

PANDAS COMPREHENSIVE ASSIGNMENT 4

Mastering Pandas Fundamentals

Data Analysis with Python

PANDAS COMPREHENSIVE ASSIGNMENT

Mastering Pandas Fundamentals

ASSIGNMENT OVERVIEW

Course: Data Science with Python

Week: 4

Topic: Comprehensive Pandas Fundamentals

Duration: 3-4 hours

Total Points: 100 points

Due Date: 16/02/2026 11:59pm (WAT)

LEARNING OBJECTIVES

By completing this assignment, you will demonstrate mastery of:

1. Creating and manipulating DataFrames and Series
2. Data selection and indexing techniques
3. Data cleaning and handling missing values
4. Basic statistical analysis and descriptive statistics
5. Sorting and filtering data
6. GroupBy operations and aggregations
7. Data transformation and manipulation
8. Working with different data types
9. Writing clean and efficient Pandas code

PREREQUISITES

Before starting this assignment, ensure you have:

- Python 3.8 or higher installed
- Pandas library installed (`pip install pandas`)

- NumPy library installed (`pip install numpy`)
- Basic Python programming knowledge

DATASETS PROVIDED

You will work with four CSV files:

1. sales_data.csv (1000 rows)
 - Contains transaction data with dates, products, regions, sales representatives, quantities, and prices
2. customer_data.csv (500 rows)
 - Contains customer information including demographics and purchase history
3. inventory_data.csv (100 rows)
 - Contains product inventory with some missing values (intentional for practice)
4. employee_data.csv (200 rows)
 - Contains employee information including departments, salaries, and performance metrics

SECTION 1: DATAFRAME BASICS

Question 1.1 (3 points)

Working with DataFrames:

- a) Display the first 10 rows of the sales dataset
- b) Display the last 5 rows of the customer dataset
- c) Show the column names of the employee dataset
- d) Display the shape (rows and columns) of all four datasets

Question 1.2

Data Information and Types:

- a) Display the data types of all columns in the sales dataset
- b) Use `info()` to get a concise summary of the inventory dataset
- c) Check which columns in the inventory dataset have missing values
- d) Count the total number of missing values in each column of inventory_data

Question 1.3

Summary Statistics:

- a) Display summary statistics (mean, median, std, etc.) for all numerical columns in the sales dataset
- b) Find the minimum and maximum salary in the employee dataset
- c) Calculate the mean age of customers
- d) Find the total count of products in the inventory dataset

Question 1.4

Creating New DataFrames:

- a) Create a new DataFrame with only these columns from sales_data: 'product', 'quantity', 'total_amount'
- b) Create a new DataFrame containing only customers from 'New York'
- c) Create a DataFrame showing employees in the 'IT' department
- d) Display the first 5 rows of each newly created DataFrame

SECTION 2: DATA SELECTION AND INDEXING

Question 2.1

Basic Selection:

- a) Select the 'product' column from sales_data and display the first 10 values
- b) Select multiple columns: 'name', 'age', 'city' from customer_data
- c) Select the first 20 rows from sales_data
- d) Select rows 50 to 60 from employee_data

Question 2.2

Conditional Selection:

- a) Filter sales_data to show only sales where quantity > 10
- b) Filter customer_data to show customers aged between 30 and 50

- c) Find all employees with salary greater than \$100,000
- d) Show products in inventory where stock_quantity is less than 100

Question 2.3

Advanced Filtering:

- a) Find sales where product is 'Laptop' AND quantity > 5
- b) Find customers who are either from 'Chicago' OR have total_spent > \$5000
- c) Find employees in 'Sales' OR 'Marketing' departments with salary > \$80,000
- d) Find inventory items where supplier is NOT 'Supplier_A'

Question 2.4

Using .loc and .iloc:

- a) Use .loc to select rows where region is 'North' and columns 'product', 'total_amount'
- b) Use .iloc to select the first 10 rows and first 3 columns of customer_data
- c) Use .loc to select specific rows by index (rows 5, 10, 15, 20) from employee_data
- d) Use .iloc to select every 5th row from sales_data (rows 0, 5, 10, 15, etc.)

SECTION 3: DATA CLEANING

Question 3.1

Identifying Missing Data:

- a) Check if there are any missing values in sales_data (True/False for entire DataFrame)
- b) Count how many missing values exist in each column of inventory_data
- c) Calculate the percentage of missing values for each column in inventory_data
- d) Identify which rows have missing values in the 'unit_cost' column

Question 3.2 (5 points)

Handling Missing Data:

- a) Fill missing values in 'stock_quantity' with the median value
- b) Fill missing values in 'unit_cost' with the mean value
- c) Fill missing values in 'supplier' with the string 'Unknown'
- d) Verify that all missing values have been handled by checking again

Question 3.3

Removing Duplicates and Data Cleaning:

- a) Check if there are any duplicate rows in customer_data
- b) If duplicates exist, remove them and show the count before and after
- c) In sales_data, check for any negative values in 'quantity' or 'total_amount'
- d) Remove any rows with negative values (if they exist)

SECTION 4: SORTING AND RANKING

Question 4.1

Basic Sorting:

- a) Sort sales_data by 'total_amount' in descending order and display top 10
- b) Sort customer_data by 'age' in ascending order
- c) Sort employee_data by 'salary' in descending order and show the top 5 earners
- d) Sort inventory_data by 'stock_quantity' to find products with lowest stock

Question 4.2

Multi-column Sorting:

- a) Sort sales_data by 'region' (ascending) and then by 'total_amount' (descending)
- b) Sort employee_data by 'department' (ascending) and 'salary' (descending)
- c) Sort customer_data by 'city' and then 'total_spent' (descending)
- d) Display the first 10 rows of each sorted result

SECTION 5: GROUPBY AND AGGREGATIONS

Question 5.1

Basic Aggregations:

- a) Group sales_data by 'product' and calculate total quantity sold for each product
- b) Group customer_data by 'city' and calculate the mean age in each city
- c) Group employee_data by 'department' and count how many employees in each department
- d) Group sales_data by 'region' and calculate the sum of total_amount for each region

Question 5.2

Multiple Aggregations:

- a) Group sales_data by 'product' and calculate:
 - Sum of quantity
 - Mean of unit_price
 - Count of transactions
- b) Group employee_data by 'department' and calculate:
 - Mean salary
 - Median salary
 - Maximum salary
- c) Display both results in a clear format

Question 5.3

Finding Insights with GroupBy:

- a) Which product has the highest total revenue? (sum of total_amount by product)
- b) Which region has the highest average transaction value?
- c) Which department has the highest average salary?
- d) Which city has the most customers?

Question 5.4

Advanced GroupBy:

- a) Group sales_data by 'sales_rep' and calculate their total sales (sum of total_amount)
- b) Rank the sales representatives from best to worst
- c) Calculate what percentage of total revenue each sales rep contributed
- d) Display the top 5 performing sales representatives

SECTION 6: DATA TRANSFORMATION

Question 6.1

Creating New Columns:

- a) In sales_data, create a new column 'revenue_per_unit' = total_amount / quantity
- b) In customer_data, create a new column 'spending_per_purchase' = total_spent / total_purchases

(Handle cases where total_purchases is 0)

c) In employee_data, create a new column 'tenure_category':

- 'New' if hire_date is after 2020
- 'Experienced' if hire_date is between 2015 and 2020
- 'Senior' if hire_date is before 2015

d) Display the first 10 rows showing the new columns

Question 6.2

Data Type Conversions:

- Convert the 'date' column in sales_data to datetime type
- Extract the month from the date and create a 'month' column
- Extract the day of week and create a 'day_of_week' column
- Display summary showing count of sales by month

SUBMISSION GUIDELINES

What to Submit:

1. Jupyter Notebook (.ipynb) OR Python Script (.py)

- Include all your code with clear comments
- Show all outputs for each question

2. README file explaining:

- How to run your code
- Any assumptions you made
- Brief summary of your findings

File Naming Convention:

- `firstname_lastname_pandas_assignment.ipynb` (or .py)
- `firstname_lastname_readme.txt`

Submission Format:

- Compress all files into a single ZIP file

- Name the ZIP file: `firstname_lastname_pandas.zip`

RESOURCES

Official Documentation:

- Pandas Documentation: <https://pandas.pydata.org/docs/>
- Pandas User Guide: https://pandas.pydata.org/docs/user_guide/index.html
- Pandas Cheat Sheet: https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf

ACADEMIC INTEGRITY

This is an individual assignment. All code must be your own work.

Prohibited:

- Copying code from other students
- Sharing your solution files
- Using complete solutions from online sources without understanding

Allowed:

- Consulting official documentation
- Reviewing course materials
- Asking clarifying questions to instructor/TA
- Using Stack Overflow for syntax help (with attribution)

Violations will result in a zero on the assignment and may be reported to the academic integrity office.