## **Input *(Data)*:** [ASDE Assignment](https://docs.google.com/spreadsheets/d/1lHYAKwfWEONislVTkkHu_zMoAWCOG1MlAt8nPoW0grY/edit#gid=0)

## **Output *(Report)***

Fetch top 3 departments along with their name and average monthly salary. Below is the format of the report.

| **DEPT\_NAME** | **AVG\_MONTHLY\_SALARY (USD)** |
| --- | --- |

**Task-1 SQL**

In the attachment above, use each worksheet as a table in a relational database and write an SQL query that generates the output report

**Task-1 Solution(Using PostGreSQL Syntax)**

SELECT d.NAME AS Dept\_Name, AVG(s.AMT) AS Average\_Monthly\_Salary

FROM DEPARTMENTS d, Salaries s, employees e

where d.ID = e.DEPT\_ID and e.ID = s.EMP\_ID

GROUP BY d.ID, d.NAME

ORDER BY Average\_Monthly\_Salary DESC

LIMIT 3;

**Task-1 Explanation**

The SQL query you provided is used to fetch the top 3 departments along with their average monthly salary. It joins the Departments, Salaries, and employees tables based on the specified conditions (d.ID = e.DEPT\_ID and e.ID = s.EMP\_ID) and calculates the average salary for each department. The result is then sorted in descending order(DESC) of the average monthly salary and limited to the top 3 records using limit clause.

## 

## **Task-2 Scripting**

With the same attachment, use each worksheet as a CSV file and write a script (Bash or Python) that generates the same report. Data is to be read from the CSV files not from a database.

## **Task-2 Solution**

import pandas as pd

# Read CSV files into pandas DataFrames

employees = pd.read\_csv("filePath/EMPLOYEES.csv")

departments = pd.read\_csv("filePath/DEPARTMENTS.csv")

salaries = pd.read\_csv("filePath/SALARIES.csv")

# Merge the DataFrames to get the necessary data

merged\_data = employees.merge(departments, left\_on='DEPT\_ID', right\_on='DEPT\_ID')

merged\_data = merged\_data.merge(salaries, left\_on='EMP\_ID', right\_on='EMP\_ID')

# Calculate average monthly salary for each department

average\_salaries = merged\_data.groupby(['DEPT\_ID', 'DEPT\_NAME'])['AMT (USD)'].mean().reset\_index()

# Sort departments based on average salary in descending order

sorted\_departments = average\_salaries.sort\_values('AMT (USD)', ascending=False)

# Fetch top 3 departments

top\_departments = sorted\_departments.head(3)

# Generate report

report = "Top 3 Departments by Average Monthly Salary:\n"

for index, row in top\_departments.iterrows():

report += "Department Name: {}\n".format(row['DEPT\_NAME'])

report += "Average Monthly Salary: ${}\n".format(row['AMT (USD)'])

report += "-------------------------\n"

# Print the report

print(report)

**Task-2 Explanation**

The above code essentially reads the employee, department, and salary data from CSV files, merges them based on common columns, calculates the average salary for each department, sorts the departments in descending order of average salary, selects the top 3 departments, and generates a report displaying the department name and average monthly salary.

## **Task-3 Debugging**

Given below is a Bash / Python script that performs following computation on an integer input (n):

1. If n is less than 10: Calculate its Square
   1. Example: 4 => 16
2. If n is between 10 and 20: Calculate the factorial of (n-10)
   1. Example: 15 => 120
3. If n is greater than 20: Calculate the sum of all integers between 1 and (n-20)
   1. Example: 25 => 15

The task is to identify the bugs in the script, fix them and share the new script. Only one of the two scripts required Bash or Python. **Hint**: You can correct the script by only changing 3-4 characters.

#### 

#### Script (Python)

| def compute(n):  if n < 10: // Square Calculation  out = n \*\* 2  elif n < 20: // Factorial Calculation  out = 1  for i in range(1, n-10): //last element should be n-10 not n-11  out \*= i  Else: // Sum Calculation  lim = n - 20  out = lim \* lim // instead these 3 lines  out = out - lim // we can use sum function  out = out / 2   print(out)  n = int(input("Enter an integer: ")) compute(n) |
| --- |

#### Correct Script (Python)

| def compute(n):  if n < 10:  out = n \*\* 2  elif n < 20:  out = 1  for i in range(1, n-10 +1): # Fixed the range to include n - 10  out \*= i  else:  lim = n - 20  out = sum(range(1, lim + 1))# Changed the calculation   print(out) # to sum of integers  n = int(input("Enter an integer: ")) compute(n) |
| --- |

**Task-3 Explanation**

1. In the factorial calculation, the range should include n - 10 to ensure the correct factorial calculation.

2. For the case when 'n' is greater than 20, the sum of integers between 1 and '(n-20)' is calculated using the 'sum()' function on the range.

3. The 'lim \* lim' operation in the else block has been removed as it doesn't align with the desired computation.

**Coding Guidelines**

1. Code (Script / SQL) has to be executable and free of any errors.
2. Identify and mention the test cases that need to be covered.
3. Mention any assumptions clearly.
4. Add brief comments to describe the logic/code.
5. Follow best coding practices and structure your code to be modular and readable.
6. Add instructions on how to run the code.
7. For the debugging problem, mention all the edge cases you have tested for and the corresponding fixes.
8. All the 3 tasks are mandatory.
9. The assignment will have to be submitted within 24 hours.
10. This document is read only, create your own files /documents for submissions.