What is Pandas?

Pandas is a Python library used for working with data sets.

It has functions for analyzing, cleaning, exploring, and manipulating data.

The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

Why Use Pandas?

Pandas allows us to analyze big data and make conclusions based on statistical theories.

Pandas can clean messy data sets, and make them readable and relevant.

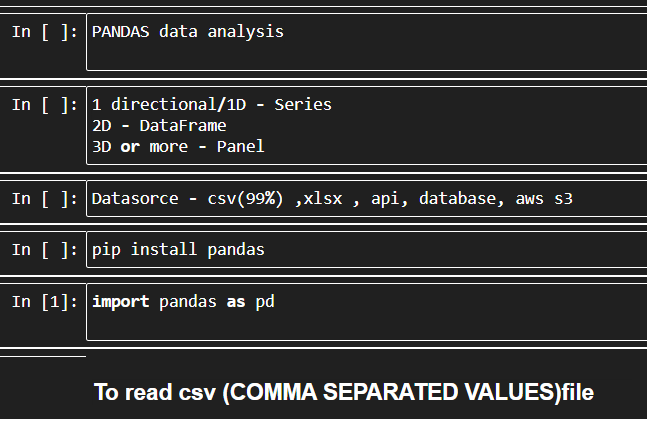
Relevant data is very important in data science.

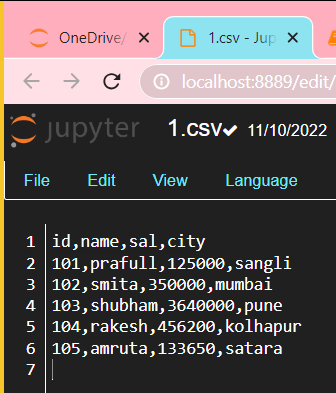
What Can Pandas Do?

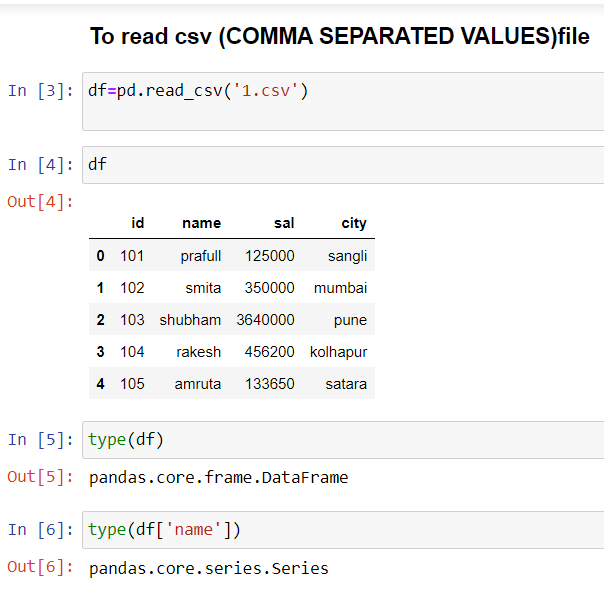
Pandas gives you answers about the data. Like:

* Is there a correlation between two or more columns?
* What is average value?
* Max value?
* Min value?

Pandas are also able to delete rows that are not relevant, or contains wrong values, like empty or NULL values. This is called *cleaning* the data.

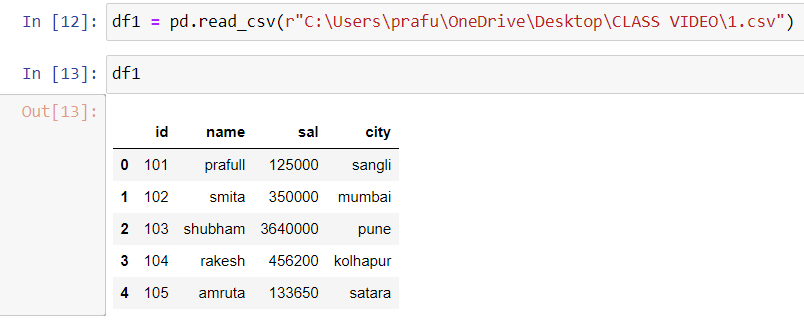


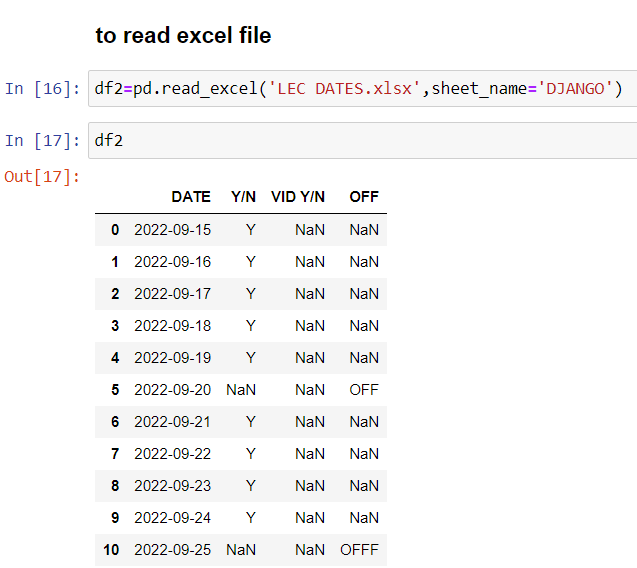


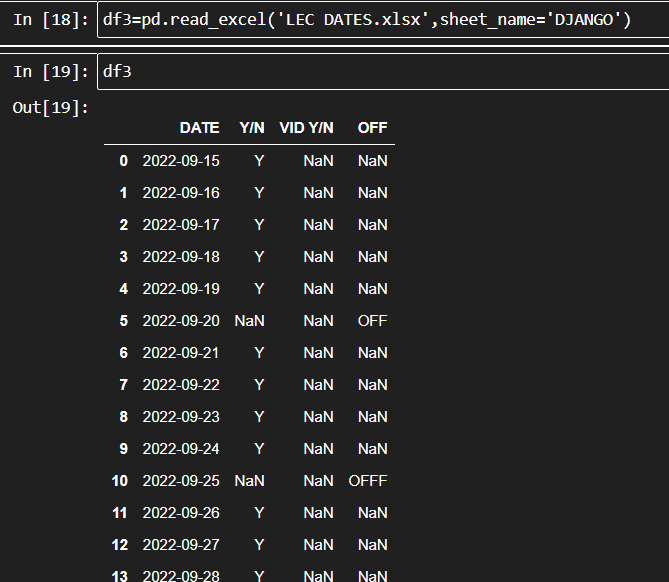


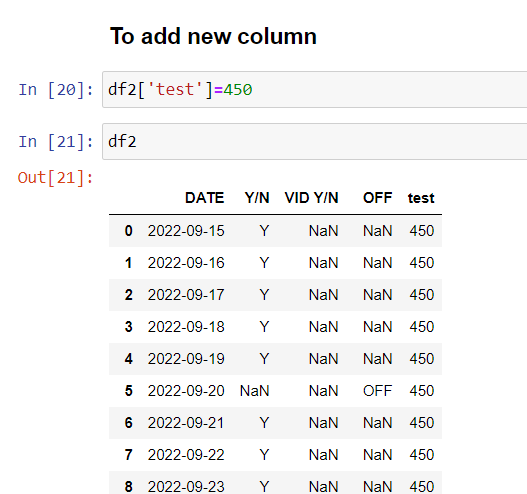
By using path of file location

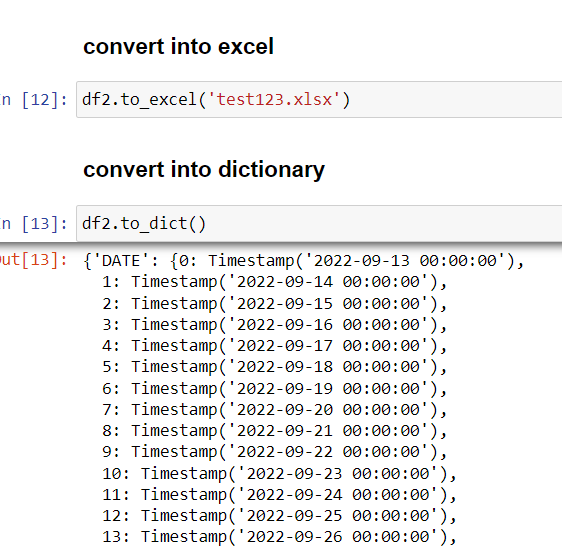
r “ “ :- r-string is very imp to fetch file using path.

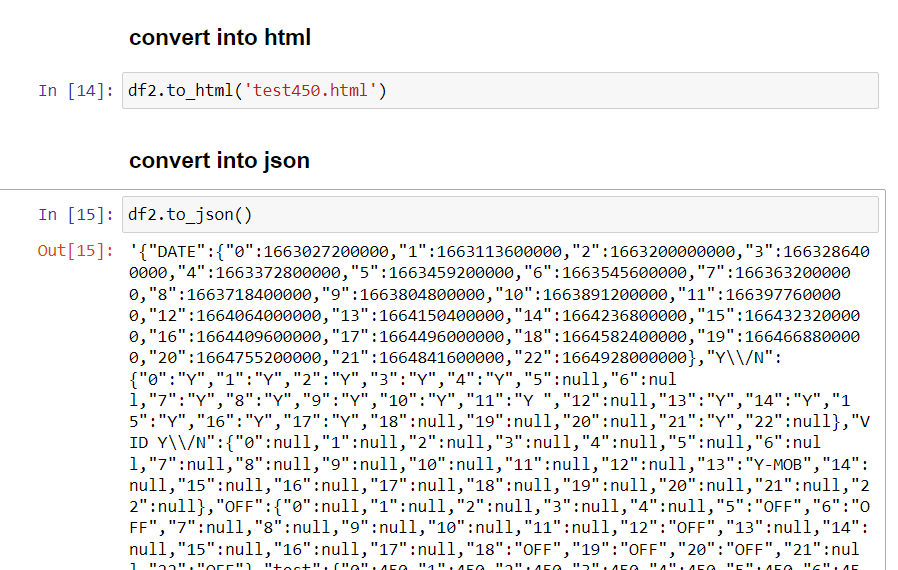


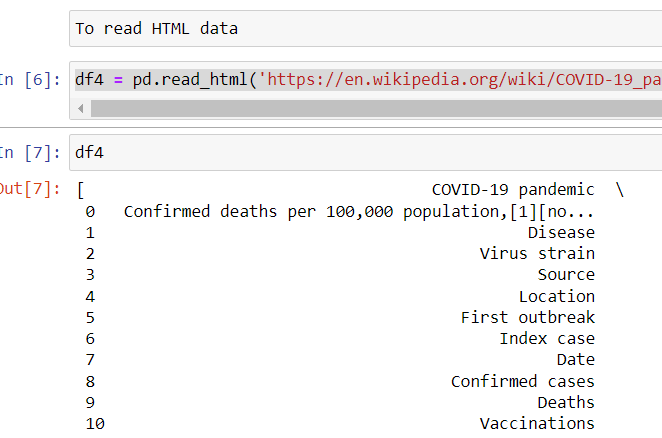








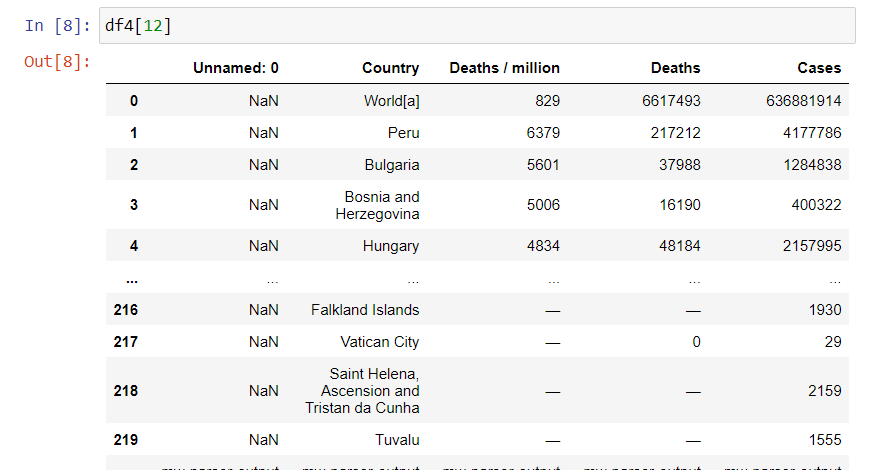
.

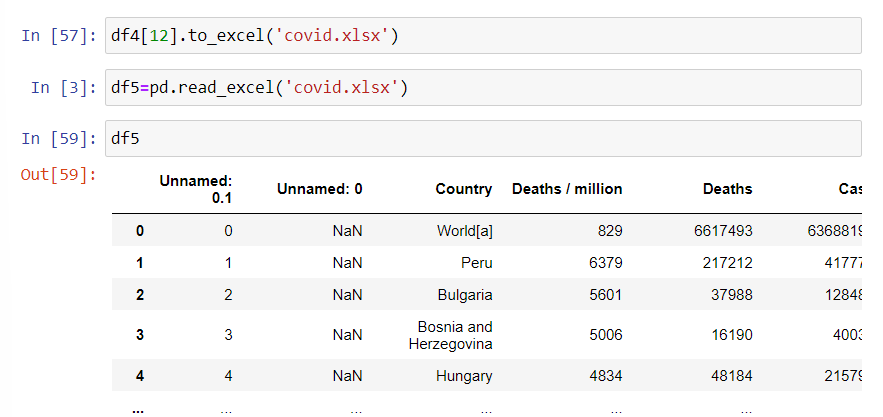


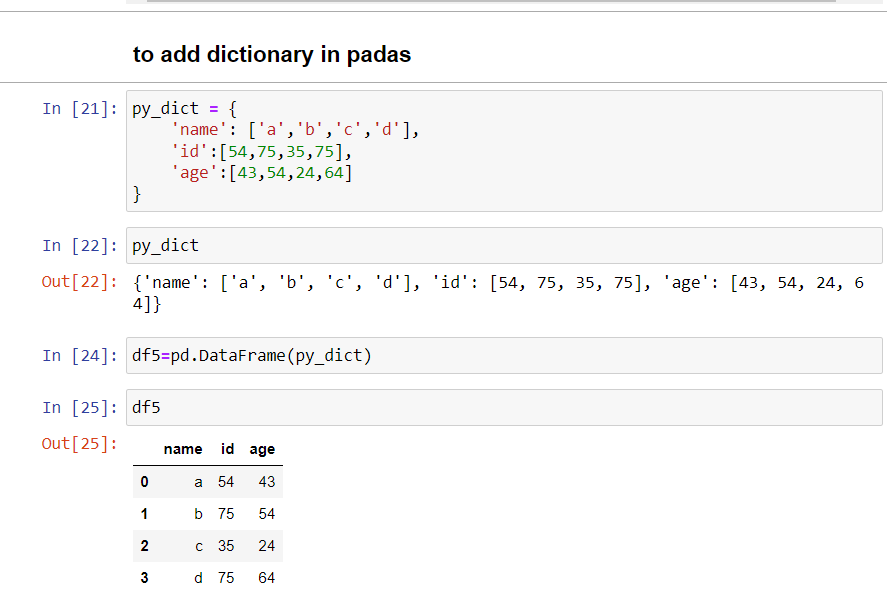
To read html data we use above method.

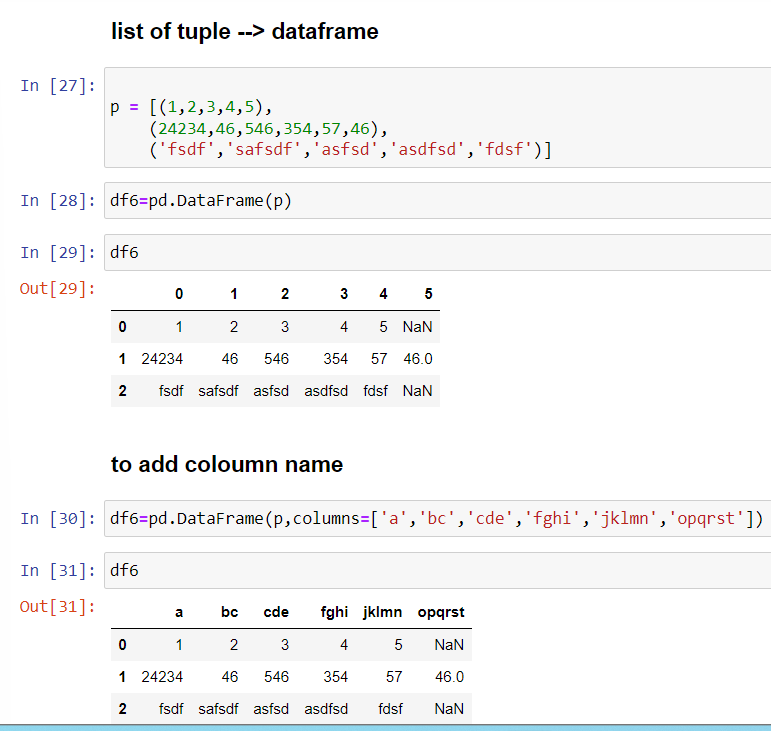
We get all the table of that html, so to fetch that tables, below

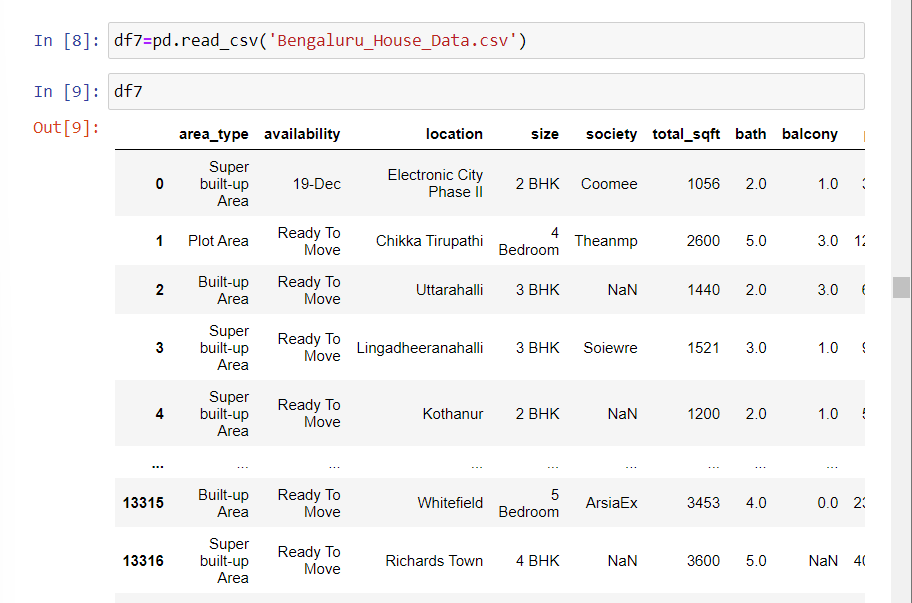
df [index no. of that table]

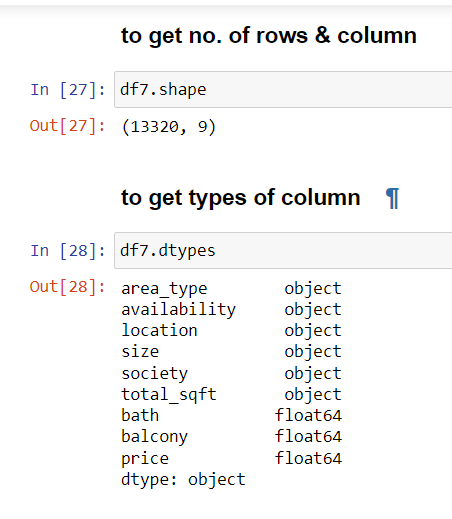


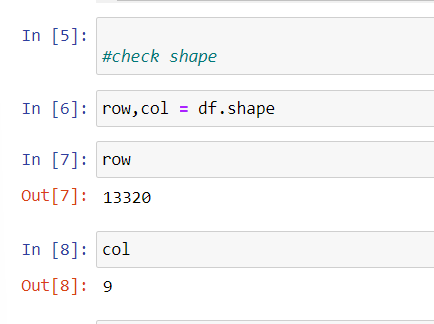




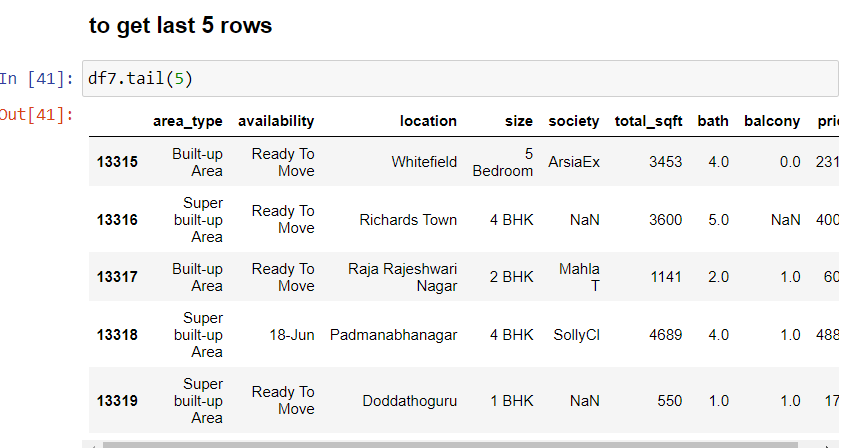


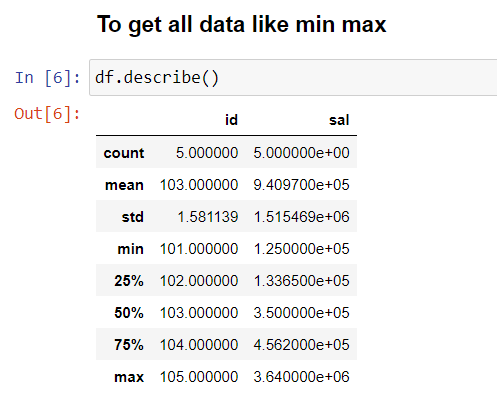


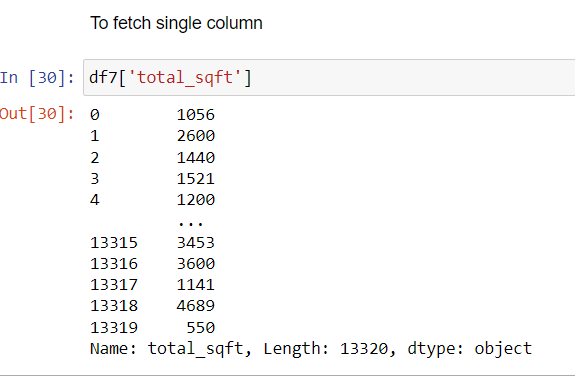


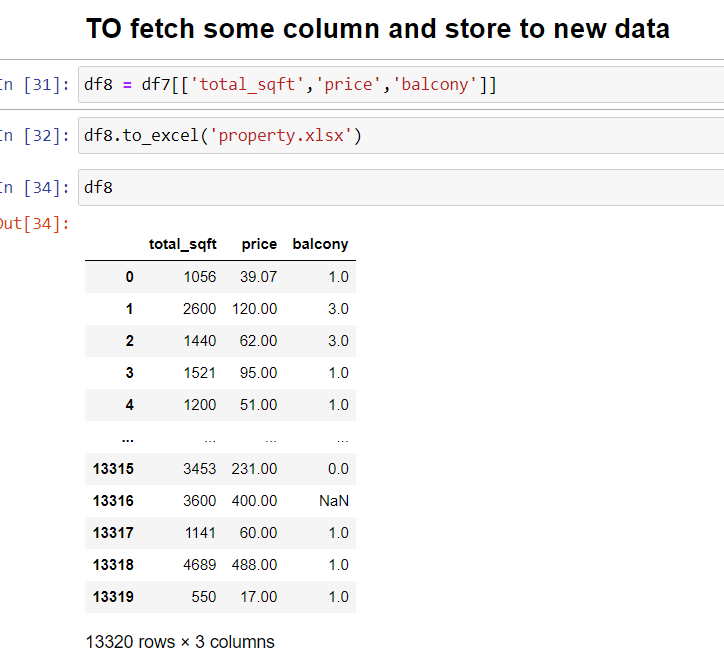


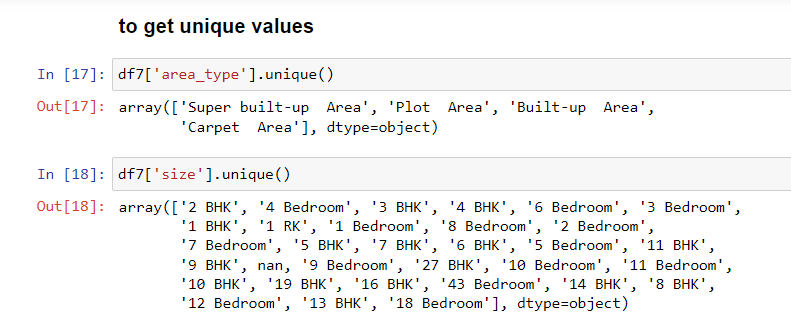


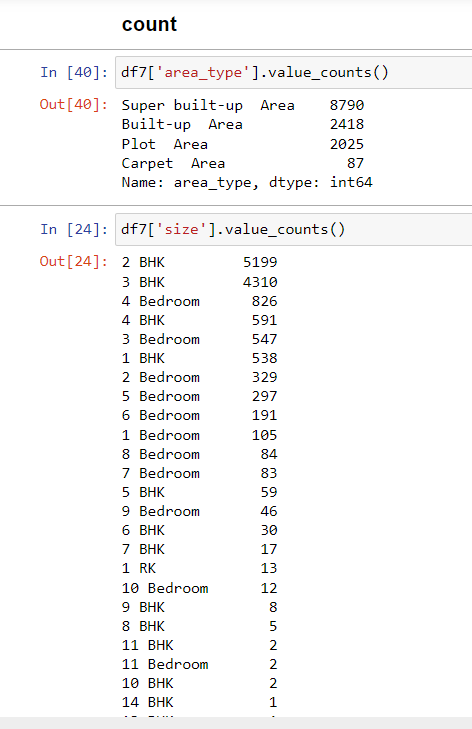


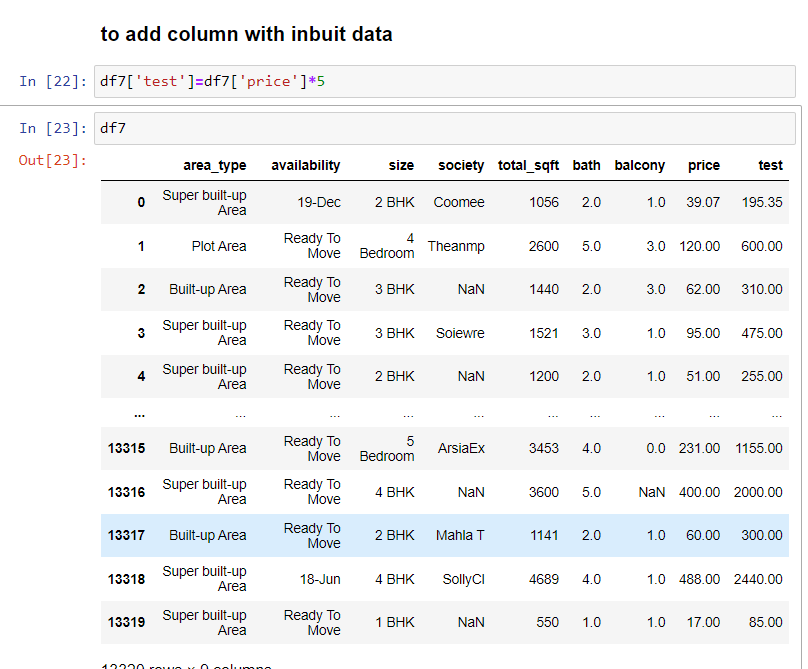
.

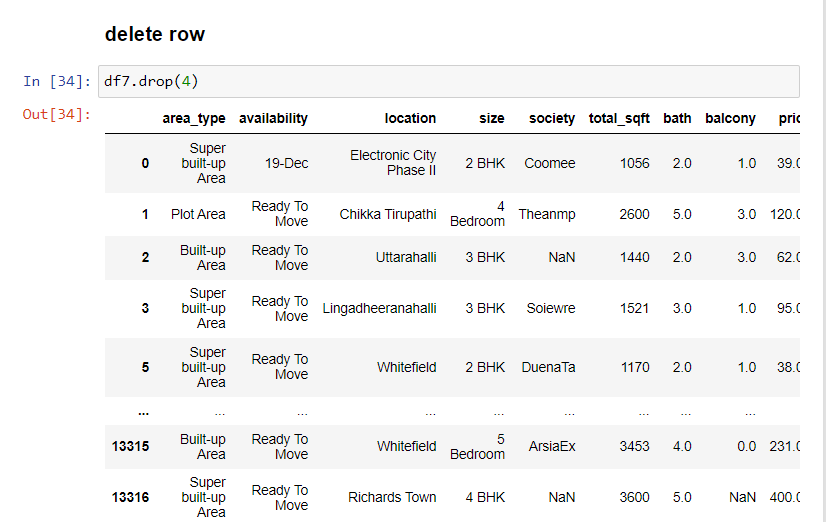


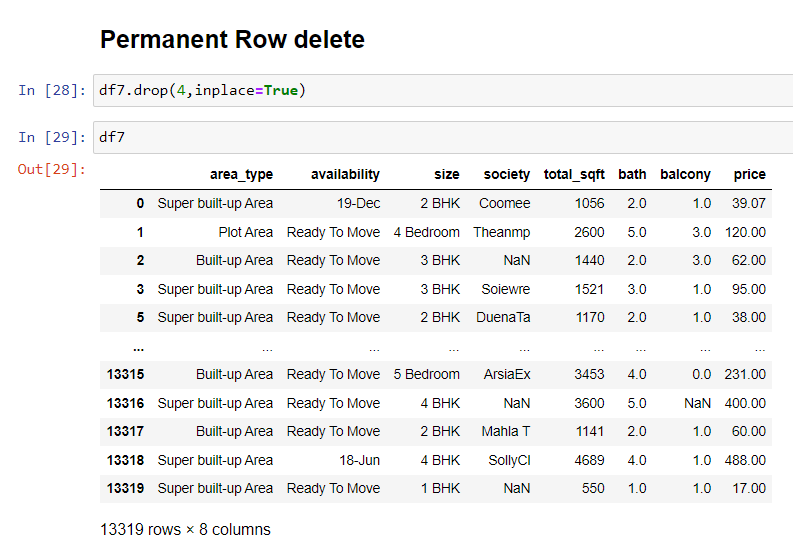






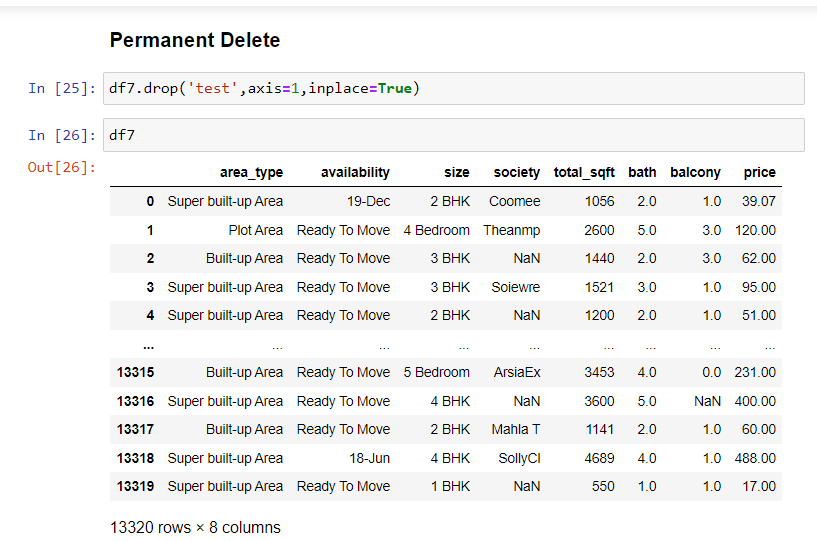


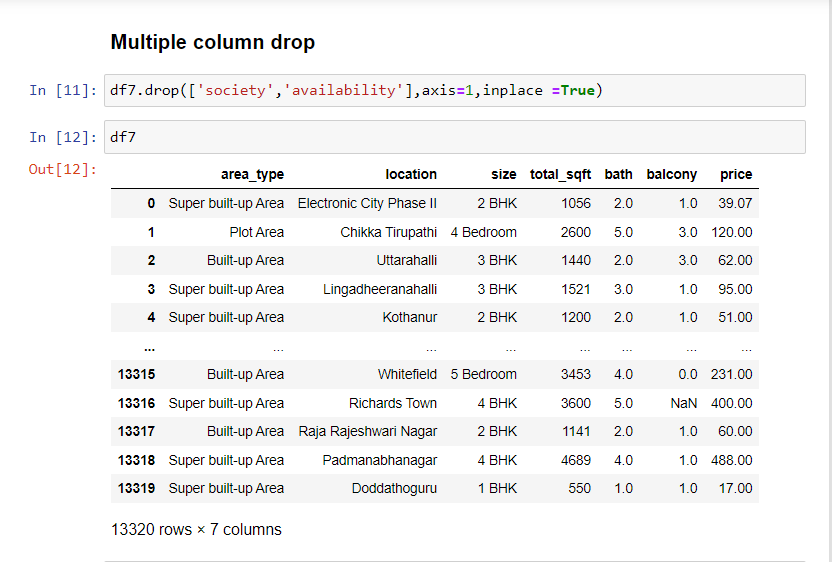


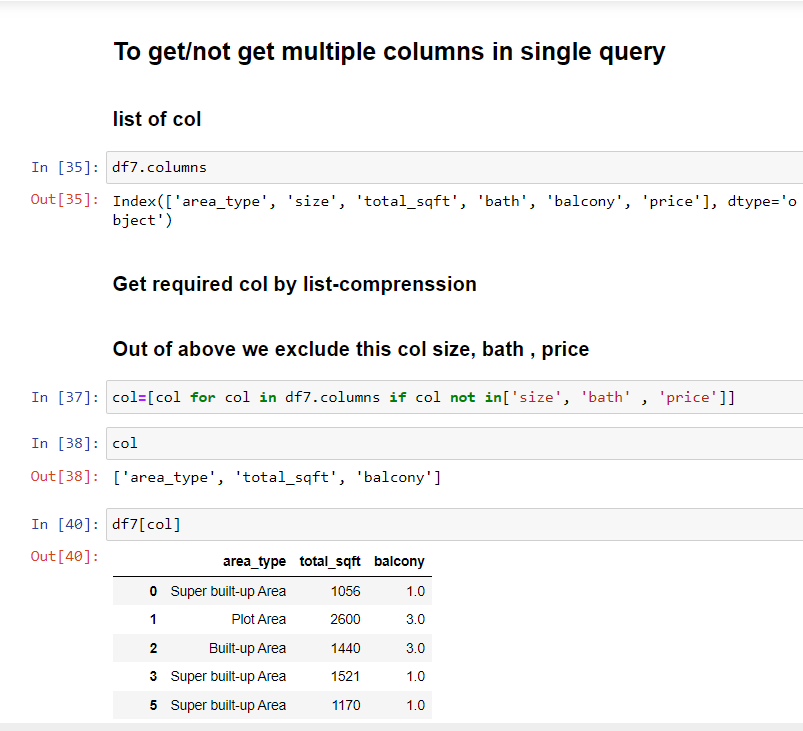


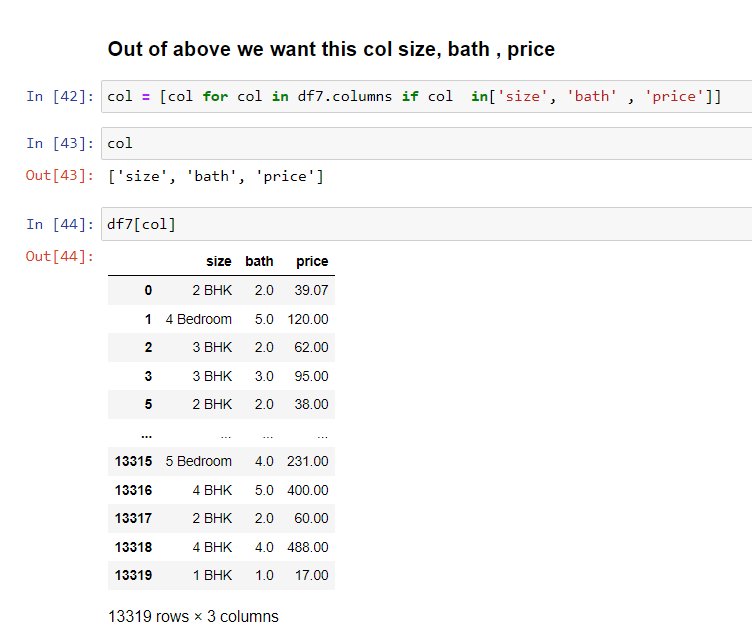


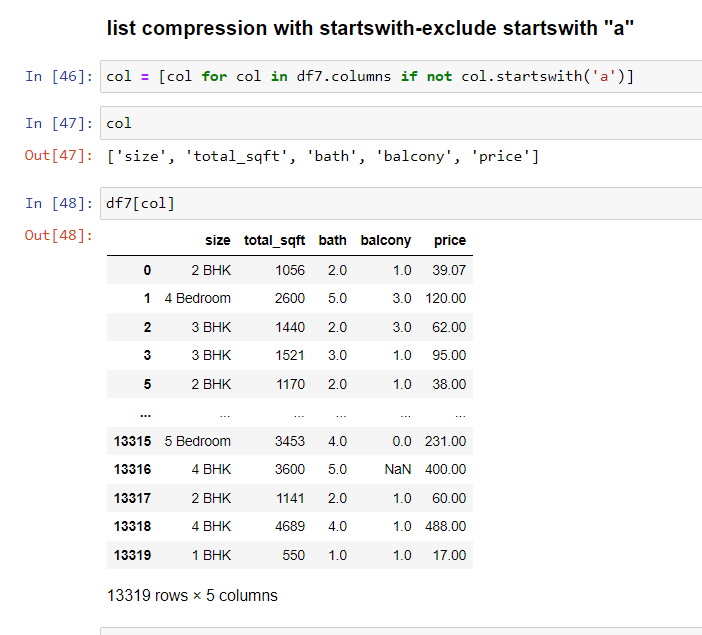


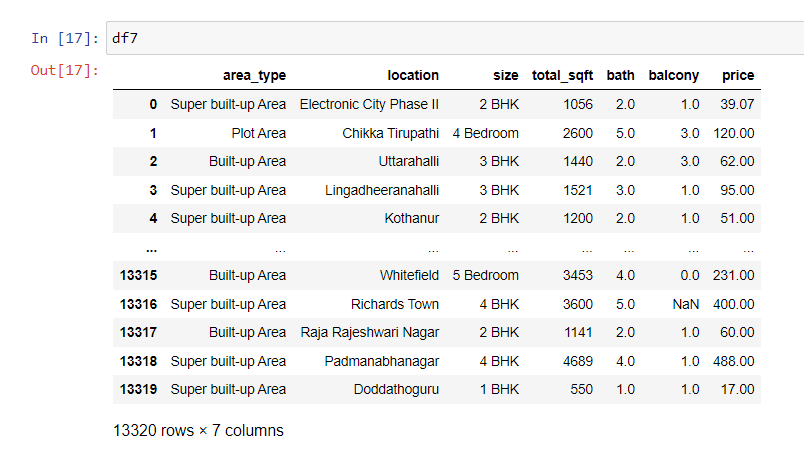


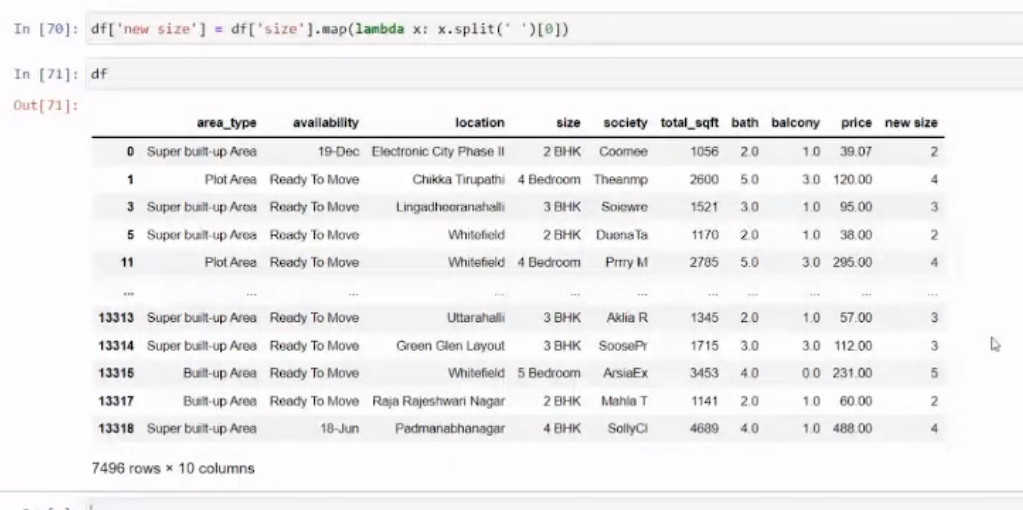


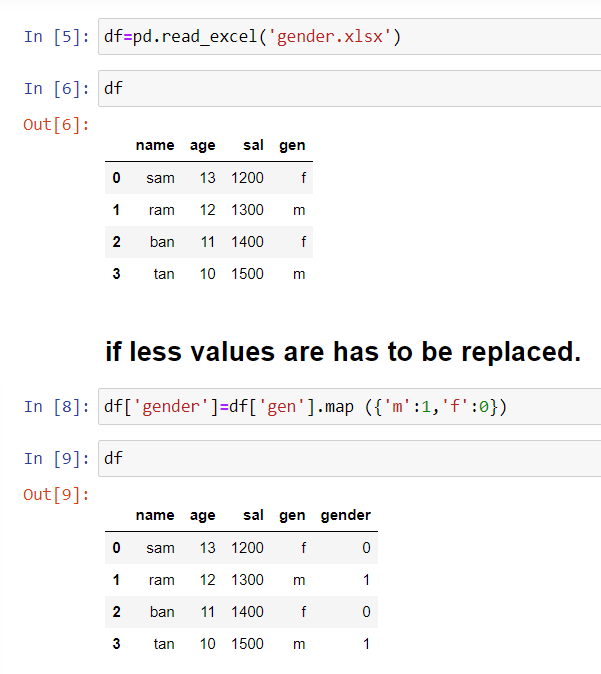


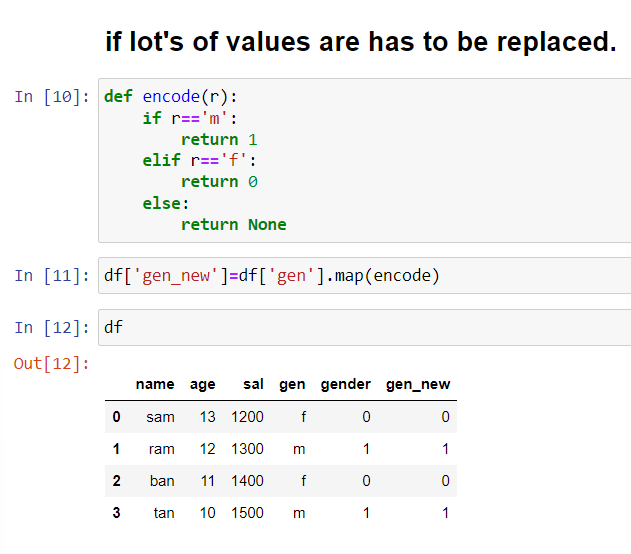


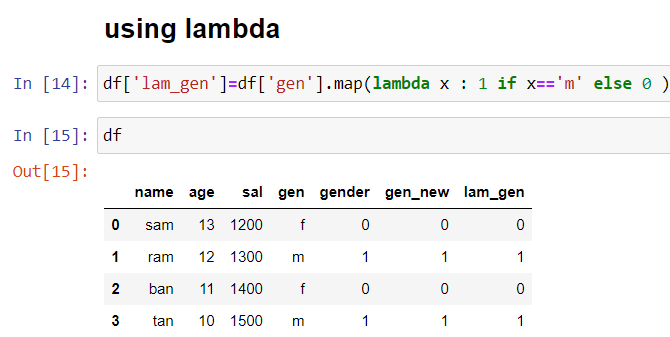


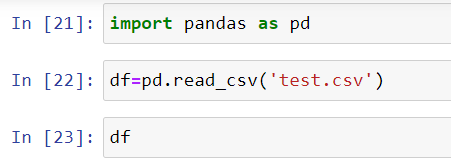


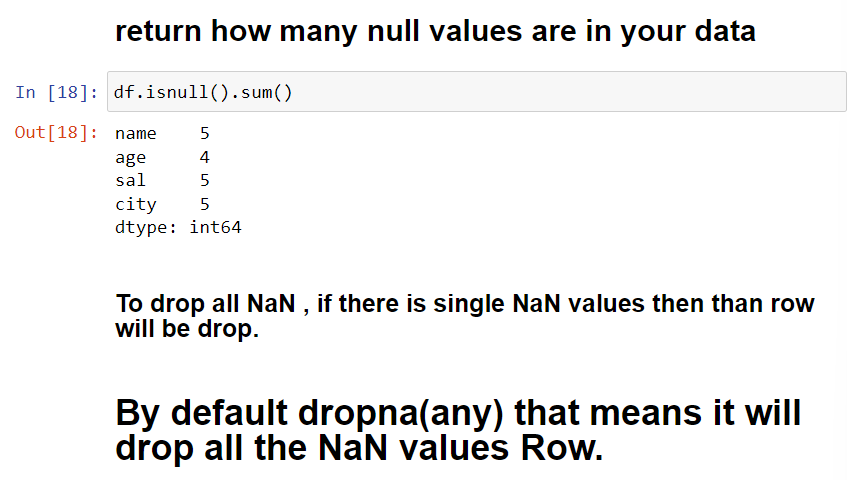
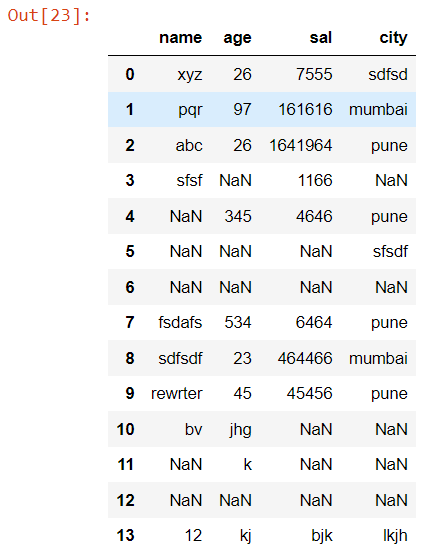


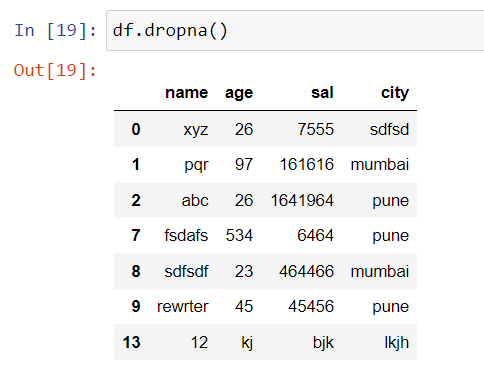


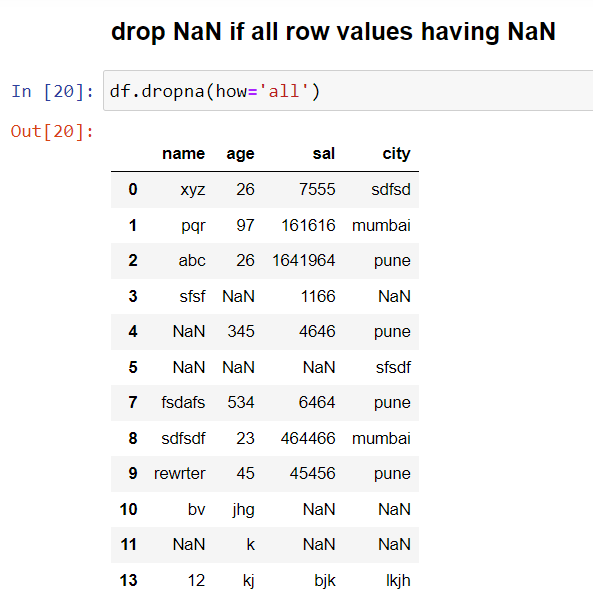


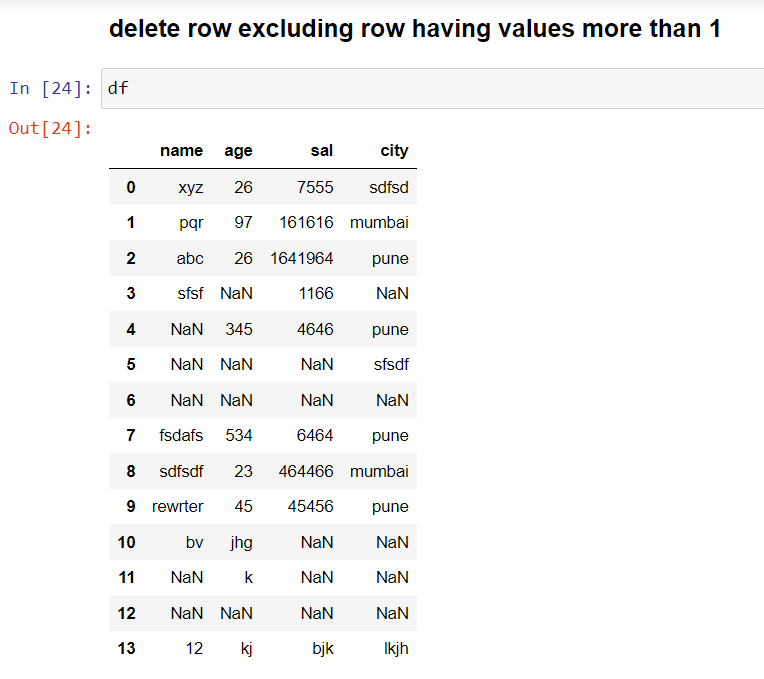


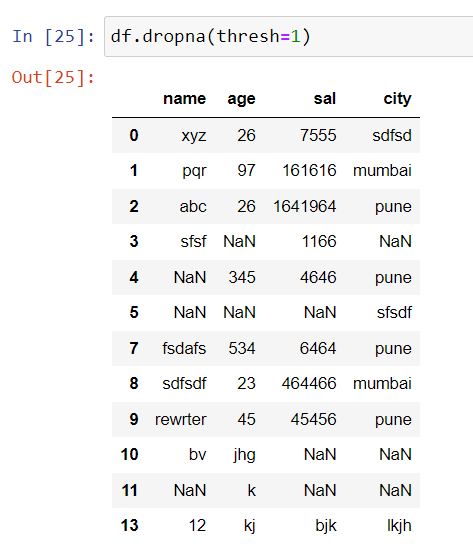


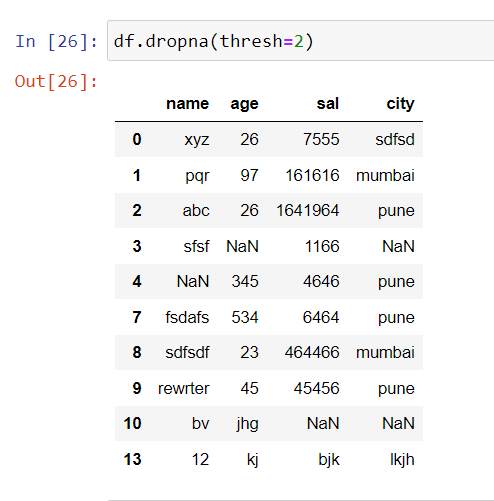


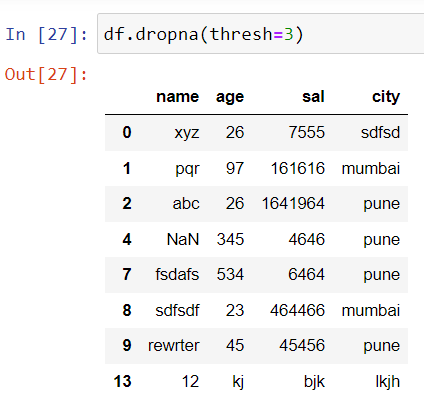


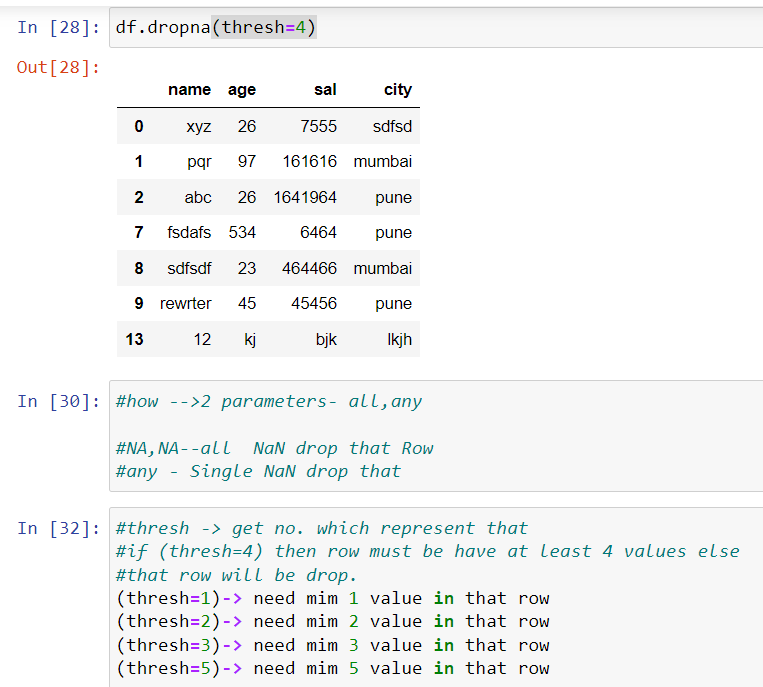


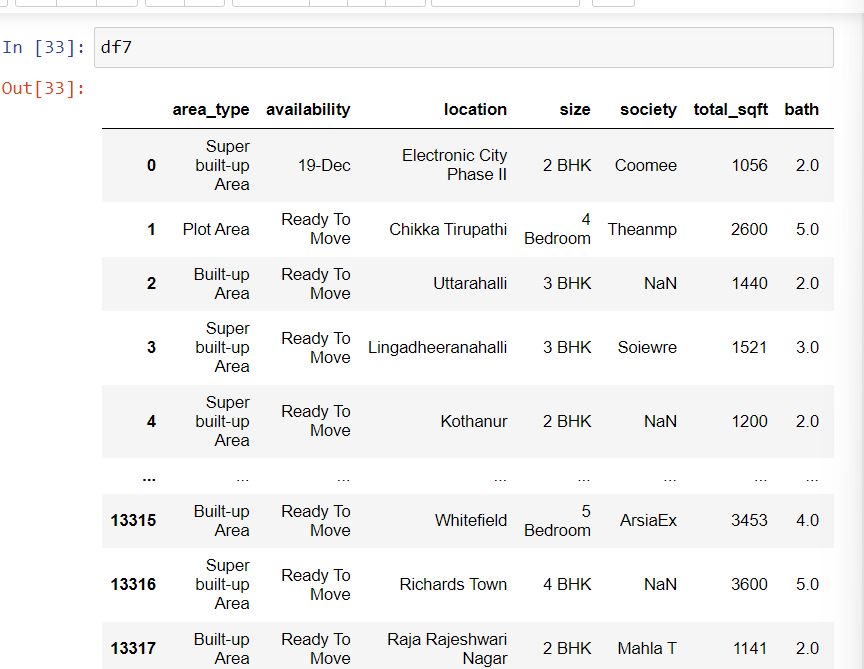




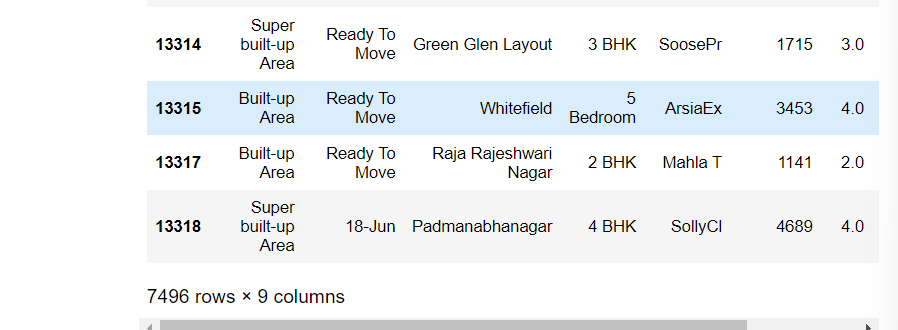


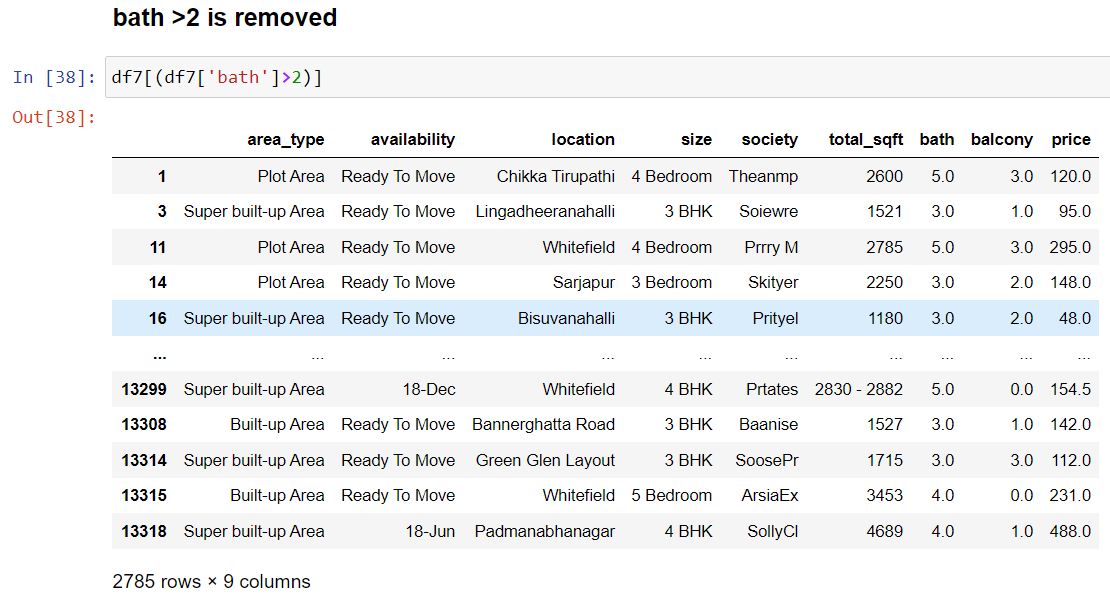


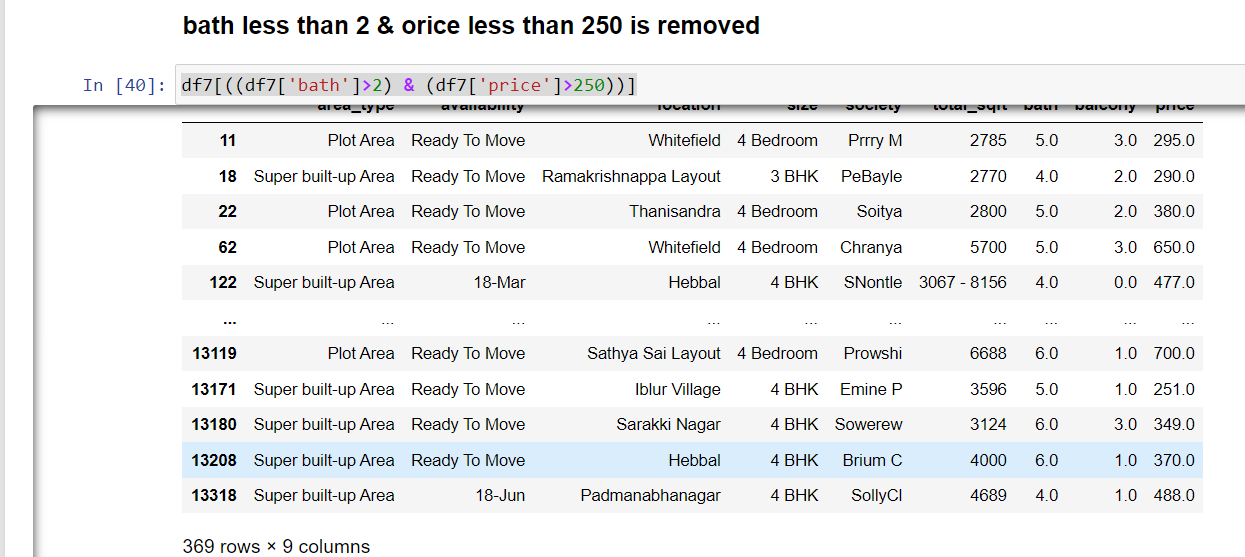


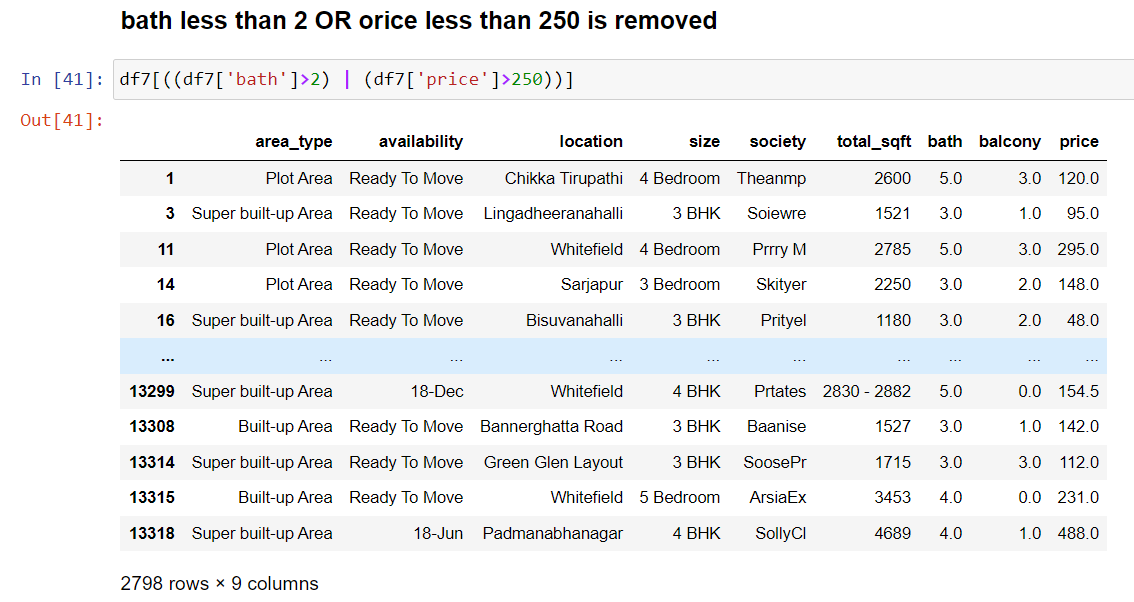


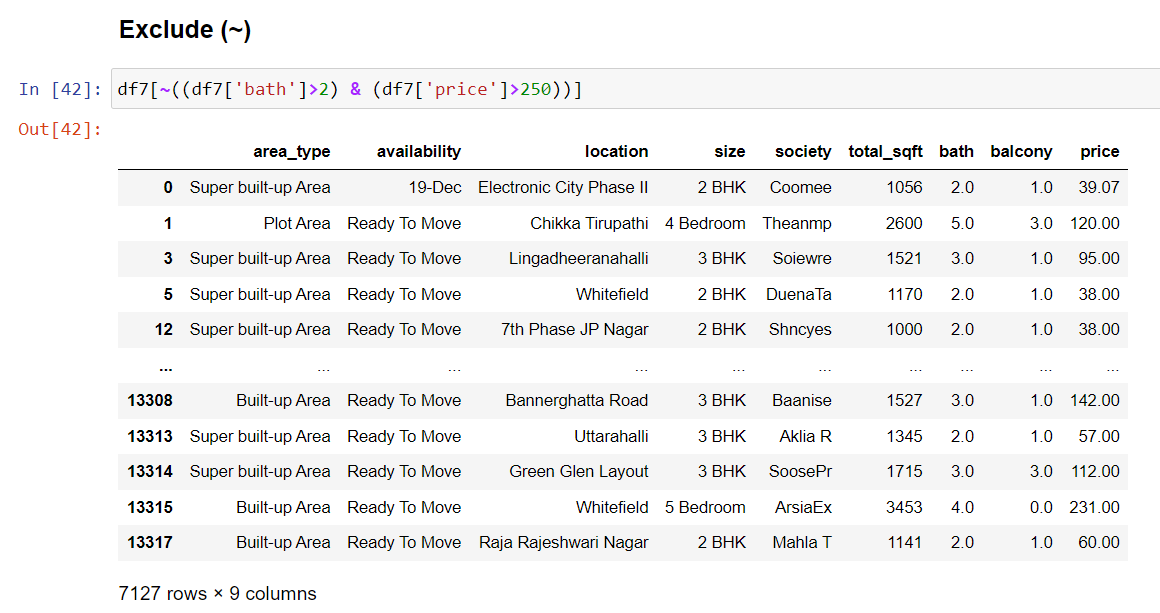


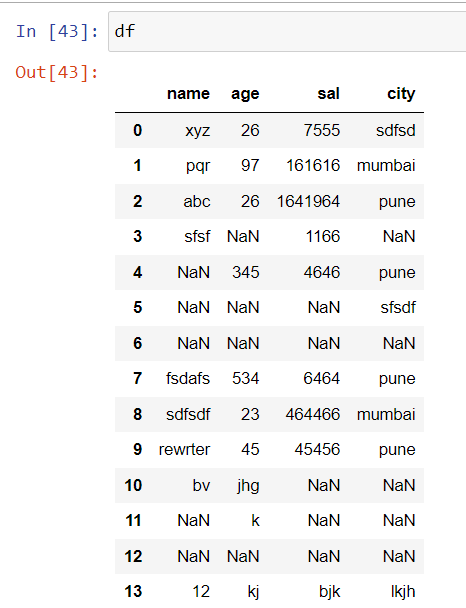


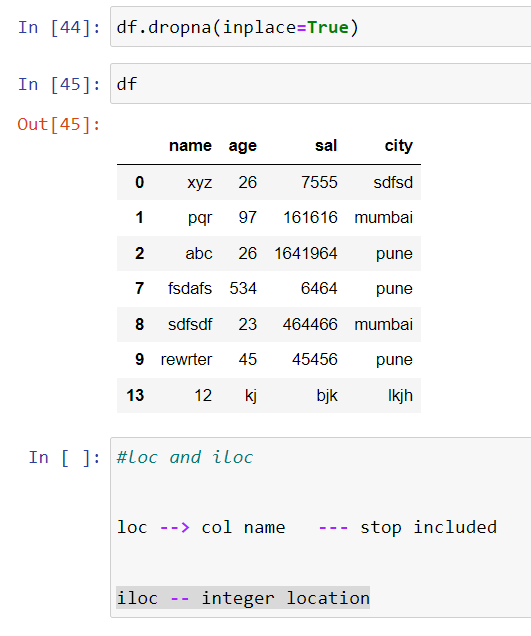


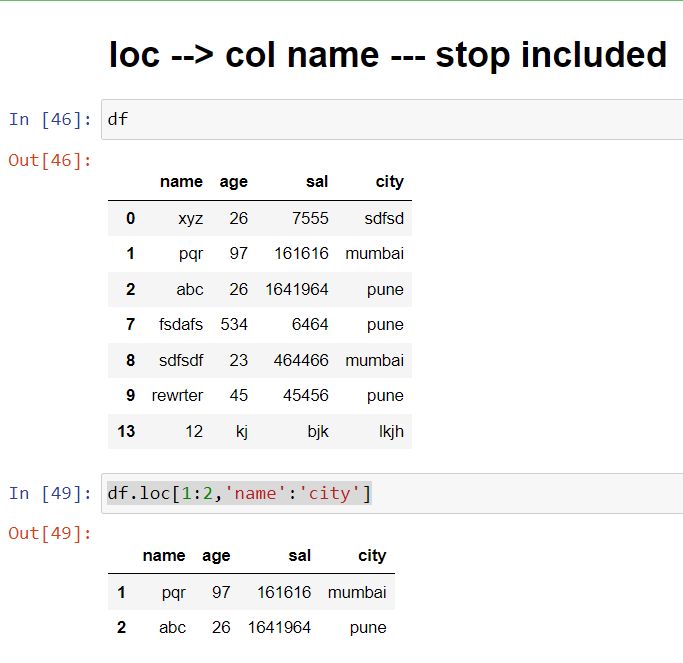


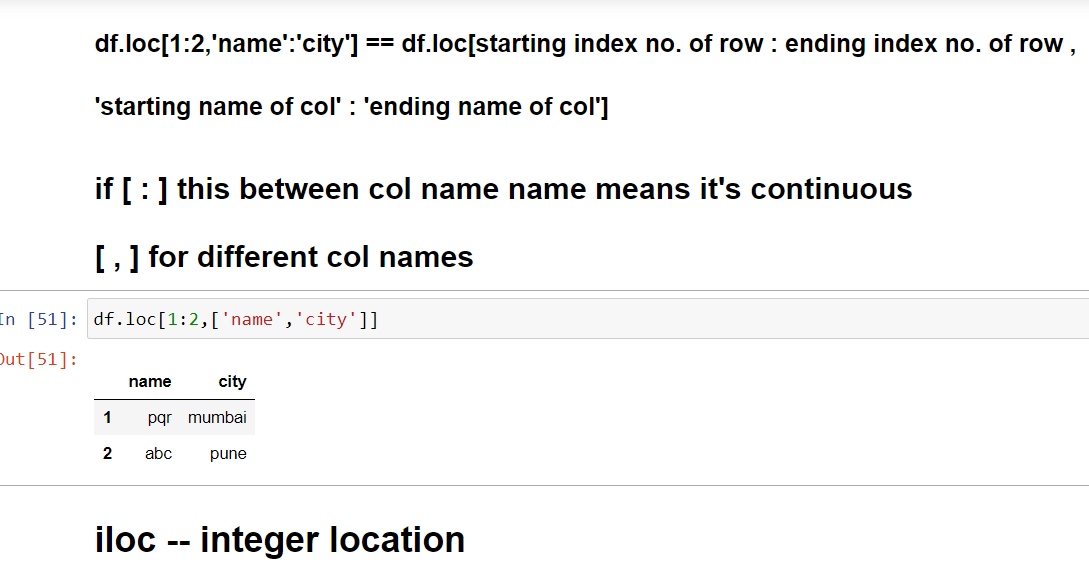


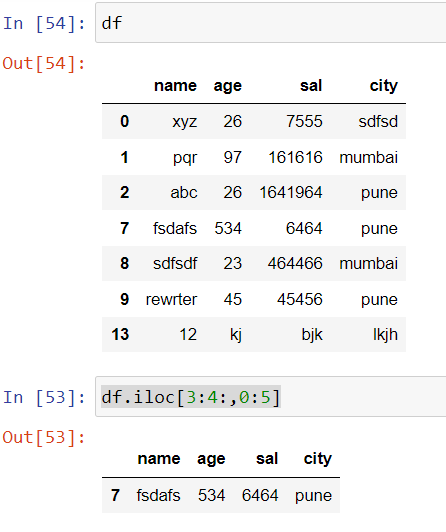


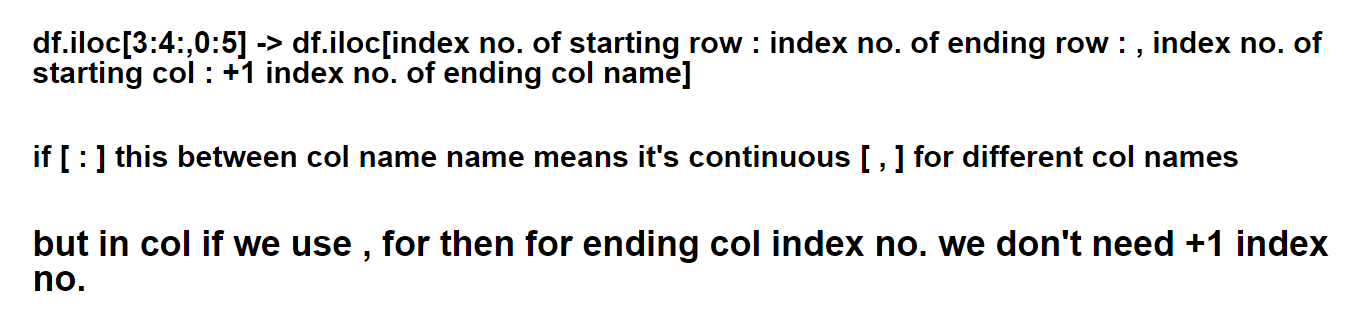


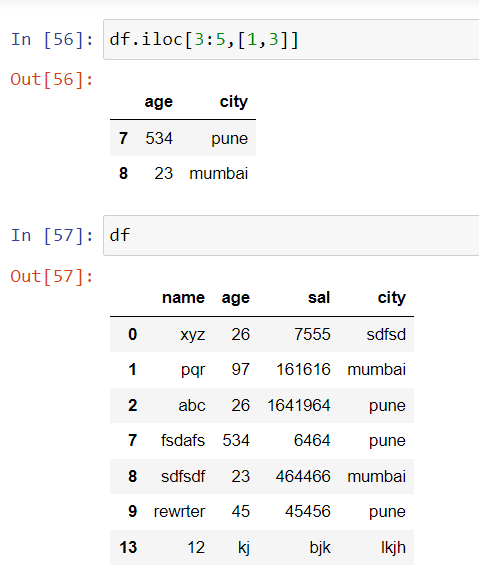


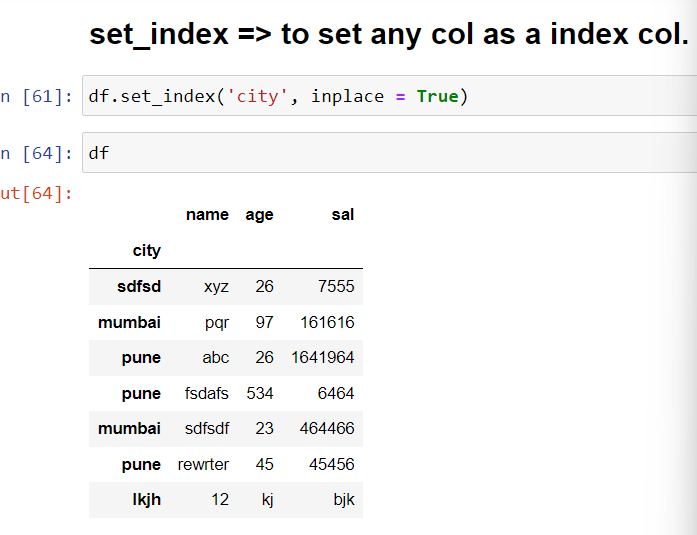


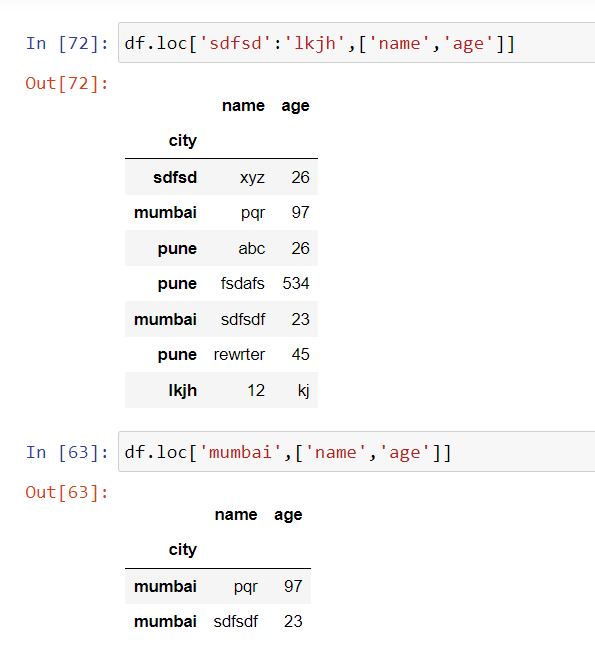


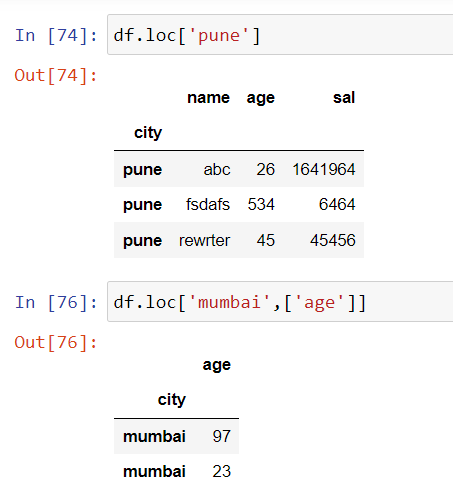


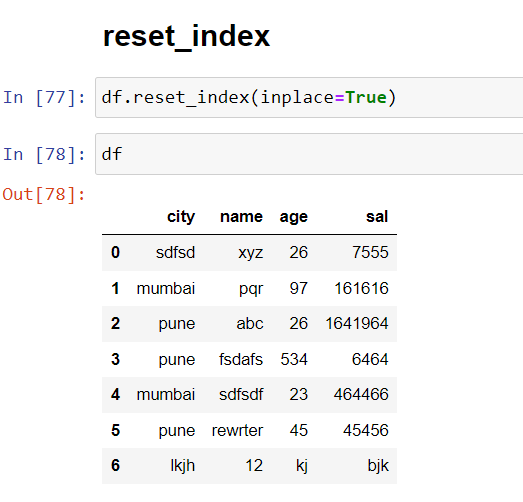












give difference between concat & merge regarding pandas python

ChatGPT

In Pandas, both the **concat** and **merge** functions are used for combining data from multiple DataFrames. However, they have some key differences in terms of their functionality and how they handle the data.

1. Purpose:
   * **concat**: The **concat** function is used for concatenating (appending) DataFrames vertically or horizontally along a particular axis. It is primarily used for combining multiple DataFrames with the same columns or different columns.
   * **merge**: The **merge** function is used for merging DataFrames based on a common column (or multiple columns) known as the "key." It is useful for combining DataFrames that have related information, similar to an SQL join operation.
2. Input:
   * **concat**: The **concat** function takes a sequence (list, tuple, or DataFrame) of DataFrames and concatenates them along a specified axis (0 for vertical concatenation, 1 for horizontal concatenation).
   * **merge**: The **merge** function takes two DataFrames and merges them based on common columns. It requires specifying the key column(s) on which the merge operation should be performed.
3. Output:
   * **concat**: The **concat** function returns a new DataFrame with the concatenated data from the input DataFrames. The resulting DataFrame will have the same columns as the input DataFrames and the concatenated rows.
   * **merge**: The **merge** function returns a new DataFrame that contains the merged data from the input DataFrames. The resulting DataFrame will have columns from both input DataFrames, with rows that satisfy the merge condition.
4. Operation:
   * **concat**: The **concat** function performs a simple concatenation operation by appending the rows (or columns) from the input DataFrames.
   * **merge**: The **merge** function performs a more complex operation by combining DataFrames based on matching values in the specified key column(s). It can perform different types of joins such as inner join, outer join, left join, and right join.
5. Key Column(s):
   * **concat**: The **concat** function does not require any key column(s) since it focuses on the order of rows or columns to concatenate.
   * **merge**: The **merge** function requires specifying one or more key column(s) on which the DataFrames will be merged. These key columns should exist in both DataFrames.

In summary, **concat** is used for combining DataFrames vertically or horizontally without any condition on common columns, while **merge** is used for merging DataFrames based on one or more key columns to combine related information from different DataFrames.

give difference between concat & merge regarding pandas python with example of code

ChatGPT

Certainly! Here's an example that demonstrates the differences between **concat** and **merge** functions in Pandas:

pythonCopy code

import pandas as pd # Create two sample DataFrames df1 = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]}) df2 = pd.DataFrame({'A': [7, 8, 9], 'C': [10, 11, 12]}) # Concatenate the DataFrames vertically using concat concatenated = pd.concat([df1, df2], axis=0) # Merge the DataFrames based on the 'A' column using merge merged = pd.merge(df1, df2, on='A') # Print the concatenated DataFrame print("Concatenated DataFrame:") print(concatenated) # Print the merged DataFrame print("\nMerged DataFrame:") print(merged)

Output:

rCopy code

Concatenated DataFrame: A B C 0 1 4.0 NaN 1 2 5.0 NaN 2 3 6.0 NaN 0 7 NaN 10.0 1 8 NaN 11.0 2 9 NaN 12.0 Merged DataFrame: A B C 0 7 4 10 1 8 5 11 2 9 6 12

In this example, we have two DataFrames, **df1** and **df2**, with some overlapping and non-overlapping columns. Let's understand the differences between **concat** and **merge**:

1. Concatenation (**concat**):
   * The **concat** function is used to vertically concatenate the two DataFrames (**df1** and **df2**) along the rows using **axis=0**.
   * The resulting DataFrame (**concatenated**) contains all the columns from both DataFrames, and the rows are appended one after another.
   * The non-overlapping columns will contain NaN values for the rows where the data was not available in the original DataFrame.
2. Merging (**merge**):
   * The **merge** function is used to merge the two DataFrames (**df1** and **df2**) based on the common column 'A'.
   * The resulting DataFrame (**merged**) contains the rows where the values in the 'A' column match in both DataFrames.
   * The columns 'B' and 'C' from the respective DataFrames are retained in the merged DataFrame.

Notice the differences in the output:

* The concatenated DataFrame contains all the columns from both DataFrames, and NaN values are present where the columns are not overlapping.
* The merged DataFrame only contains the rows where the values in the 'A' column match in both DataFrames, and the overlapping columns ('B' and 'C') are included.

These examples illustrate how **concat** and **merge** differ in their operations and the resulting output when combining DataFrames.