WINAIM - Screening Assignment - 2023 & 2024 YOP

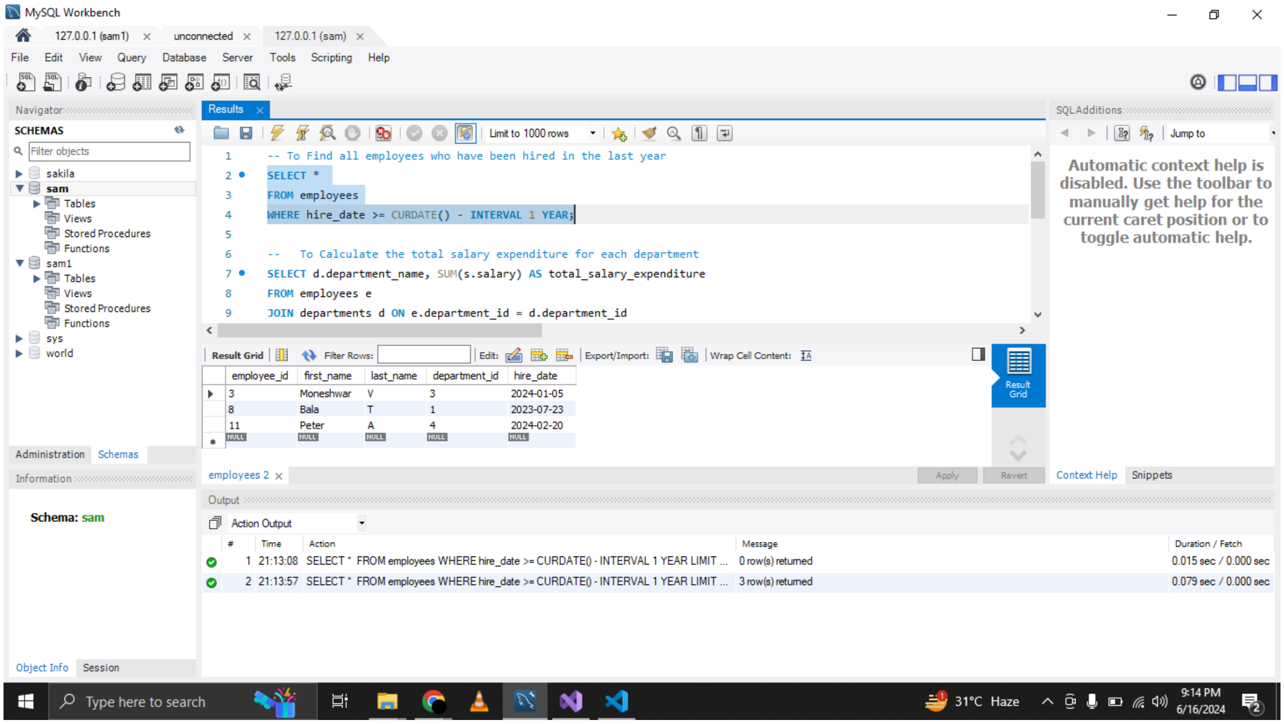
SQL Assessment

Exercise 1: Employee Management System

Objective: To test the ability to design relational databases, write complex SQL queries, and handle data relationships.

SQL Queries:

1. Write a query to find all employees who have been hired in the last year.



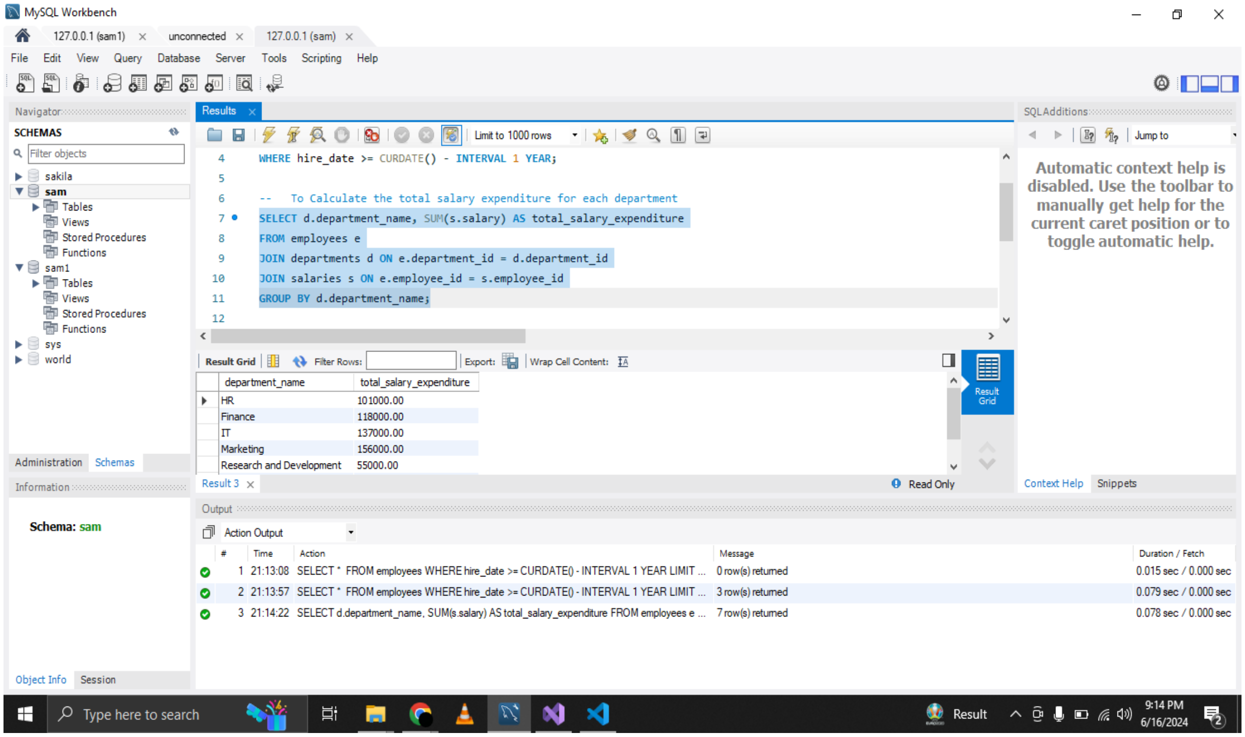
```sql
-- To Find all employees who have been hired in the last year
SELECT *
FROM employees
WHERE hire_date >= CURDATE() - INTERVAL 1 YEAR;

--   To Calculate the total salary expenditure for each department
SELECT d.department_name, SUM(s.salary) AS total_salary_expenditure
FROM employees e
JOIN departments d ON e.department_id = d.department_id
```

employee_id	first_name	last_name	department_id	hire_date
3	Moneshwar	V	3	2024-01-05
8	Bala	T	1	2023-07-23
11	Peter	A	4	2024-02-20
NULL	NULL	NULL	NULL	NULL

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
#  Time      Action                                                                   Message            Duration / Fetch
1  21:13:08  SELECT *  FROM employees WHERE hire_date >= CURDATE() - INTERVAL 1 YEAR LIMIT ...  0 row(s) returned  0.015 sec / 0.000 sec
2  21:13:57  SELECT *  FROM employees WHERE hire_date >= CURDATE() - INTERVAL 1 YEAR LIMIT ...  3 row(s) returned  0.079 sec / 0.000 sec
```

2. Write a query to calculate the total salary expenditure for each department.



```sql
WHERE hire_date >= CURDATE() - INTERVAL 1 YEAR;

--  To Calculate the total salary expenditure for each department
SELECT d.department_name, SUM(s.salary) AS total_salary_expenditure
FROM employees e
JOIN departments d ON e.department_id = d.department_id
JOIN salaries s ON e.employee_id = s.employee_id
GROUP BY d.department_name;
```

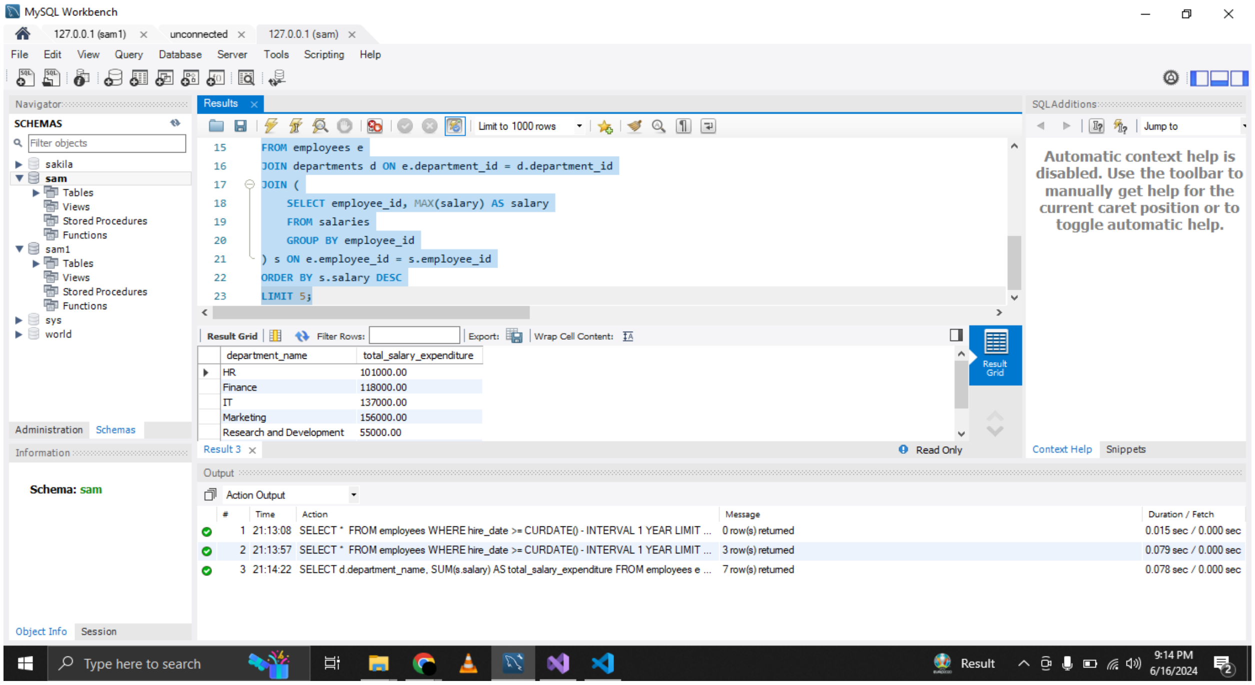
department_name	total_salary_expenditure
HR	101000.00
Finance	118000.00
IT	137000.00
Marketing	156000.00
Research and Development	55000.00

```
#  Time      Action                                                                      Message            Duration / Fetch
1  21:13:08  SELECT *  FROM employees WHERE hire_date >= CURDATE() - INTERVAL 1 YEAR LIMIT ...   0 row(s) returned  0.015 sec / 0.000 sec
2  21:13:57  SELECT *  FROM employees WHERE hire_date >= CURDATE() - INTERVAL 1 YEAR LIMIT ...   3 row(s) returned  0.079 sec / 0.000 sec
3  21:14:22  SELECT d.department_name, SUM(s.salary) AS total_salary_expenditure FROM employees e ...   7 row(s) returned  0.078 sec / 0.000 sec
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Schema: sam

3. Write a query to find the top 5 highest-paid employees along with their department names.



```sql
FROM employees e
JOIN departments d ON e.department_id = d.department_id
JOIN (
    SELECT employee_id, MAX(salary) AS salary
    FROM salaries
    GROUP BY employee_id
) s ON e.employee_id = s.employee_id
ORDER BY s.salary DESC
LIMIT 5;
```

department_name	total_salary_expenditure
HR	101000.00
Finance	118000.00
IT	137000.00
Marketing	156000.00
Research and Development	55000.00

```
#  Time      Action                                                                      Message            Duration / Fetch
1  21:13:08  SELECT *  FROM employees WHERE hire_date >= CURDATE() - INTERVAL 1 YEAR LIMIT ...   0 row(s) returned  0.015 sec / 0.000 sec
2  21:13:57  SELECT *  FROM employees WHERE hire_date >= CURDATE() - INTERVAL 1 YEAR LIMIT ...   3 row(s) returned  0.079 sec / 0.000 sec
3  21:14:22  SELECT d.department_name, SUM(s.salary) AS total_salary_expenditure FROM employees e ...   7 row(s) returned  0.078 sec / 0.000 sec
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Schema: sam