Answer

Ahmed Ashraf Mohamed

1 How can you create a DataFrame in Pandas from a Numpy array or a Python list?

Using `pandas.DataFrame(array,index,columns)` function where `index` and `columns` are optional arguments

```
import pandas as pd
a = [[1,2,3],[4,5,6],[7,8,9]]
df = pd.DataFrame(a)
```

2 How can you perform operations on DataFrame columns and rows in Pandas?

We can select any row or columns by passing the name of the row or column, which will be turned into a pandas series.

List of possible operations:

- `isnull()` : check for null value
- `notnull()` : check for non-null values
- `dropna()` : drops rows/columns that contain null values
- `fillna()` : replaces the null values with some other values
- `replace()` : replaces a string, regex, series, dictionary, etc.
- `lower()` : converts strings to lower case
- `upper()` : converts strings to upper case
- `mean()` : mean of values
- `sum()` : sum of values
- `count()` : counts number of non-NA elements
- `mean()` : mean of values
- `min()` : minimum
- `max()` : maximum

3 How can you sort a DataFrame in Pandas?

using method `DataFrame.sort_values(by)` which sorts by values along either axis

for example

```
df = pd.DataFrame({
    'col1': ['A', 'A', 'B', np.nan, 'D', 'C'],
    'col2': [2, 1, 9, 8, 7, 4],
    'col3': [0, 1, 9, 4, 2, 3],
    'col4': ['a', 'B', 'c', 'D', 'e', 'F']
})

df.sort_values(by=['col1'])
```

4 How can you drop missing values from a DataFrame in Pandas?

by using method `dropna()` to drop rows/columns that contain null values

```
df = pd.DataFrame({"name": ['Alfred', 'Batman', 'Catwoman'],
                   "toy": [np.nan, 'Batmobile', 'Bullwhip'],
                   "born": [pd.NaT, pd.Timestamp("1940-04-25"),
                            pd.NaT]})
df

df.dropna()
```

5 How can you perform groupby operations on a DataFrame in Pandas?

by calling `groupby()` method though a string passed to `groupby` may refer to either a column or a row, if the string matches both a column and a row it returns a `ValueError` error

for example

```
df = pd.DataFrame(
     {
        "A": ["foo", "bar", "foo", "bar", "foo", "bar", "foo", "foo"],
         "B": ["one", "one", "two", "three", "two", "two", "one", "three"],
        "C": np.random.randn(8),
        "D": np.random.randn(8),
     }
 )
```

```
df.groupby("A") # returns a group object

df.groupby("A")["C"].mean()
# returns a group object and performs mean method on the new object
```

6 How can you merge two DataFrames in Pandas based on a common column or index?

we can use both `join()` and `merge()` methods

`join()` method performs a left join, so each indexes in the first data frame are kept

`merge()` performs inner join, so only indexes that appear in both are kept

```
df1 = pd.DataFrame({'rating': [90, 85, 82, 88, 94, 90, 76, 75],
                   'points': [25, 20, 14, 16, 27, 20, 12, 15]},
                   index=["a","b","c","d","e","f","g","h"])

df2 = pd.DataFrame({'rating': [90, 85, 82, 88, 94, 90, 76, 75],
                   'points': [25, 20, 14, 16, 27, 20, 12, 15]},
                   index=["a","c","d","g","m","n"])
pd.merge(df1,df2)
```

7 How can you calculate the mean, median, and mode of a Pandas DataFrame column?

by using `mean()`, `median()`, and `mode()` methods after selecting the column

```
df1 = pd.DataFrame({'rating': [90, 85, 82, 88, 94, 90, 76, 75],
                   'points': [25, 20, 14, 16, 27, 20, 12, 15]},
                   index=["a","b","c","d","e","f","g","h"])
print(df1["ratings"].mean())
print(df1["ratings"].median())
print(df1["ratings"].mode())
```

8 What is the difference between iloc and loc in Pandas?

`loc` is label based selection, which means that it returns the row or column that matches the label, while `iloc` is integer based selection (from 0 to $length - 1$ along the the axis), which means that it returns the corresponding row or column (axis) based on the its number.