# *Practice Tasks:*
# Employee Performance and Salary Dataset

Follow these instructions carefully to complete the assignment on the Kaggle notebook linked to the dataset:

## Step 1: Set Up

1. Open the dataset page on Kaggle and click on the **"New Notebook"** button.
2. Load the dataset into the notebook using Pandas. For example:

```
import pandas as pd
df = pd.read_csv('/kaggle/input/employee-performance-and-salary-dataset/dataset.csv')
```

## Step 2: Solving Questions

1. **Sequential Order**: Solve the questions in the order provided, starting from basic operations to advanced topics.
2. **Write Comments**: For each question, write a comment indicating the question number and what you're solving. For example:

```
# Question 1: Load the dataset and display the first 10 rows
print(df.head(10))
```

3. **Code Cleanliness**: Use meaningful variable names and clear formatting in your code.
4. **Output Verification**: Always check and verify your output for correctness. Add comments to explain your results briefly.

## Step 3: Saving Your Work

1. **Organize**: Ensure your notebook is well-organized, with all questions labeled appropriately.
2. **Save the Notebook**: Save your work frequently to prevent data loss. You can do this using Kaggle's auto-save or manually clicking **"Save Version"**.

## Step 4: Submission

1. Once you've completed all questions:
   - Verify all outputs and ensure no errors in the notebook.
   - Add a summary cell at the end of your notebook briefly describing your learning.
2. Save and submit the final notebook for review and share publicly.
   if you face any issue download it as .ipynb and dataset and submit it on google classroom in zip format. (only if you face issue to public it.)

1. Load the dataset into a Pandas DataFrame and display the first 10 rows.
2. What are the column names and their respective data types in the dataset?
3. How many rows and columns are there in the dataset?
4. Select the Name, Age, and Salary columns using df[].
5. Select the first 5 rows of the DataFrame using the iloc method.
6. Select all rows where the Department is "Sales" using the loc method.
7. Select rows where the Salary is greater than 8000 and Status is "Active".
8. Extract the rows where the employee has been with the company for more than 5 years and has a Performance Score of at least 4.
9. Select every alternate row from the DataFrame using slicing with iloc.
10. Retrieve rows where the employee belongs to "Sales" and has a Salary in the top 10% of all salaries.
11. Replace all "Inactive" values in the Status column with "Retired".
12. Add a new column Bonus where values are 10% of the Salary column.
13. Update the Performance Score of employees in the "IT" department to 5 if it is NaN.
14. Modify the Experience of employees with less than 2 years to 2.
15. Normalize the Salary column so that all values are between 0 and 1.
16. Replace all Location values with "Remote" for employees with a Performance Score below 3.
17. Create a new column Experience Level with values "Junior", "Mid", or "Senior" based on the Experience.
18. Add a new row with specific values using the append() method.
19. Delete the column Session from the DataFrame.
20. Delete rows where the Age is greater than 60.
21. Add a new column Age Group with values "Young", "Middle-aged", or "Senior" based on Age.
22. Identify all columns with missing values using the isna() method.
23. Fill the missing values in Performance Score with the average score of that column.
24. Drop all rows with missing values using the dropna() method.
25. Replace missing values in Performance Score with the median of the scores, grouped by Department.
26. Identify rows where the employee is in the "IT" department and their Age is above the average Age of "IT" employees
27. Identify rows with duplicate Name and Joining Date combinations, and remove duplicates.
28. Remove rows where the Experience value is inconsistent with the Age (e.g., Experience > Age - 18).

29. Sort the DataFrame by Salary in descending order.
30. Sort the DataFrame by Department and then by Joining Date in ascending order.
31. Rank all employees by Salary within their Department and add the rank as a new column.
32. Sort the dataset by Experience in descending order, but break ties using Performance Score in ascending order.
33. Create a new DataFrame containing only employees in "HR" with a Salary above 5000.
34. Filter out all rows where the Location is "Chicago".
35. Delete all rows where the Performance Score is below 2.
36. Find the average Salary for each Department.
37. Calculate the total Salary of all employees grouped by Status.
38. Find the maximum Performance Score for employees in each Location.
39. Calculate the total Salary paid to employees who joined in each year.
40. Find the department with the highest average Performance Score.
41. Count the number of employees in each Location grouped by their Session.
42. Group employees by Status and calculate the sum, mean, and standard deviation of their Salary.
43. Use the apply() method to create a new column Monthly Salary by dividing the Salary by 12.
44. Use the applymap() method to convert all string values in the DataFrame to uppercase.
45. Create a second dataset containing Department and Budget. Perform an inner join with the original dataset.
46. Merge the dataset with another dataset that includes additional information about employee ID and their recent Projects.
47. Perform a left join with a dataset containing Department and Team Lead details, adding Team Lead to the DataFrame.
48. Create a DataFrame with employee ID and Project Count, then merge it with the original dataset. Calculate the average Salary per project.
49. Calculate the percentage increase in Salary for each employee compared to the average Salary in their Department.
50. Print “Thanks”