**Pandas exam 2**

**Section A: Data Manipulation**

1. Applying Functions to Columns

df['Price'] = df['Price'].apply(lambda x: x \* 2)

2. Mapping Values in Series

df['Passed'] = df['Passed'].map({'Yes': True, 'No': False})

3. Lowercase Strings

df['Names'] = df['Names'].str.lower()

4. Uppercase Strings

df['City'] = df['City'].str.upper()

5. Splitting Strings

df[['FirstName', 'LastName']] = df['FullName'].str.split(' ', n=1, expand=True)

6. String Contains

gmail\_df = df[df['Email'].str.contains('@gmail.com', na=False)]

7. Replacing String Patterns

df['Email'] = df['Email'].str.replace('@example.com', '@newdomain.com', regex=False)

**Section B: Grouping and Aggregation**

8. Grouping Data

mean\_salary = df.groupby('Department')['Salary'].mean()

9. Aggregating Data

agg\_sales = df['Sales'].agg(['mean', 'max'])

10. Aggregate Multiple Functions

marks\_stats = df['Marks'].agg(['sum', 'count'])

11. Filtering with isin()

filtered\_df = df[df['City'].isin(['New York', 'Los Angeles'])]

12. Grouping and Aggregating

marks\_sum = df.groupby('Gender')['Marks'].sum()

13. Multiple Aggregations on Multiple Columns

agg\_data = df[['Age', 'Salary']].agg(['min', 'max', 'mean'])

14. Grouping and Counting

city\_counts = df.groupby('City').size()

15. Using apply() with Groupby

salary\_range = df.groupby('Department')['Salary'].apply(lambda x: x.max() - x.min())

**Section C: Merging, Joining, Concatenating**

16. Concatenating DataFrames (rows)

combined\_df = pd.concat([df1, df2], axis=0)

17. Merging DataFrames (on ID)

merged\_df = pd.merge(df1, df2, on='ID')

18. Merging with Different Keys

merged\_df = pd.merge(df1, df2, left\_on='EmployeeID', right\_on='ID')

19. Concatenating Along Columns

combined\_df = pd.concat([df1, df2], axis=1)

20. Joining DataFrames with Outer Join

joined\_df = df1.merge(df2, on='ID', how='outer')

**Section D**

21. Transposing DataFrames

transposed\_df = df.transpose()

22. Using T Attribute

transposed\_df = df.T

23. Writing to CSV

df.to\_csv('output.csv', index=False)

24. Writing to Excel

df.to\_excel('output.xlsx', index=False)

25. Writing to JSON

df.to\_json('output.json', orient='records', lines=True)

26. Rendering DataFrame as HTML

df.to\_html('output.html', index=False)

27. Loading CSV File

df = pd.read\_csv('student\_data.csv')

28. Loading Excel File

df = pd.read\_excel('sales\_data.xlsx')

29. Saving a DataFrame as CSV with Selected Columns

df[['Name', 'Salary']].to\_csv('employees.csv', index=False)

30. Saving as JSON with Specific Columns

df[['Name', 'Department']].to\_json('output.json', orient='records', lines=True)