PandasAssignment

Q1. How do you load a CSV file into a Pandas DataFrame?

A-1-

```
import pandas as pd
# Load the CSV file into a DataFrame
df = pd.read_csv('file.csv')
# View the first few rows of the DataFrame
print(df.head())
```

Q2. How do you check the data type of a column in a Pandas DataFrame?

A-2-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'column1': [1, 2, 3], 'column2': ['a', 'b', 'c']})
# Check the data type of column1
print(df['column1'].dtype)
# Check the data type of column2
print(df['column2'].dtype)
```

Q3. How do you select rows from a Pandas DataFrame based on a condition?

A-3-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'column1': [1, 2, 3], 'column2': ['a', 'b', 'c']})
# Select rows where column1 is greater than 1
condition = df['column1'] > 1
filtered_df = df[condition]
# View the filtered DataFrame
print(filtered_df)
```

Q4. How do you rename columns in a Pandas DataFrame?

A-4-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'column1': [1, 2, 3], 'column2': ['a', 'b', 'c']})
# Rename the columns
df = df.rename(columns={'column1': 'new_column1', 'column2': 'new_column2'})
# View the renamed DataFrame
print(df)
```

Q5. How do you drop columns in a Pandas DataFrame?

A-5-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'column1': [1, 2, 3], 'column2': ['a', 'b', 'c'], 'column3':
[4.5, 5.2, 6.1]})
# Drop column3
df = df.drop(columns=['column3'])
# View the modified DataFrame
print(df)
```

Q6. How do you find the unique values in a column of a Pandas DataFrame?

A-6-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'column1': [1, 2, 3, 2], 'column2': ['a', 'b', 'c', 'a']})
# Find the unique values in column1
unique_values = df['column1'].unique()
# View the unique values
print(unique_values)
```

Q7. How do you find the number of missing values in each column of a Pandas DataFrame?

A-7-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'column1': [1, 2, None, 4], 'column2': ['a', 'b', 'c', None],
'column3': [5.6, None, 7.1, 8.3]})
# Find the number of missing values in each column
missing_values = df.isnull().sum()
# View the number of missing values
print(missing_values)
```

Q8. How do you fill missing values in a Pandas DataFrame with a specific value?

A-8-

```
import pandas as pd
```

```
# Create a DataFrame with missing values
df = pd.DataFrame({'column1': [1, 2, None, 4], 'column2': ['a', 'b', 'c', None],
'column3': [5.6, None, 7.1, 8.3]})
# Fill missing values with a specific value (here, 0)
df = df.fillna(0)
# View the modified DataFrame
print(df)
```

Q9. How do you concatenate two Pandas DataFrames?

A-9-

```
import pandas as pd
# Create two DataFrames
df1 = pd.DataFrame({'column1': [1, 2, 3], 'column2': ['a', 'b', 'c']})
df2 = pd.DataFrame({'column1': [4, 5, 6], 'column2': ['d', 'e', 'f']})
# Concatenate the two DataFrames
df_concatenated = pd.concat([df1, df2])
# View the concatenated DataFrame
print(df_concatenated)
```

Q10. How do you merge two Pandas DataFrames on a specific column?

A-10-

```
import pandas as pd
# Create two DataFrames
df1 = pd.DataFrame({'column1': [1, 2, 3], 'column2': ['a', 'b', 'c'], 'column3':
['x', 'y', 'z']})
df2 = pd.DataFrame({'column1': [2, 3, 4], 'column4': ['d', 'e', 'f'], 'column5':
['p', 'q', 'r']})
# Merge the two DataFrames on 'column1'
df_merged = pd.merge(df1, df2, on='column1')
# View the merged DataFrame
print(df_merged)
```

Q11. How do you group data in a Pandas DataFrame by a specific column and apply an aggregation function?

A-11-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'category': ['A', 'B', 'A', 'B', 'A', 'B', 'A', 'A'],
                   'value': [1, 2, 3, 4, 5, 6, 7, 8]})
# Group the DataFrame by the 'category' column and compute the mean of the 'value'
```

```
column for each group
grouped_df = df.groupby('category').agg({'value': 'mean'})
# View the grouped DataFrame
print(grouped_df)
```

Q12. How do you pivot a Pandas DataFrame?

A-12-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'category': ['A', 'B', 'A', 'B', 'A', 'B', 'A', 'A'],
                   'value': [1, 2, 3, 4, 5, 6, 7, 8],
                   'date': ['2022-01-01', '2022-01-01', '2022-01-02', '2022-01-02',
                            '2022-01-01', '2022-01-01', '2022-01-02',
'2022-01-03']})
# Pivot the DataFrame
pivoted_df = df.pivot(index='date', columns='category', values='value')
# View the pivoted DataFrame
print(pivoted_df)
```

Q13. How do you change the data type of a column in a Pandas DataFrame?

A-13-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'name': ['Alice', 'Bob', 'Charlie'],'age': [25, 30,
35],'income': ['1000', '2000', '3000']})
# Convert the 'income' column from string to integer
df['income'] = df['income'].astype(int)
# View the updated DataFrame
print(df)
```

Q14. How do you sort a Pandas DataFrame by a specific column?

A-14-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'name': ['Alice', 'Bob', 'Charlie'],'age': [25, 30,
35],'income': [1000, 2000, 1500]})
# Sort the DataFrame by the 'income' column in ascending order
df_sorted = df.sort_values('income')
# View the sorted DataFrame
print(df_sorted)
```

Q15. How do you create a copy of a Pandas DataFrame?

A-15-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'name': ['Alice', 'Bob', 'Charlie'],'age': [25, 30,
35],'income': [1000, 2000, 3000]})
# Create a copy of the DataFrame
df_copy = df.copy()
# View the original and copied DataFrames
print(df)
print(df_copy)
```

Q16. How do you filter rows of a Pandas DataFrame by multiple conditions?

A-16-

```
import pandas as pd
# Create a DataFrame
df = pd.DataFrame({'name': ['Alice', 'Bob', 'Charlie'],'age': [25, 30,
35],'gender': ['F', 'M', 'M']})
# Filter the DataFrame by multiple conditions
df_filtered = df[(df['age'] > 25) & (df['gender'] == 'M')]
# View the filtered DataFrame
print(df_filtered)
```

Q17. How do you calculate the mean of a column in a Pandas DataFrame?

A-17-

```
import pandas as pd
# create a DataFrame
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
# calculate the mean of column 'B'
mean_b = df['B'].mean()
print(mean_b)
```

Q18. How do you calculate the standard deviation of a column in a Pandas DataFrame?

A-18-

```
import pandas as pd
# create a DataFrame
```

```
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
# calculate the standard deviation of column 'B'
std_b = df['B'].std()
print(std_b)
```

Q19. How do you calculate the correlation between two columns in a Pandas DataFrame?

A-19-

```
import pandas as pd
# create a DataFrame
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
# calculate the correlation between columns 'A' and 'B'
corr_a_b = df['A'].corr(df['B'])
print(corr_a_b)
```

Q20. How do you select specific columns in a DataFrame using their labels?

A-20-

```
import pandas as pd
# create a DataFrame
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6], 'C': [7, 8, 9]})
# select columns 'A' and 'C'
selected_cols = df[['A', 'C']]
print(selected_cols)
```

Q21. How do you select specific rows in a DataFrame using their indexes?

A-21-

```
import pandas as pd
# create a DataFrame
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6], 'C': [7, 8, 9]}, index=['x',
'y', 'z'])
# select rows with index 'y' and 'z'
selected_rows = df.loc[['y', 'z']]
print(selected_rows)
```

Q22. How do you sort a DataFrame by a specific column?

A-22-

```
import pandas as pd
# Create a DataFrame
```

```
df = pd.DataFrame({'name': ['Alice', 'Bob', 'Charlie'],'age': [25, 30,
35],'income': [1000, 2000, 1500]})
# Sort the DataFrame by the 'income' column in ascending order
df_sorted = df.sort_values('income')
# View the sorted DataFrame
print(df_sorted)
```

Q23. How do you create a new column in a DataFrame based on the values of another column?

A-23-

```
import pandas as pd
# create a DataFrame
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
# create a new column 'C' based on column 'A'
df['C'] = df['A'] * 2
print(df)
```

Q24. How do you remove duplicates from a DataFrame?

A-24-

```
import pandas as pd
# create a DataFrame with duplicates
df = pd.DataFrame({'A': [1, 2, 2], 'B': [4, 5, 5]})
# remove duplicates
df = df.drop_duplicates()
print(df)
```

Q25. What is the difference between .loc and .iloc in Pandas?

A-25-

.loc is used to select rows and columns by label, which means you use the row and column labels to select the data. It takes two arguments: the row label(s) and the column label(s).

Ex-

```
import pandas as pd
# create a DataFrame
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6], 'C': [7, 8, 9]}, index=['a',
'b', 'c'])
# select a subset of rows and columns using .loc
subset = df.loc[['a', 'c'], ['B', 'C']]
print(subset)
```

.iloc is used to select rows and columns by integer location,
which means you use the integer index to select the data.
It takes two arguments: the row index(es) and the column index(es).
Ex-

```
import pandas as pd
# create a DataFrame
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6], 'C': [7, 8, 9]}, index=['a',
'b', 'c'])
# select a subset of rows and columns using .iloc
subset = df.iloc[[0, 2], [1, 2]]
print(subset)
```