Laboratory Work #3 - DML Operations

Objective: Practice advanced SQL DML operations based on lecture concepts. Save all queries as `lab3_advanced_dml.sql`.

Part A: Database and Table Setup

1. Create database and tables

```
-- Create database called 'advanced_lab'
-- Create table 'employees' with columns: emp_id (primary key, auto increment
),
-- first_name (string), last_name (string), department (string),
-- salary (integer), hire_date (date), status (string with default 'Active')

-- Create table 'departments' with columns: dept_id (primary key, auto increm
ent),
-- dept_name (string), budget (integer), manager_id (integer)

-- Create table 'projects' with columns: project_id (primary key, auto increm
ent),
-- project_name (string), dept_id (integer), start_date (date),
-- end_date (date), budget (integer)
```

Part B: Advanced INSERT Operations

2. INSERT with column specification

Insert data into employees table specifying only certain columns (emp_id, first_name, last_name, department).

3. INSERT with DEFAULT values

Insert a row into employees where salary uses DEFAULT value and status uses DEFAULT value.

4. INSERT multiple rows in single statement

Insert 3 departments using one INSERT statement with multiple VALUES clauses.

5. INSERT with expressions

Insert an employee where hire_date is calculated as current date and salary is calculated as 50000 * 1.1.

6. INSERT from SELECT (subquery)

Create a temporary table 'temp_employees' and insert data from employees table where department equals 'IT'.

Part C: Complex UPDATE Operations

7. UPDATE with arithmetic expressions

Increase all employee salaries by 10% using UPDATE with multiplication.

8. UPDATE with WHERE clause and multiple conditions

Update employee status to 'Senior' where salary is greater than 60000 AND hire_date is before '2020-01-01'.

9. UPDATE using CASE expression

Update employee department using CASE: - If salary > 80000 then department = 'Management' - If salary between 50000 and 80000 then department = 'Senior' - Else department = 'Junior'

10. UPDATE with DEFAULT

Set department to DEFAULT value for employees where status equals 'Inactive'.

11. UPDATE with subquery

Update department budget to be 20% higher than the average salary of employees in that department.

12. UPDATE multiple columns

Update employees set salary = salary * 1.15 and status = 'Promoted' where department = 'Sales' in single statement.

Part D: Advanced DELETE Operations

13. DELETE with simple WHERE condition

Delete all employees where status equals 'Terminated'.

14. DELETE with complex WHERE clause

Delete employees where salary < 40000 AND hire_date > '2023-01-01' AND department IS NULL.

15. DELETE with subquery

Delete departments where dept_id NOT IN (SELECT DISTINCT department FROM employees WHERE department IS NOT NULL).

16. DELETE with RETURNING clause

Delete all projects where end_date < '2023-01-01' and return all deleted data.

Part E: Operations with NULL Values

17. INSERT with NULL values

Insert employee with NULL salary and NULL department.

18. UPDATE NULL handling

Update all employees set department = 'Unassigned' where department IS NULL.

19. DELETE with NULL conditions

Delete all employees where salary IS NULL OR department IS NULL.

Part F: RETURNING Clause Operations

20. INSERT with RETURNING

Insert new employee and return the auto-generated emp_id and full name (concatenated).

21. UPDATE with RETURNING

Update salary for employees in 'IT' department (increase by 5000) and return emp_id, old salary, and new salary.

22. DELETE with RETURNING all columns

Delete employees where hire_date < '2020-01-01' and return all columns of deleted rows.

Part G: Advanced DML Patterns

23. Conditional INSERT

Write INSERT that only adds employee if no employee with same first_name and last_name already exists (use WHERE NOT EXISTS).

24. UPDATE with JOIN logic using subqueries

Update employee salaries based on department budget: if department budget > 100000, increase salary by 10%, otherwise by 5%.

25. Bulk operations

Insert 5 employees in single statement, then update all their salaries to be 10% higher in single UPDATE.

26. Data migration simulation

Create new table 'employee_archive', move all employees with status 'Inactive' from employees to employee_archive, then delete them from original table.

27. Complex business logic

Update project end_date to be 30 days later for projects where budget > 50000 AND associated department has more than 3 employees.

Submission Requirements

1. Create all tables with proper data types and constraints
2. Insert sample data to test your queries
3. Include comments explaining complex operations
4. Test each query to ensure it works correctly
5. Use proper SQL formatting and indentation

Notes

- Focus on concepts covered in lectures: INSERT variations, UPDATE with expressions, DELETE with conditions, RETURNING clause
- Ensure proper handling of NULL values and DEFAULT values
- Use arithmetic expressions and conditional logic where specified
- Test queries with sample data to verify functionality