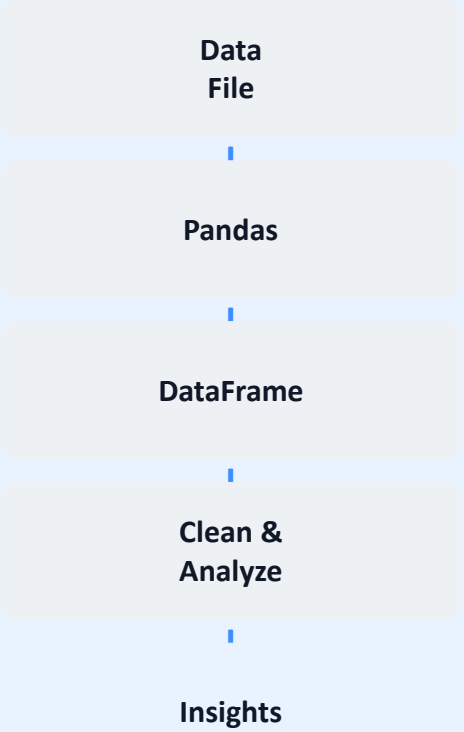
DATA SCIENCE

Working with Data Files in Pandas

From raw files to clean DataFrames

UNC GREENSBORO

THE DATA PATH



CSV File

CSV files store a table as plain text separated by commas.

```
Name,Age,Major,GPA
Alice,20,CS,3.7
Bob,21,Math,3.5
Carol,19,CS,3.9
```

- CSV means Comma-Separated Values
- Each row is one record
- Each column is one attribute
- The first row often contains column names

This looks very similar to a pandas DataFrame.

Reading a CSV File

Pandas converts a CSV file into a DataFrame in one command.

```
import pandas as pd

df = pd.read_csv("students.csv")
```

- pd is the conventional alias for pandas
- read_csv() reads the file
- The result is a DataFrame

Inspect Data Before Cleaning

Use a quick inspection workflow before changing anything.

```
df.head()
df.tail()
df.shape
df.columns
df.info()
```

Command	Question
head()	What does the data look like?
tail()	What does the end look like?
shape	How large is the dataset?
columns	What attributes do we have?
info()	What are the data types and missing values?

Do not start cleaning before you understand what you are looking at.

Working with Excel Files

Excel workbooks may contain more than one sheet.

```
df = pd.read_excel("students.xlsx")
```

```
df = pd.read_excel("students.xlsx",
                   sheet_name="Students")
```

Excel workbook

Sheet 1
Sheet 2
Students

sheet_name selects the sheet you want pandas to read.

Saving Cleaned Data

After cleaning, write the DataFrame back to a file.

```
df.to_csv("cleaned_students.csv", index=False)
```

```
df.to_excel("cleaned_students.xlsx", index=False)
```

index=False

Prevents pandas row indexes from becoming a column

Exercise

Test Your Understanding…

Download the Titanic dataset from course website.

Part A: Read the CSV File

1. Read titanic.csv into a DataFrame.
2. Display the first 6 rows.
3. Show:
    - Shape of the DataFrame
    - Column names
    - Data types

Part B: DataFrame and Series Operations

1. Select the Age column and compute:
    - Mean age
    - Maximum age
2. Count the number of passengers in each Pclass.
3. Filter passengers:
    - Age greater than 30
4. Compute:
    - Average age of survivors

Part C: Excel Operations

1. Save the DataFrame as titanic.xlsx.
2. Read the Excel file into a new DataFrame.
3. Display the first 3 rows.

Data Cleaning

Identify the problems in this deliberately messy table.

- Data cleaning means fixing bad data in your dataset.

- Bad data could be:
  - Missing data
  - Data in wrong format
  - Wrong data/ Outliers
  - Duplicates

Name	Age	Major	GPA
Alice	20	CS	3.7
Bob		CS	3.5
Carol	19	Computer Science	3.9
David	200	CS	3.6
Alice	20	CS	3.7
Eve	twenty	cs	3.8

Problem 1: Missing Data

NaN represents a missing numerical value.

Name	Age	Major	GPA
Alice	20	CS	3.7
Bob	NaN	CS	3.5
Carol	19	NaN	3.9

- A person did not answer
- Data was not collected
- A measurement failed
- Data was lost

Finding Missing Values

Find where values are missing before deciding what to do.

```
df.isna()

df.isna().sum()
```

```
Name      0
Age       1
Major     1
GPA       0
```

First identify where missing values are and how much data is missing.

Remove Rows with Missing Values

dropna() removes rows that contain missing information.

```
df.dropna()
```

Is removing data always a good idea?

- You may lose valuable information
- Many missing values can remove a large portion of the dataset

SMALL AMOUNT MISSING

10,000 responses
20 rows missing GPA

Dropping 20 rows may be reasonable.

LARGE AMOUNT MISSING

10,000 responses
4,000 rows missing GPA

Dropping rows could bias the analysis.

The right strategy depends on the amount and meaning of missing data.

Fill with a Specific Value

A label such as Unknown can preserve the fact that a value was missing.

- Replace all missing data with a specific data

```
df.fillna(200, inplace = True)
```

- Replace missing data with a specific data in a specific column

```
df["col_name"].fillna(130, inplace = True)
```

Deprecated starting from Pandas 2.1.0

```
df["Major"] = df["Major"].fillna(130)
```

- Replace missing data with Mean/Median of the column

```
df[" col_name "].fillna(df[" col_name "].mean(), inplace = True)
df[" col_name "].fillna(df[" col_name "].median(), inplace = True)
```

```
df["Age"] = df["Age"].fillna(
    df["Age"].mean()
)
```

Mean or Median?

Extreme values can pull the mean away from a typical value.

MEAN

- Uses all values
- Sensitive to extreme values

MEDIAN

- Middle value
- More resistant to extreme values

Salaries: $40K $42K $45K $47K $500K

For a missing salary, the median is often more useful.

Missing Data Decision Guide

Think about the quantity, type, and meaning of the missing values.

Always consider what the missing value means.

Problem 2: Wrong Format

These values represent dates, but they use inconsistent formats.

Name	Date
Alice	2026-01-10
Bob	01/15/2026
Carol	Jan 20, 2026

Converting Data to the Correct Format

Pandas should understand dates as dates, not ordinary strings.

```
df["Date"] = pd.to_datetime(df["Date"])
```

In newer versions of pandas, pd.to_datetime() requires specifying **format='mixed'** when the format is inconsistent.

Inconsistent Text Format

Pandas treats CS and cs as different values.

Major
CS
cs
Computer Science
CS
computer science

```
df["Major"] = df["Major"].str.lower()
```

```
CS
cs
Computer
Science
```

```
cs
cs
computer
science
```

Lowercase standardizes case. It does not make cs and computer science identical.

Mapping Inconsistent Values

A mapping converts different labels that share the same meaning.

```
major_map = {
    "cs": "Computer Science",
    "computer science": "Computer Science"
}

df["Major"] = df["Major"].replace(major_map)
```

Problem 3: Outliers

An extreme value can distort statistics and machine-learning models.

20, 21, 22, 23, 24

Mean ≈ 22

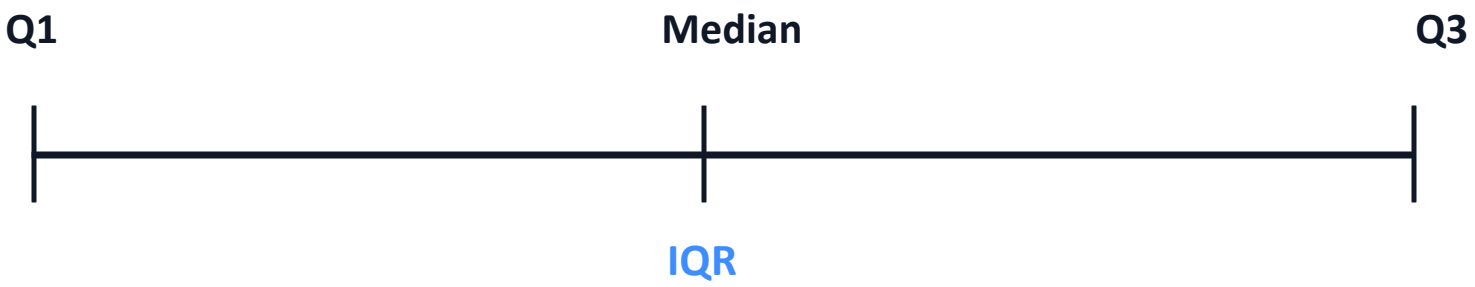
20, 21, 22, 23, 24, 500

Mean changes dramatically

Outliers can change the story your summary statistics tell.

Detecting Outliers with the IQR

Use the interquartile range to identify values far from the middle half of the data.




```
IQR = Q3 - Q1
```

```
Below Q1 - 1.5 × IQR
Above Q3 + 1.5 × IQR
```

Detecting Outliers in Pandas

Focus on the logic. The syntax follows the same IQR rule.

```
Q1 = df["Age"].quantile(0.25)
Q3 = df["Age"].quantile(0.75)

IQR = Q3 - Q1

outliers = df[
    (df["Age"] < Q1 - 1.5 * IQR) |
    (df["Age"] > Q3 + 1.5 * IQR)
]
```

What Should We Do with Outliers?

The appropriate response depends on whether the value is legitimate and useful.

Keep it
legitimate observation → **Correct it**
entry error → **Remove it**
invalid value → **Transform it**
when analysis needs it

Do not automatically delete outliers.

Problem 4: Duplicate Data

Repeated records can make a dataset look larger than it really is.

Name	Email	GPA
Alice	alice@email.com	3.7
Bob	bob@email.com	3.5
Alice	alice@email.com	3.7

Detecting Duplicates

duplicated() identifies rows that pandas considers duplicates.

```
df.duplicated()
```

If the corresponding data is duplicated, duplicated() will return True; otherwise, it will return False.

```
df = df.drop_duplicates()
```

Remove duplicate

```
df.drop_duplicates(subset=["Email"])
```

Sometimes an entire row does not need to match. A unique email may be enough.

COMPLETE WORKFLOW

A Complete Data Cleaning Workflow

The process moves from file input to a validated output.


```
df = pd.read_csv("students.csv")

df.info()

df["Major"] = df["Major"].fillna("Unknown")
df["Major"] = df["Major"].str.lower()
df = df.drop_duplicates()
df.to_csv("cleaned_students.csv", index=False)
```

IN-CLASS PRACTICE

Let’s Clean a Messy Dataset

Identify the problems first. Do not write pandas commands yet.

In-class exercise

- Download Staff.csv and read file using Pandas
- Handle missing values:
- Replace missing values in Age with the number 30.
- Replace missing values in Salary with the mean salary.
- Remove duplicated rows from the dataset.
- Handle outliers:
- Replace any salary greater than 200000 with the median salary of the column.
- Replace any age greater than 120 with 120.
- Standardize data types
- Convert Age to integer type.
- Write clean data to a csv file

IN-CLASS PRACTICE

Check Your Work

Cleaning requires validation. Run checks again after every important change.

```
df.info()
df.isna().sum()
df.duplicated().sum()
df.describe()
```

Is our dataset actually clean now?