Alternative Advanced Pandas Coding Questions

1. **Given a DataFrame with `User_ID`, `Session_Start`, and `Session_End`, calculate the total duration of each userâ€™s sessions in hours.**

2. **Use `pd.merge_ordered` to merge two DataFrames on a `Date` column, filling missing values using forward-fill (ffill) method for non-overlapping dates.**

3. **Create a DataFrame with multi-level columns `('Sales', 'Q1')`, `('Sales', 'Q2')`, etc. Aggregate sales data across all quarters for each product.**

4. **Apply `pd.to_numeric` with error coercion to a DataFrame where some values in a column are non-numeric strings. Handle the conversion gracefully.**

5. **Perform a time-based split of a DataFrame into training and testing sets using a `Date` column, ensuring that all data before a specific date is used for training.**

6. **Use `pd.DataFrame.aggregate` to perform multiple aggregation functions (e.g., mean, sum, max) on a DataFrame grouped by `Category`.**

7. **Create a DataFrame with `Employee_ID`, `Hours_Worked`, and `Salary`. Apply a custom function that adjusts salaries based on hours worked to compute a new salary.**

8. **Transform a DataFrame with hierarchical indexing (`Year`, `Month`, `Day`) and use `xs` to select data for a specific month across all years.**

9. **Given a DataFrame with `Region` and `Sales`, calculate the cumulative sales for each region over time and plot the results.**

10. **Use `pd.cut` to segment `Revenue` data into quantile-based bins and visualize the distribution of data points within these bins.**

11. **Apply the `pipe` method to chain multiple custom data transformations in a DataFrame, ensuring each stepâ€™s output feeds into the next.**

12. **Detect and handle missing data using `pd.isna()` and `pd.fillna()` to fill gaps with interpolated values in a time series DataFrame.**

13. **Create a DataFrame with `Transaction_ID`, `Amount`, and `Category`. Use `pd.DataFrame.groupby` to compute the proportion of total `Amount` by `Category`.**

14. **Perform a hierarchical merge on two DataFrames with multi-level indexes and resolve conflicts where necessary.**

15. **Resample a time series DataFrame with irregular time intervals to a regular frequency (e.g., daily) using the `resample` method and fill missing values.**

16. **Implement a custom rolling window function that calculates a weighted average over a rolling window in a DataFrame.**

17. **Handle a large dataset by using `pd.HDFStore` to read and write data to/from HDF5 format, and perform a query on the stored data.**

18. **Use `pd.melt` to reshape a DataFrame from wide format to long format and analyze the results.**

19. **Perform data imputation using the K-nearest neighbors (KNN) algorithm on a DataFrame with missing values and compare the imputed results.**

20. **Create a DataFrame with `Transaction_Date`, `Customer_ID`, and `Amount`. Use `groupby` and `agg` to compute the top 5 customers by total transaction amount.**