null value..
Cleaned Data
    ord_no  purch_amt    ord_date  customer_id  salesman_id
0  70001.0     150.50  2012-10-05         3002       5002.0
5  70005.0    2400.60  2012-07-27         3001       5001.0
8  70003.0    2480.40  2012-10-10         3003       5003.0
9  70012.0     250.45  2012-06-27         3002       5002.0
```

Q3. Drop the rows where all elements of a certain rows are missing.
Solution:



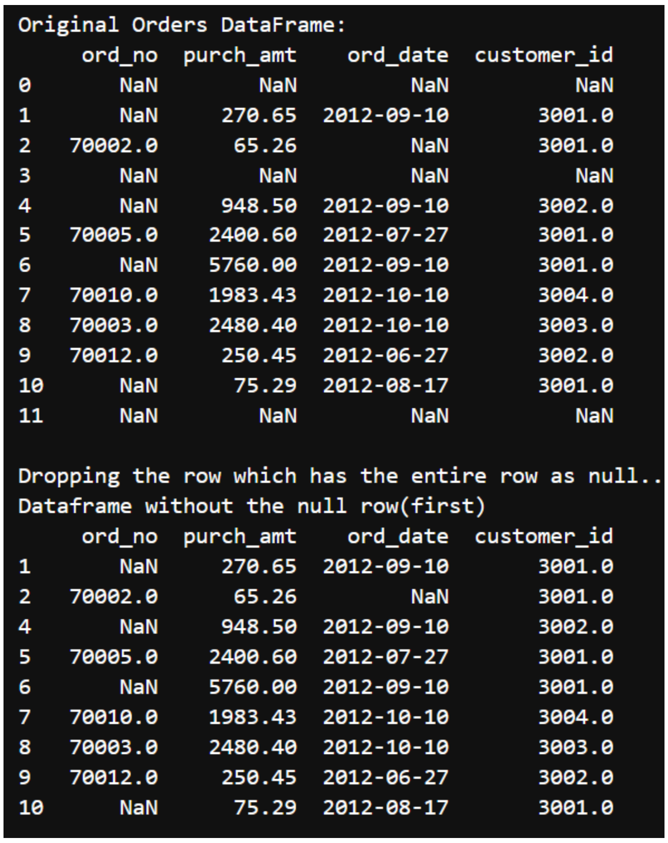
```python
import pandas as pd
import numpy as np

# Create a DataFrame with sample order data, some rows containing all missing (NaN) values
df = pd.DataFrame({
    'ord_no': [np.nan, np.nan, 70002, np.nan, np.nan, 70005, np.nan, 70010, 70003, 70012, np.nan,
np.nan],  # Order numbers
    'purch_amt': [np.nan, 270.65, 65.26, np.nan, 948.5, 2400.6, 5760, 1983.43, 2480.4, 250.45, 75.29,
np.nan],  # Purchase amounts
    'ord_date': [np.nan, '2012-09-10', np.nan, np.nan, '2012-09-10', '2012-07-27', '2012-09-10',
'2012-10-10', '2012-10-10', '2012-06-27', '2012-08-17', np.nan],  # Order dates
    'customer_id': [np.nan, 3001, 3001, np.nan, 3002, 3001, 3001, 3004, 3003, 3002, 3001, np.nan]  #
Customer IDs
})

# Print the original DataFrame containing rows with all NaN values
print("Original Orders DataFrame:")
print(df)

# Drop rows where all values are NaN using dropna with how="all"
print("\nDropping the row which has the entire row as null...\nDataframe without the null row:")
df.dropna(how="all", inplace=True)  # inplace=True modifies the original DataFrame without creating a
copy
print(df)
```

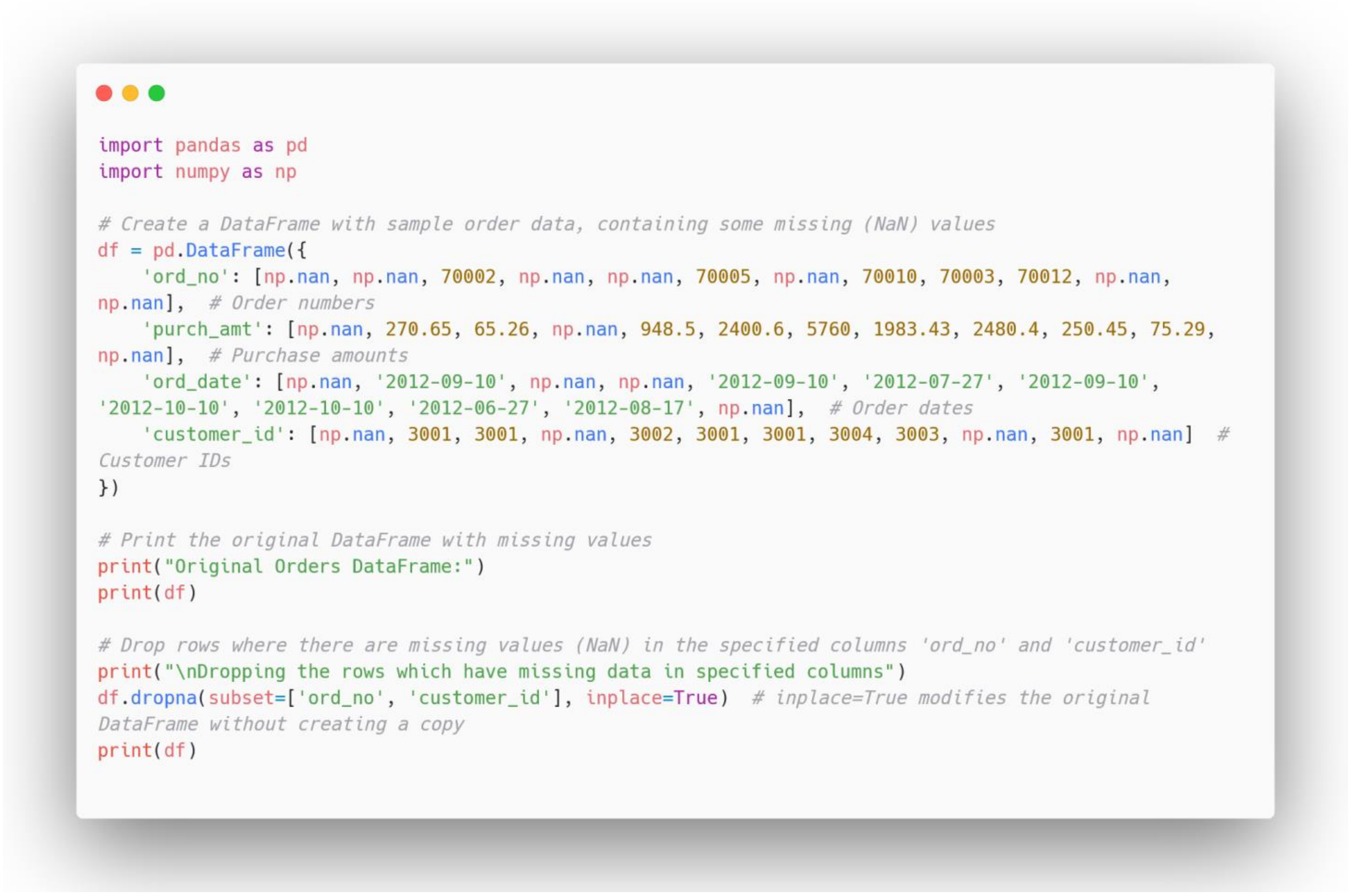
Output:



```
Original Orders DataFrame:
     ord_no  purch_amt    ord_date  customer_id
0       NaN        NaN         NaN          NaN
1       NaN     270.65  2012-09-10       3001.0
2   70002.0      65.26         NaN       3001.0
3       NaN        NaN         NaN          NaN
4       NaN     948.50  2012-09-10       3002.0
5   70005.0    2400.60  2012-07-27       3001.0
6       NaN    5760.00  2012-09-10       3001.0
7   70010.0    1983.43  2012-10-10       3004.0
8   70003.0    2480.40  2012-10-10       3003.0
9   70012.0     250.45  2012-06-27       3002.0
10      NaN      75.29  2012-08-17       3001.0
11      NaN        NaN         NaN          NaN

Dropping the row which has the entire row as null..
Dataframe without the null row(first)
     ord_no  purch_amt    ord_date  customer_id
1       NaN     270.65  2012-09-10       3001.0
2   70002.0      65.26         NaN       3001.0
4       NaN     948.50  2012-09-10       3002.0
5   70005.0    2400.60  2012-07-27       3001.0
6       NaN    5760.00  2012-09-10       3001.0
7   70010.0    1983.43  2012-10-10       3004.0
8   70003.0    2480.40  2012-10-10       3003.0
9   70012.0     250.45  2012-06-27       3002.0
10      NaN      75.29  2012-08-17       3001.0
```

Q4: Drop those rows which have null values based on specific columns.
Solution:



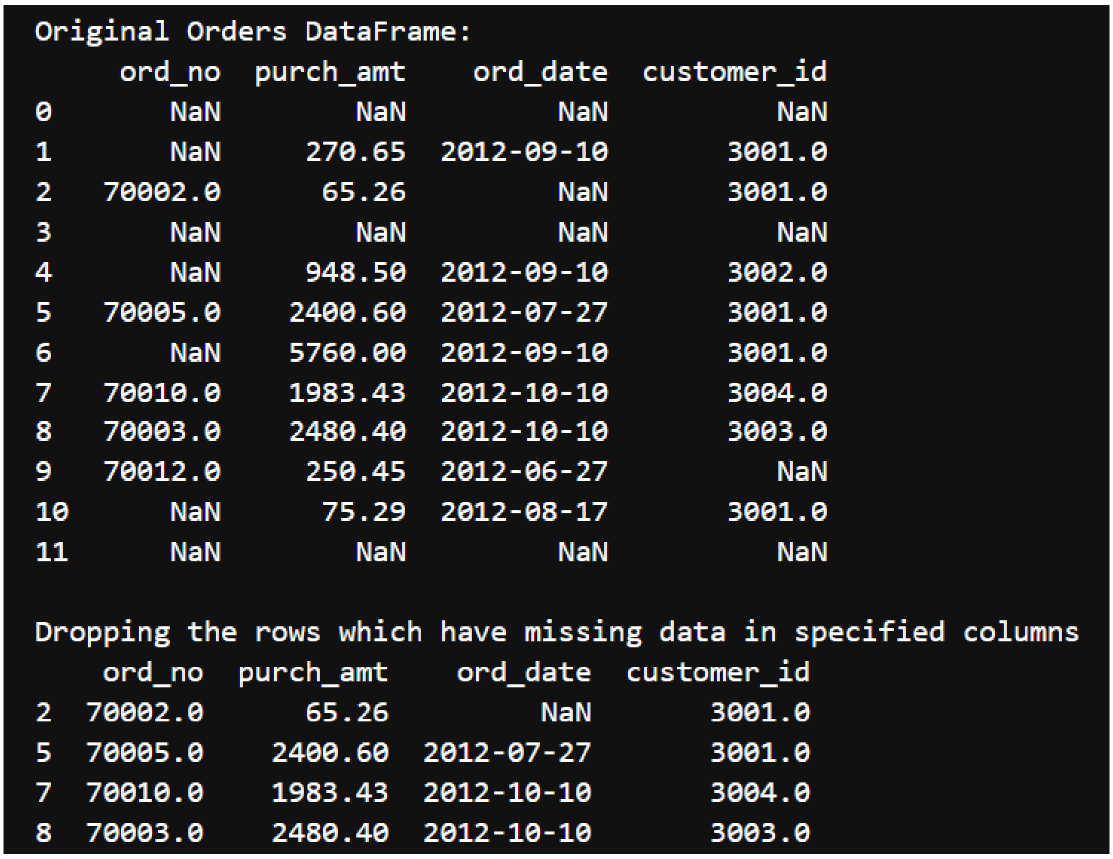
```python
import pandas as pd
import numpy as np

# Create a DataFrame with sample order data, containing some missing (NaN) values
df = pd.DataFrame({
    'ord_no': [np.nan, np.nan, 70002, np.nan, np.nan, 70005, np.nan, 70010, 70003, 70012, np.nan,
np.nan],  # Order numbers
    'purch_amt': [np.nan, 270.65, 65.26, np.nan, 948.5, 2400.6, 5760, 1983.43, 2480.4, 250.45, 75.29,
np.nan],  # Purchase amounts
    'ord_date': [np.nan, '2012-09-10', np.nan, np.nan, '2012-09-10', '2012-07-27', '2012-09-10',
'2012-10-10', '2012-10-10', '2012-06-27', '2012-08-17', np.nan],  # Order dates
    'customer_id': [np.nan, 3001, 3001, np.nan, 3002, 3001, 3001, 3004, 3003, np.nan, 3001, np.nan]  #
Customer IDs
})

# Print the original DataFrame with missing values
print("Original Orders DataFrame:")
print(df)

# Drop rows where there are missing values (NaN) in the specified columns 'ord_no' and 'customer_id'
print("\nDropping the rows which have missing data in specified columns")
df.dropna(subset=['ord_no', 'customer_id'], inplace=True)  # inplace=True modifies the original
DataFrame without creating a copy
print(df)
```

Output:



```
Original Orders DataFrame:
     ord_no  purch_amt    ord_date  customer_id
0       NaN        NaN         NaN          NaN
1       NaN     270.65  2012-09-10       3001.0
2   70002.0      65.26         NaN       3001.0
3       NaN        NaN         NaN          NaN
4       NaN     948.50  2012-09-10       3002.0
5   70005.0    2400.60  2012-07-27       3001.0
6       NaN    5760.00  2012-09-10       3001.0
7   70010.0    1983.43  2012-10-10       3004.0
8   70003.0    2480.40  2012-10-10       3003.0
9   70012.0     250.45  2012-06-27          NaN
10      NaN      75.29  2012-08-17       3001.0
11      NaN        NaN         NaN          NaN

Dropping the rows which have missing data in specified columns
    ord_no  purch_amt    ord_date  customer_id
2  70002.0      65.26         NaN       3001.0
5  70005.0    2400.60  2012-07-27       3001.0
7  70010.0    1983.43  2012-10-10       3004.0
8  70003.0    2480.40  2012-10-10       3003.0
```