Lab 07

Student Name: **John Okon Ansa**
Student ID: **619952**

---

3. Query Exercises

Write the SQL query that answers each of the following questions.

You got it! I'll add four more questions to the **Basic Selection** section of the lab assignment, focusing on `DISTINCT`, filtering (`WHERE` clause), and ordering (`ORDER BY`).

Here are the four new questions integrated into the lab assignment:

---

3. Query Exercises:

Write the SQL query that answers each of the following questions.

A. Selection:

1. List the **name** and **salary** of all employees.

   SELECT name, salary **FROM** employee

2. Find the names of all **projects** located in **Florida (FL)**.

SELECT project_name

FROM project

WHERE location = 'FL'

3. Retrieve the `emp_id` and `project_id` for employees working on **Project 1**.

SELECT emp_id, project_id

FROM employee_project

WHERE project_id = 1

4. **Find all unique (distinct) states** where employee addresses are located.

SELECT DISTINCT state

FROM address

5. List the **names** and **salaries** of all employees who earn a salary **less than $150,000**.

SELECT name, salary

FROM employee

WHERE salary < 150000

6. List the **project names** and their **estimated days**, ordered from the **longest duration to the shortest**.

SELECT project_name, estimated_days

FROM project

ORDER BY estimated_days **DESC**

7. Find the `emp_id`s of employees who are assigned to a project, listing each `emp_id` only **once**.

SELECT DISTINCT emp_id

FROM employee_project

B. Aggregates and Grouping:

1. Calculate the **average salary** of all employees.

SELECT AVG(salary) **AS** AverageSalary

FROM employee

2. Find the **maximum** `estimated_days` for any single project.

SELECT project_id, **MAX**(estimated_days) **As** Maximium

FROM project

GROUP BY project_id;

3. For each **department**, report the `dept_id` and the **total salary** expenditure.

SELECT dept_id, **SUM**(salary) **AS** TotalSalary

FROM employee

GROUP BY dept_id;

4. Find the `dept_id` of departments that have an **average employee salary greater than $150,000**.

SELECT dept_id

FROM employee

GROUP BY dept_id

HAVING AVG(salary) > 150000;

C. Joins:

1. List the **employee name** and the **city** where they live. (Join `Employee` and `Address`).

SELECT e.name **AS** EmployeeName, a.city

FROM employee e

INNER JOIN address a **ON** e.address_id = a.address_id;

2. List **all departments** and the **names** of the employees who belong to them. Include departments that may not currently have any employees. (Join `Department` and `Employee`). *Note: Based on the sample data, all departments have employees, but this query structure is key for future scenarios.*

SELECT d.name **AS** DepartmentName, e.name **AS** EmployeeName

FROM department d

LEFT JOIN employee e **ON** d.dept_id = e.dept_id;

3. Find the **employee name** and the **name of the projects** they are working on. (Join `Employee`, `Employee_Project`, and `Project`).

   SELECT e.name **AS** employee_name, p.project_name

   FROM employee e

   JOIN employee_project ep **ON** e.emp_id = ep.emp_id

JOIN project p **ON** ep.project_id = p.project_id;

D. Subqueries

1. Find the **name** of the employee who has the **highest salary** (Use a subquery in the `WHERE` clause).

SELECT name

FROM employee

WHERE salary = (**SELECT MAX**(salary) **FROM** employee);

2. List the **names** of employees who work on a project that has an `estimated_days` of **180** (Use an `IN` or `EXISTS` subquery).

SELECT e.name

FROM employee e

WHERE e.emp_id **IN** (

SELECT ep.emp_id

FROM employee_Project ep

JOIN project p ON ep.project_id = p.project_id

WHERE p.estimated_days = 180

);

3. Find the `project_id` of all projects that have an **estimated duration greater than the average estimated duration** of all projects (Use a subquery in the `WHERE` clause).

SELECT project_id

FROM project

WHERE estimated_days > (**SELECT AVG**(estimated_days) **FROM** project);

```
-- *** 1. Insert Data into Address Table ***
INSERT INTO Address (address_id, city, state, zipcode) VALUES
(1, 'Fairfield', 'IA', '52556'),
(2, 'Iowa City', 'IA', '52440'),
(3, 'Morrison', 'IL', '61270'),
(4, 'Orlando', 'FL', '34565'),
(5, 'Tampa', 'FL', '31765');

-- *** 2. Insert Data into Department Table ***
INSERT INTO Department (dept_id, name) VALUES
(1, 'Tech'),
(2, 'HR'),
(3, 'Finance'),
(4, 'Marketing');

-- *** 3. Insert Data into Project Table ***
INSERT INTO Project (project_id, project_name, estimated_days, location)
VALUES
(1, 'X', 180, 'FL'),
(2, 'Y', 60, 'FL'),
(3, 'Z', 80, 'IA');

-- *** 4. Insert Data into Employee Table ***
-- NOTE: This depends on Address (address_id) and Department (dept_id) being
populated first.
INSERT INTO Employee (emp_id, name, salary, address_id, dept_id) VALUES
(111, 'Zaineh', 100000, 1, 1),
(112, 'Yasmeen', 160000, 2, 4),
(113, 'Mira', 140000, 3, 3),
(114, 'Shimaa', 200000, 4, 2),
(115, 'Dean', 150000, 5, 1);

-- *** 5. Insert Data into Employee_Project Table ***
-- NOTE: This depends on Employee (emp_id) and Project (project_id) being
populated first.
INSERT INTO Employee_Project (emp_id, project_id) VALUES
(115, 1),
(115, 2),
(115, 3),
(114, 1),
(114, 3),
(111, 1),
(111, 2);
```