

Analyx Academy

Python for Data Analyst – Interview Questions

1️⃣ Core Python (Must-Know)

Q1. Difference between list, tuple, set, and dictionary?

Answer (interview-ready):

- **List** → Ordered, mutable, allows duplicates
- **Tuple** → Ordered, immutable, faster than list
- **Set** → Unordered, unique elements only
- **Dict** → Key–value pairs, fast lookup

👉 *Use case:*

- List → row-wise data
- Tuple → fixed config
- Set → remove duplicates
- Dict → mappings (id → value)

Q2. What is list comprehension? Example.

squares = [x*x for x in range(1, 6)]

✅ Cleaner, faster, readable
Used heavily in data transformation steps.

Q3. Difference between == and is?

- == → compares **values**
- is → compares **memory location**

a = [1,2]; b = [1,2]

a == b # True

a is b # False

Q4. What are lambda functions?

Anonymous one-line functions.

lambda x: x*2

Used in map(), apply(), sorting, filtering.

2️⃣ NumPy (Interview Expectations)

Q5. Why NumPy over Python lists?

- Faster (C-based)
- Vectorized operations
- Less memory
- Mathematical operations

import numpy as np

arr = np.array([1,2,3])

arr * 2

Q6. Difference between np.array and Python list?

Feature	List	NumPy Array
Speed	Slow	Fast
Type	Mixed	Same datatype
Ops	Loop needed	Vectorized

Q7. How do you handle missing values in NumPy?

np.nanmean(arr)

np.isnan(arr)

3️⃣ Pandas – MOST IMPORTANT

Q8. Difference between loc and iloc?

- loc → label-based
- iloc → index-based

df.loc[1, 'salary']

df.iloc[1, 2]

Q9. How do you handle missing values?

df.isnull().sum()

df.dropna()

df.fillna(0)

df['salary'].fillna(df['salary'].mean())

👉 Always explain **business logic** behind filling.

Q10. Difference between apply(), map(), and applymap()?

- map() → Series only
- apply() → row/column-wise
- applymap() → element-wise (DataFrame)

Q11. How do you remove duplicates?

```
df.drop_duplicates()
df.drop_duplicates(subset=['user_id'])
```

Q12. GroupBy – common interview pattern

```
df.groupby('department')['salary'].mean()
```

Advanced:

```
df.groupby('dept').agg(
    avg_salary=('salary','mean'),
    emp_count=('emp_id','count')
)
```

Q13. Difference between merge, join, concat?

Function	Use
merge	SQL-like join
join	Index-based
concat	Stack rows/columns

```
pd.merge(df1, df2, on='id', how='left')
```

Q14. How do you filter data?

df[df['salary'] > 50000]

df.query("salary > 50000 and dept == 'IT'")

Q15. How do you handle date & time?

df['date'] = pd.to_datetime(df['date'])

df['year'] = df['date'].dt.year

df['month'] = df['date'].dt.month

4️⃣ EDA & Business Logic Questions

Q16. Steps of Exploratory Data Analysis?

1. Understand data
2. Check nulls & duplicates
3. Data types correction
4. Outlier detection
5. Distribution analysis
6. Business insights

Q17. How do you detect outliers?

Q1 = df['salary'].quantile(0.25)

Q3 = df['salary'].quantile(0.75)

IQR = Q3 - Q1

df[(df['salary'] < Q1 - 1.5*IQR) | (df['salary'] > Q3 + 1.5*IQR)]

Q18. Explain a real project using Python

Best structure (STAR):

- **Situation:** Data source & problem
- **Task:** What analysis needed
- **Action:** pandas, NumPy, cleaning, logic
- **Result:** Business impact

5️⃣ Performance & Optimization

Q19. How do you optimize pandas code?

- Avoid loops
- Use vectorization
- Use categorical dtype
- Filter early
- Use .copy() properly

Bad ❌

```
for i in df:
    ...
```

Good ✅

```
df['new'] = df['a'] * df['b']
```

Q20. Difference between shallow copy & deep copy?

```
df2 = df.copy()  # deep copy
```

Avoid SettingWithCopyWarning.

6️⃣ Scenario-Based Questions

Q21. Dataset has 10M rows, pandas is slow – what will you do?

✅ Expected answer:

- Read selective columns
- Filter at source
- Use chunking
- Convert dtypes
- Move aggregation to SQL

Q22. SQL vs Python – when to use what?

- **SQL** → heavy joins, aggregation
- **Python** → cleaning, complex logic, visualization

👉 Interviewers LOVE this answer.

7️⃣ Python Interview Coding Questions

1. Find second highest salary
2. Remove duplicates without using drop_duplicates
3. Count frequency of words
4. Convert column to rank
5. Fill missing salary with dept average
6. Detect duplicate transactions within 10 minutes
7. Normalize column values

Project-Based Python Interview Questions & Answers

1️⃣ Data Cleaning Project Questions

Q1. ***Explain how you cleaned raw data using Python in your project.***

Answer:

I received raw data from multiple sources containing missing values, duplicate records, incorrect data types, and inconsistent formats.
I used **pandas** to:

- Identify null values using isnull()
- Remove or impute missing values based on business logic
- Remove duplicates using drop_duplicates()
- Standardize column names and formats
- Convert data types using astype() and to_datetime()

```
df.isnull().sum()
```

```
df.drop_duplicates(inplace=True)
```

```
df['date'] = pd.to_datetime(df['date'])
```

```
df['sales'].fillna(df['sales'].median(), inplace=True)
```

Q2. ***How did you handle missing values differently for different columns?***

Answer:

I didn’t use a single approach.

- Numerical columns → mean/median
- Categorical columns → mode or “Unknown”
- Critical fields → row removal after validation

```
df['salary'].fillna(df['salary'].median(), inplace=True)
```

```
df['city'].fillna('Unknown', inplace=True)
```

2️⃣ Exploratory Data Analysis (EDA) Questions

Q3. ***How did you perform EDA in your project using Python?***

Answer:

I started with data understanding and structure, then analyzed distributions, trends, and anomalies.

Steps:

1. Shape & schema
2. Missing values & duplicates
3. Statistical summary
4. Outlier detection
5. Business insights

```
df.info()

df.describe()

df['sales'].value_counts()
```

Q4. ***How did you identify outliers and what did you do with them?***

Answer:

I used the IQR method.
I removed extreme values only after confirming they were data errors.

```
Q1 = df['sales'].quantile(0.25)

Q3 = df['sales'].quantile(0.75)

IQR = Q3 - Q1


df = df[(df['sales'] >= Q1 - 1.5*IQR) &

    (df['sales'] <= Q3 + 1.5*IQR)]
```

3️⃣ Business Logic & KPI Questions

Q5. ***Explain a KPI you calculated using Python.***

Answer:

I calculated **Monthly Revenue Growth** to track performance trends.

```
df['month'] = df['date'].dt.to_period('M')

monthly = df.groupby('month')['revenue'].sum()

growth = monthly.pct_change() * 100
```

Q6. ***How did you segment users or customers in your project?***

Answer:

I used purchase frequency and total spend to segment customers.

```
cust_seg = df.groupby('customer_id').agg(
    total_spend=('amount','sum'),
    order_count=('order_id','count')
)
```

4️⃣ Performance & Optimization Questions

Q7. ***Your dataset was large. How did you optimize Python performance?***

Answer:

I avoided loops and used vectorized operations.
I also filtered data early and reduced memory usage.

```
df = df[['id','amount','date']]

df['amount'] = df['amount'].astype('float32')
```

Q8. ***How did you handle large files in pandas?***

Answer:

I used chunk processing.

chunks = pd.read_csv('data.csv', chunksize=100000)

df = pd.concat(chunks)

5️⃣ SQL + Python Integration Questions

Q9. ***How did you combine SQL and Python in your project?***

Answer:

Heavy joins and aggregations were done in SQL, while data cleaning and EDA were done in Python.

import sqlalchemy as sa

engine = sa.create_engine(conn_str)

df = pd.read_sql("SELECT * FROM sales_summary", engine)

6️⃣ Time-Based & Scenario Questions

Q11. ***How did you analyze trends over time?***

Answer:

I converted dates and created time-based aggregations.

df['date'] = pd.to_datetime(df['date'])

df.set_index('date', inplace=True)

df.resample('M')['sales'].sum()

Q12. ***How did you detect duplicate or suspicious transactions?***

Answer:

I used sorting and time difference logic.

```
df.sort_values(['user_id','timestamp'], inplace=True)

df['time_diff'] = df.groupby('user_id')['timestamp'].diff()

df[df['time_diff'] < pd.Timedelta(minutes=10)]
```