**Task-1**: SQL

To generate the output report using SQL, assuming each worksheet in the attachment represents a table in a relational database, you can write the following SQL query:

**SQL CODE**

SELECT NAME AS DEPT\_NAME, AVG(salary) AS AVG\_MONTHLY\_SALARY

FROM department

JOIN employees ON department.ID = employees.dept\_id

GROUP BY NAME

ORDER BY AVG\_MONTHLY\_SALARY DESC

LIMIT 3;

This query assumes that there is a table called "department" with columns "ID" and "NAME," and another table called "employees" with columns "dept\_id" and "salary." Adjust the table and column names according to the actual structure of your database.

**Task-2**: Scripting

To generate the same report using the CSV files, you can use the following Python script:

**Python Code**

import csv

def generate\_report(csv\_file):

with open(csv\_file, 'r') as file:

reader = csv.DictReader(file)

data = list(reader)

data.sort(key=lambda row: float(row['salary']), reverse=True)

report\_data = data[:3]

report = "DEPT\_NAME\nAVG\_MONTHLY\_SALARY (USD)\n"

for row in report\_data:

report += f"{row['NAME']}\n{row['salary']}\n"

return report

output\_report = generate\_report('employees.csv')

print(output\_report)

This script assumes that you have separate CSV files for each table, such as "department.csv" and "employees.csv." Adjust the CSV file names and column names according to your actual CSV files.

**Task-3**: Debugging

The provided script has several bugs. Here's the corrected version:

def compute(n):

if n < 10:

return n \*\* 2

elif 10 <= n <= 20:

factorial = 1

for i in range(1, n - 9):

factorial \*= i

return factorial

else:

return sum(range(1, n - 19))

# Example usage:

print(compute(4)) # Output: 16

print(compute(15)) # Output: 120

print(compute(25)) # Output: 15

The bugs and their fixes are as follows:

1. Changed the comparison in the second condition from 10 < n < 20 to 10 <= n <= 20.

2. Corrected the factorial calculation by starting the range from 1 instead of 0.

3. Updated the range in the else condition to exclude the upper bound (n - 20) to get the correct sum of integers.

**Note**: The provided fixes assume that the logic of the computation is correct, and only the syntax and comparisons needed correction.