INDUSTRIAL TRAINING DAILY DIARY
DAY 11

07 July, 2025

Topic: Introduction to Pandas – Data Analysis with Python

Objectives:

- To understand the importance of pandas in data manipulation and preprocessing.
- To learn about pandas data structures: Series and DataFrame.
- To perform basic operations such as:
  - Creating Series and DataFrames.
  - Accessing rows and columns using `.loc[]`, `.iloc[]`, and indexing.
  - Importing data from CSV files using `read_csv()`.
  - Exploring datasets using functions like `.head()`, `.tail()`, `.info()`, and `.describe()`.
- To apply basic filtering, slicing, and data selection techniques for real-world datasets.

What's Pandas for?

Pandas has so many uses that it might make sense to list the things it can't do instead of what it can do.

This tool is essentially your data's home. Through pandas, you get acquainted with your data by cleaning, transforming, and analyzing it.

For example, say you want to explore a dataset stored in a CSV on your computer. Pandas will extract the data from that CSV into a DataFrame — a table, basically — then let you do things like:

- Calculate statistics and answer questions about the data, like
- What's the average, median, max, or min of each column?
- Does column A correlate with column B?
- What does the distribution of data in column C look like?

- Clean the data by doing things like removing missing values and filtering rows or columns by some criteria
- Visualize the data with help from Matplotlib. Plot bars, lines, histograms, bubbles, and more.
- Store the cleaned, transformed data back into a CSV, other file or database

Pandas First Steps

Install and import

Pandas is an easy package to install. Open up your terminal program (for Mac users) or command line (for PC users) and install it using either of the following commands:

```
conda install pandas
```

OR

```
pip install pandas
```

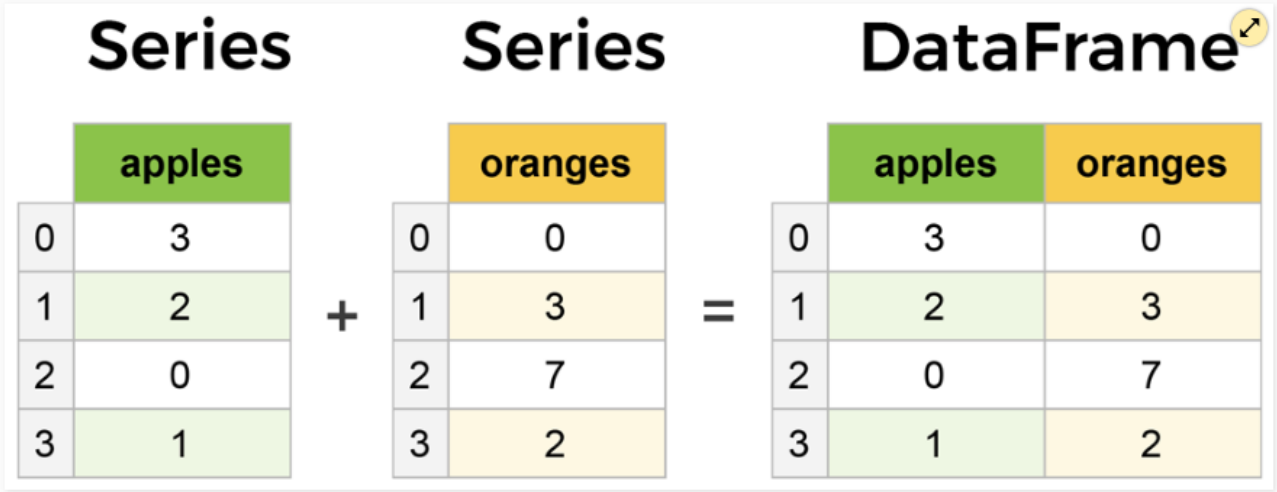
To import pandas we usually import it with a shorter name since it's used so much:

```
import pandas as pd
```

Now to the basic components of pandas.

Core components of pandas: Series and DataFrames

The primary two components of pandas are the `Series` and `DataFrame`. A `Series` is essentially a column, and a `DataFrame` is a multi-dimensional table made up of a collection of Series.



DataFrames and Series are quite similar in that many operations that you can do with one you can do with the other, such as filling in null values and calculating the mean.

You'll see how these components work when we start working with data below.

Creating DataFrames from scratch

Creating DataFrames right in Python is good to know and quite useful when testing new methods and functions you find in the pandas docs.

There are *many* ways to create a DataFrame from scratch, but a great option is to just use a simple `dict`.

Let's say we have a fruit stand that sells apples and oranges. We want to have a column for each fruit and a row for each customer purchase. To organize this as a dictionary for pandas we could do something like:

```
data = {
    'apples': [3, 2, 0, 1],
    'oranges': [0, 3, 7, 2]
}
```

And then pass it to the pandas DataFrame constructor:

```
purchases = pd.DataFrame(data)

purchases
```

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

How did that work?

Each *(key, value)* item in `data` corresponds to a *column* in the resulting DataFrame.
The **Index** of this DataFrame was given to us on creation as the numbers 0-3, but we could also create our own when we initialize the DataFrame.
Let's have customer names as our index:

```
purchases = pd.DataFrame(data, index=['June', 'Robert', 'Lily', 'David'])

purchases
```

	apples	oranges
June	3	0
Robert	2	3
Lily	0	7
David	1	2

So now we could **loc**ate a customer's order by using their name:

```
purchases.loc['June']
```

```
apples     3
oranges    0
Name: June, dtype: int64
```

SOME BASIC QUESTIONS RELATED TO PANDAS

Step 1. Go to https://www.kaggle.com/openfoodfacts/world-food-facts/data
Step 2. Download the dataset to your computer and unzip it.
Step 3. Use the tsv file and assign it to a dataframe called food

```
import pandas as pd
food = pd.read_csv(r"E:\en.openfoodfacts.org.products.tsv",sep="\t")
food
```

Step 4. See the first 5 entries

```
food.head()
```

Step 5. What is the number of observations in the dataset?

```
food.shape
```

```
(356027, 163)
```

Step 6. What is the number of columns in the dataset?

```
food.shape
```

```
(356027, 163)
```

Step 7. Print the name of all the columns.

```
food.columns
```

```
Index(['code', 'url', 'creator', 'created_t', 'created_datetime',
       'last_modified_t', 'last_modified_datetime', 'product_name',
       'generic_name', 'quantity',
       ...
       'fruits-vegetables-nuts_100g', 'fruits-vegetables-nuts-estimate_100g',
       'collagen-meat-protein-ratio_100g', 'cocoa_100g', 'chlorophyl_100g',
       'carbon-footprint_100g', 'nutrition-score-fr_100g',
       'nutrition-score-uk_100g', 'glycemic-index_100g',
       'water-hardness_100g'],
      dtype='object', length=163)
```

Step 8. What is the name of 105th column?

Step 9. What is the type of the observations of the 105th column?

Step 10. How is the dataset indexed?

Step 11. What is the product name of the 19th observation?

Step 2. Import the dataset from this address.

https://raw.githubusercontent.com/justmarkham/DAT8/master/data/u.user

Step 3. Assign it to a variable called users and use the 'user_id' as index

```
users = pd.read_csv(r"https://raw.githubusercontent.com/justmarkham/DAT8/master/data/u.user",sep = "|",header = 0)
```

Step 4. See the first 25 entries

```
users.head(25)
```

Step 5. See the last 10 entries

```
users.tail(10)
```

	user_id	age	gender	occupation	zip_code
933	934	61	M	engineer	22902
934	935	42	M	doctor	66221
935	936	24	M	other	32789
936	937	48	M	educator	98072
937	938	38	F	technician	55038
938	939	26	F	student	33319
939	940	32	M	administrator	02215
940	941	20	M	student	97229
941	942	48	F	librarian	78209
942	943	22	M	student	77841

Step 8. Print the name of all the columns.

```
users.columns
```

```
Index(['user_id', 'age', 'gender', 'occupation', 'zip_code'], dtype='object')
```

Step 9. How is the dataset indexed?

```
users.index
```

```
RangeIndex(start=0, stop=943, step=1)
```

Step 10. What is the data type of each column?

```
users.dtypes
```

```
user_id        int64
age            int64
gender        object
occupation    object
zip_code      object
dtype: object
```

Step 11. Print only the occupation column

```
print(users['occupation'])
```

```
0         technician
1              other
2             writer
3         technician
4              other
           ...
938          student
939    administrator
940          student
941        librarian
942          student
Name: occupation, Length: 943, dtype: object
```

Step 12. How many different occupations are in this dataset?

```
print(users['occupation'].value_counts().max())
```

```
196
```

Step 13. What is the most frequent occupation?

```
print(users['occupation'].value_counts().idxmax())
```

```
student
```

Step 14. Summarize the DataFrame.

```
users.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 943 entries, 0 to 942
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   user_id     943 non-null    int64 
 1   age         943 non-null    int64 
 2   gender      943 non-null    object
 3   occupation  943 non-null    object
 4   zip_code    943 non-null    object
dtypes: int64(2), object(3)
memory usage: 37.0+ KB
```

Step 15. Summarize all the columns

```
users.columns
```

```
Index(['user_id', 'age', 'gender', 'occupation', 'zip_code'], dtype='object')
```

Step 16. Summarize only the occupation column

```
print(users['occupation'].unique())
```

```
['technician' 'other' 'writer' 'executive' 'administrator' 'student'
 'lawyer' 'educator' 'scientist' 'entertainment' 'programmer' 'librarian'
 'homemaker' 'artist' 'engineer' 'marketing' 'none' 'healthcare' 'retired'
 'salesman' 'doctor']
```

Step 17. What is the mean age of users?

```
print(users['age'].mean())
```

```
34.05196182396607
```

Step 18. What is the age with least occurrence?

```
print(users['age'].value_counts().idxmin())
```

```
7
```