Lab 2 Assignment

Mohamed Emary

February 8, 2025

1 Assignment Answers

1.1 Question 1

> Write a query that gets the date of the third day in the next month. Print it in the format like `14-December-2020, Saturday`.

```sql
SELECT TO_CHAR(
    DATE_TRUNC('month', NOW()) + INTERVAL '1 month' + INTERVAL '2 days',
    'dd-Month-yyyy, Day'
);
```

1.2 Question 2

> Write a query that gets the last day date of the current month from today. Print it in the format like `14-December-2020, Saturday`.

```sql
SELECT TO_CHAR(
    (DATE_TRUNC('month', NOW()) + INTERVAL '1 month' - INTERVAL '1 day'),
    'dd-Month-yyyy, Day'
);
```

1.3 Question 3

> Display the employee's name, hire date, and salary review date. The salary review date is the day after six months and five days of service. Label the column as `Review`. Format the dates to appear in the format similar to `Sunday, the 7th of September, 1981`.

```sql
SELECT
    last_name,
    TO_CHAR(hire_date, 'Day, the DDth of Month, YYYY') as hire_date,
    TO_CHAR(hire_date + INTERVAL '6 months' + INTERVAL '6 days', 'Day, the
    ↪  DDth of Month, YYYY') as review
FROM employees;
```

1.4 Question 4

> Write a query that will display the difference between the highest and lowest salaries in each department.

```
SELECT
    department_id,
    MAX(salary) - MIN(salary) as salary_difference
FROM employees
WHERE department_id IS NOT NULL
GROUP BY department_id
ORDER BY department_id;
```

1.5 Question 5

> Write a query that will display the city, department name, number of employees, and the average salary for all employees in that department. Round the average salary to two decimal places.

```
SELECT
    l.city,
    d.department_name,
    COUNT(e.employee_id) as employee_count,
    ROUND(AVG(e.salary), 2) as avg_salary
FROM departments d
    INNER JOIN locations l ON d.location_id = l.location_id
    LEFT JOIN employees e ON d.department_id = e.department_id
GROUP BY l.city, d.department_name
ORDER BY l.city, d.department_name;
```

1.6 Question 6

> Display the employee number, name, and salary for all employees who earn more than the average salary in their department.

```
SELECT
    e.employee_id,
    e.last_name,
    e.salary
FROM employees e
WHERE salary > (
    SELECT AVG(salary)
    FROM employees inner_emp
    WHERE inner_emp.department_id = e.department_id
)
ORDER BY employee_id;
```

1.7 Question 7

> Show employees' data whose salary is higher than their manager's and show the manager's name and salary (use subquery, not join).

```
SELECT
    e.employee_id,
    e.last_name,
    e.salary,
    (SELECT last_name FROM employees WHERE employee_id = e.manager_id) as
↪   manager_name,
    (SELECT salary FROM employees WHERE employee_id = e.manager_id) as
↪   manager_salary
FROM employees e
WHERE e.salary > (
    SELECT salary
    FROM employees
    WHERE employee_id = e.manager_id
);
```

1.8 Question 8

> Show employees' data who earn the lowest salary in their department (use subquery, not join).

```
SELECT *
FROM employees e
WHERE salary = (
    SELECT MIN(salary)
    FROM employees
    WHERE department_id = e.department_id
);
```

1.9 Question 9

> Find employees who have been hired earlier than anyone else in the same job (use subquery, not join).

```
SELECT *
FROM employees e
WHERE hire_date = (
    SELECT MIN(hire_date)
    FROM employees
    WHERE job_id = e.job_id
);
```

1.10 Question 10

> Write a query to display `employee_id`, `last_name`, `salary`, `dept_id`, `dept_name`, `job_id`, `job_title`, `city`, `street_address`, `country_id`, `country_name`, `region_id`, `region_name` for all employees, including those employees who have no department too.

```sql
SELECT
    e.employee_id,
    e.last_name,
    e.salary,
    d.department_id as dept_id,
    d.department_name as dept_name,
    j.job_id,
    j.job_title,
    l.city,
    l.street_address,
    c.country_id,
    c.country_name,
    r.region_id,
    r.region_name
FROM employees e
    LEFT JOIN departments d ON e.department_id = d.department_id
    LEFT JOIN jobs j ON e.job_id = j.job_id
    LEFT JOIN locations l ON d.location_id = l.location_id
    LEFT JOIN countries c ON l.country_id = c.country_id
    LEFT JOIN regions r ON c.region_id = r.region_id
ORDER BY e.employee_id;
```