

PANDAS PRACTICE SHEET (25 QUESTIONS)

◆ LEVEL 1 — Core Pandas Fundamentals (Q1–Q7)

Q1. Data Loading & Inspection

Given employees.csv:

emp_id name dept salary join_date

1. Load the dataset
 2. Display:
 - shape
 - column names
 - data types
 3. Show first 3 and last 2 rows
 4. Find memory usage
-

Q2. Column Operations

1. Rename dept → department
 2. Create a new column salary_annual = salary * 12
 3. Drop join_date
 4. Reorder columns logically
-

Q3. Indexing & Selection

1. Select only name and salary
 2. Select employees earning > 60,000
 3. Select rows with index 5 to 10
 4. Select employees in HR or IT
-

Q4. Sorting & Ranking

1. Sort employees by salary (descending)

2. Rank employees by salary (dense ranking)
 3. Find top 5 highest paid employees
-

Q5. Missing Values

Given missing salaries and departments:

1. Count missing values per column
 2. Fill salary NaNs with median
 3. Drop rows where department is missing
 4. Forward-fill remaining NaNs
-

Q6. Value Counts & Unique

1. Count employees per department
 2. Find number of unique departments
 3. Get department names sorted alphabetically
-

Q7. Data Type Conversion

1. Convert join_date to datetime
 2. Extract year, month, day
 3. Convert salary to integer
-

◆ LEVEL 2 — Intermediate Data Manipulation (Q8–Q16)

Q8. Conditional Columns

1. Create column salary_band:
 - <40k → Low
 - 40k–70k → Medium
 - >70k → High
2. Count employees per salary band

Q9. Apply vs Vectorization

1. Increase salary by 10% for IT department
2. Do it:
 - once using vectorization
 - once using apply
3. Compare performance (conceptual)

Q10. GroupBy Aggregations

Group by department and compute:

1. Mean salary
2. Max salary
3. Employee count
4. Total payroll

Q11. GroupBy + Filtering

1. Find departments with:
 - avg salary > 60,000
2. Find departments having more than 10 employees

Q12. Multi-Aggregation

Use agg() to compute:

- min salary
- max salary
- std deviation
per department

Q13. String Operations

On email column:

1. Extract domain
 2. Convert to lowercase
 3. Check which emails belong to gmail.com
 4. Replace domain with company.com
-

Q14. Datetime Analysis

1. Employees joined after 2020
 2. Employees with tenure > 3 years
 3. Monthly hiring count
-

Q15. Duplicate Handling

1. Detect duplicate employees
 2. Remove duplicates keeping latest entry
 3. Reset index after removal
-

Q16. Data Validation

1. Check if any salary is negative
 2. Assert no duplicate emp_id
 3. Validate salary column is numeric
-

◆ LEVEL 3 — Advanced / Interview-Grade Pandas (Q17–Q25)

Q17. Merging DataFrames

Given:

- employees
 - departments (dept_id, dept_name)
1. Perform inner join

2. Perform left join
 3. Identify employees without department
-

Q18. Multiple Data Sources

Given monthly sales files (sales_jan.csv, sales_feb.csv...):

1. Combine all into one DataFrame
 2. Add column month
 3. Compute total revenue per month
-

Q19. Window Functions

For each department:

1. Rank employees by salary
 2. Compute salary difference from department average
 3. Find top earner per department
-

Q20. Pivot Tables

Create pivot table:

- rows → department
 - columns → salary_band
 - values → employee count
-

Q21. Time-Series Resampling

Given daily sales data:

1. Convert date to index
2. Compute:
 - weekly sum
 - monthly mean
3. Identify highest sales month

Q22. Performance Optimization

1. Convert department to category
2. Explain why this improves memory
3. Measure memory before & after

Q23. Real-World Log Analysis

Given website logs:

| user_id | page | timestamp | duration |

1. Sessions per user
2. Average session duration
3. Most visited page
4. Peak traffic hour

Q24. Data Cleaning Case Study

Given messy customer data:

1. Standardize names
2. Fix inconsistent city names
3. Remove invalid ages
4. Handle missing emails

Q25. Interview-Style Problem (Capstone)

Given orders DataFrame:

| order_id | user_id | product | price | order_date |

1. Monthly revenue trend
2. Top 5 customers by spending
3. Repeat customer rate
4. Average order value per month

5. Detect outlier orders