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

1. Load the CSV into a DataFrame: import pandas as pd. df = pd.read_csv('data.csv') ...
2. Print the DataFrame without the to_string() method: import pandas as pd. ...
3. Check the number of maximum returned rows: import pandas as pd. ...
4. Increase the maximum number of rows to display the entire DataFrame: import pandas as pd.

For eg:

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.to_string())
```

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

To check the data type in pandas DataFrame we can use the “dtype” attribute. The attribute returns a series with the data type of each column. And the column names of the DataFrame are represented as the index of the resultant series object and the corresponding data types are returned as values of the series object.

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

1. Create a Pandas DataFrame with data. ...
2. Selecting rows using loc[] ...
3. Select rows based on condition using loc. ...
4. Using 'loc' and '! ...
5. Combine multiple conditions with & operator. ...
6. Selected columns using loc. ...
7. Using loc[] and isin() ...
8. Selected column using loc[] and isin()

For eg:

```
# importing pandas

import pandas as pd

 record = {

'Name': ['Ankit', 'Amit', 'Aishwarya', 'Priyanka', 'Priya', 'Shaurya' ],

 'Age': [21, 19, 20, 18, 17, 21],

 'Stream': ['Math', 'Commerce', 'Science', 'Math', 'Math', 'Science'],

 'Percentage': [88, 92, 95, 70, 65, 78] }

 # create a dataframe
```

```
dataframe = pd.DataFrame(record, columns = ['Name', 'Age', 'Stream', 'Percentage'])

  print("Given Dataframe :\n", dataframe)

  # selecting rows based on condition

rslt_df = dataframe[dataframe['Percentage'] > 80]

  print('\nResult dataframe :\n', rslt_df)
```

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

One way of renaming the columns in a Pandas Dataframe is by using the **rename()** function.This method is quite useful when we need to rename some selected columns because we need to specify information only for the columns which are to be renamed.

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

The drop() method removes the specified row or column.

By specifying the column axis (axis='columns'), the drop() method removes the specified column.

By specifying the row axis (axis='index'), the drop() method removes the specified row.

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

You can get unique values in column (multiple columns) from pandas DataFrame **using unique() or Series**.

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

You can count the number of missing values in each row and column **by calling sum() from the result of isnull()**

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

The fillna() method replaces the NULL values with a specified value.

The fillna() method returns a new DataFrame object unless the inplace parameter is set to True,

in that case the fillna() method does the replacing in the original DataFrame instead.

For eg:

```
import pandas as pd

df = pd.read_csv('data.csv')

newdf = df.fillna(222222)
```

Q9. How do you concatenate two Pandas DataFrames?

The concat() function in pandas is used to append either columns or rows from one DataFrame to another. The concat() function does all the heavy lifting of performing concatenation operations along an axis while performing optional set logic (union or intersection) of the indexes (if any) on the other axes.

For eg:

```
import pandas as pd

# First DataFrame

df1 = pd.DataFrame({'id': ['A01', 'A02', 'A03', 'A04'],

                    'Name': ['ABC', 'PQR', 'DEF', 'GHI']})

# Second DataFrame

df2 = pd.DataFrame({'id': ['B05', 'B06', 'B07', 'B08'],

                    'Name': ['XYZ', 'TUV', 'MNO', 'JKL']})

frames = [df1, df2]

result = pd.concat(frames)

display(result)
```

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

You can pass two DataFrame to be merged to the pandas. merge() method. This collects all common columns in both DataFrames and replaces each common column in both DataFrame with a single one.

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

Pandas comes with a whole host of sql-like aggregation functions you can apply when grouping on one or more columns. This is Python's closest equivalent to dplyr's group_by + summarise logic. Here's a quick example of how to group on one or multiple columns and summarise data with aggregation functions using Pandas.

You can group by one column and count the values of another column per this column value using value_counts. Using groupby and value_counts we can count the number of activities each person did.

One aspect that I've recently been exploring is the task of grouping large data frames by different variables, and applying summary functions on each group. This is accomplished in Pandas using the "groupby()" and "agg()" functions of Panda's DataFrame objects.

Q12. How do you pivot a Pandas DataFrame?

The pivot() function is used to reshaped a given DataFrame organized by given index / column values. This function does not support data aggregation, multiple values will result in a MultiIndex in the columns.

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

You have four main options for converting types in pandas:

1)to_numeric() - provides functionality to safely convert non-numeric types (e.g. strings) to a suitable numeric type. (See also to_datetime() and to_timedelta().)

2)astype() - convert (almost) any type to (almost) any other type (even if it's not necessarily sensible to do so).

Also allows you to convert to categorial types (very useful).

3)infer_objects() - a utility method to convert object columns holding Python objects to a pandas type if possible.

4)convert_dtypes() - convert DataFrame columns to the "best possible" dtype that supports pd.NA (pandas' object to indicate a missing value).

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

DataFrame.sort_values(by, *, axis=0, ascending=True, inplace=False, kind='quicksort', na_position='last', ignore_index=False, key=None)

Parameters are by,axis,implace,kind etc.

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

DataFrame.copy(deep=True)[source]

Make a copy of this object's indices and data.

When deep=True (default), a new object will be created with a copy of the calling object's data and indices. Modifications to the data or indices of the copy will not be reflected in the original object (see notes below).

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

The loc function in pandas can be used to access groups of rows or columns by label. Add each condition you want to be included in the filtered result and concatenate them with the & operator. You'll see our code sample will return a pd. dataframe of our filtered rows.

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

To get column average or mean from pandas DataFrame use either mean() and describe() method. The DataFrame. mean() method is used to return the mean of the values for the requested axis.

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

Standard deviation is calculated using the function . std() . However, the Pandas library creates the Dataframe object and then the function . std() is applied on that Dataframe **.**

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

By using corr() function we can get the correlation between two columns in the dataframe

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

Use DataFrame. loc[] and DataFrame. iloc[] to select a single column or multiple columns from pandas DataFrame by column names/label or index position respectively. where loc[] is used with column labels/names and iloc[] is used with column index/position.

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

The axis labeling information in pandas objects serves many purposes:

Identifies data (i.e. provides metadata) using known indicators, important for analysis, visualization, and interactive console display.

Enables automatic and explicit data alignment.

Allows intuitive getting and setting of subsets of the data set.

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

To sort the DataFrame based on the values in a single column, you'll **use . sort_values()** . By default, this will return a new DataFrame sorted in ascending order. It does not modify the original DataFrame.

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

Using apply() method

If you need to apply a method over an existing column in order to compute some values that will eventually be added as a new column in the existing DataFrame, then pandas. DataFrame. apply() method should do the trick.

Q24. How do you remove duplicates from a DataFrame?

Pandas drop_duplicates() method helps in removing duplicates from the Pandas Dataframe In Python

Syntax of df. drop_duplicates()

subset: Subset takes a column or list of column label. ...

keep: keep is to control how to consider duplicate value.

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

The main distinction between the two methods is: loc gets rows (and/or columns) with particular labels. iloc gets rows (and/or columns) at integer locations.

The main difference between pandas loc[] vs iloc[] is loc gets DataFrame rows & columns by labels/names and iloc[] gets by integer Index/position. For loc[], if the label is not present it gives a key error. For iloc[], if the position is not present it gives an index error. In this article, I will cover the difference and similarities between loc[] and iloc[] in Pandas DataFrame by exploring with examples.