Lab Report

Course Title: Database Management Sessional
Course Code: CSE 2206

Submitted by:
Name: Md. Khairul Islam
ID: CSE2302029034
Section: 29A1

Supervised by:
Md. Rashedul Islam
Lecturer:
Department of Computer Science & Engineering
Sonargaon University

Submission Date: 13 Aypril 2025

Name: Md. Khairul Islam
ID: CSE2302029034

Lab SQL Query

1. Creating Tables

CARS Table:
```
CREATE TABLE cars (
    md_num number(3),
    md_name varchar(10),
    style varchar(10),
    year number(3)
);
```

SPECS Table:
```
CREATE TABLE specs(
    md_num number(3),
    mpg number(3),
    radio varchar(10),
);
```

STOCK Table:
```
CREATE TABLE stock(
    md_num number(3),
    qty number(3),
    price number(3)
);
```

2. Viewing Existing Tables:

```
SELECT * FROM tab;
```

3. Dropping Tables:

```
DROP TABLE cars;
DROP TABLE specs;
DROP TABLE stock;
```

4. Modifying Tables Using ALTER

* Adding a single Column:

```
ALTER TABLE specs ADD tyree VARCHAR(10);
```

* Adding Multiple Columns:

```
~~ALTER Multiple Columns~~
ALTER TABLE cars ADD(
    company VARCHAR(10),
    supplier VARCHAR(10),
);
```

* Modifying Column Types:

```
ALTER TABLE specs MODIFY tyre NUMBER(3);
ALTER TABLE cars MODIFY (
    company VARCHAR(20),
    supplier VARCHAR(20),
);
```

* Dropping a Column:

```
ALTER TABLE specs DROP COLUMN tyree;
```

* Renaming a Column:

```
ALTER TABLE cars RENAME COLUMN company
TO manufacturers;
```

5. Inserting Data

```
INSERT INTO table_name(column1, column2, ...)
VALUES (value1, value2, ...);
```

6. Viewing Data

```
SELECT * FROM cars;
SELECT * FROM specs;
SELECT * FROM stock;
```

Name: Md. Khairul Islam
ID: CSE2302029034

7. Updating Records:

```
UPDATE cars SET year = 2007 WHERE
md_name = 'Ford';
```

8. Deleting Records:

```
DELETE FROM cars WHERE md_name = 'Ford';
```

9. Creating Tables with Keys

* Creating Keys During Table Creation:

```
CREATE TABLE employee(
    ssn NUMBER PRIMARY KEY,
    fname VARCHAR(20),
    lname VARCHAR(20),
    dno NUMBER,
    FOREIGN KEY (dno) REFERENCES department
    (dnumber));
```

* Creating Keys After Table Creation:

```
ALTER TABLE employee ADD CONSTRANT
Pk_emp PRIMARY KEY(ssn),
ALTER TABLE employee ADD CONSTRAINT
Pk_dno FOREIGN KEY(dno) REFERENCES
department(dnumber);
```

10. Key Constraints Behaivor:

* Attempting to Delete with Active Constraints

```
DELETE FROM department WHERE dnumber=5;
DELETE FROM employee WHERE dno=5;
```

11. ON DELETE CASCADE:

```
FOREIGN KEY(dno) REFERENCES department
(dnumber) on DELETE CASCADE
```

12. UNIQUE KEY Constraint:

```
CREATE TABLE course(
course_name VARCHAR2(50) UNIQUE);
```

13. CHECK and DEFAULT Constraints:

```
CREATE TABLE course (
course_id NUMBER,
credit NUMBER CHECK (credit BETWEEN 1
AND 4),
pass_mark NUMBER DEFAULT 40);
```

14. SQL JOIN Types Demonstrated:

* Basic Join(using WHERE clause):

```
SELECT d.dept_id, d_name, l.reginal_group
FROM location l, department d
WHERE l.location_id = d.location_id;
```

15. INNER JOIN:

```
SELECT d.dept_id, d.name, l.regional_group
FROM department d JOIN location l
ON d.location_id = l.location_id;
```

16. JOIN with USING clause:

```
SELECT d.dept_id, d.name, l.regional_group
FROM department d JOIN location l
USING (location_id);
```

17. NATURAL JOIN:

```
SELECT d.name, l.regional_group
FROM department d NATURAL JOIN location l;
```

* Join with Multiple Conditions:

```
SELECT *
FROM A JOIN B
ON A.c1 = B.c1 AND A.c2 = B.c2;
```

* Or using:

```
USING (c1, c2)
```

Name: Md. Khairul Islam
ID: CSE2302029034

* Cross Join (Cartesian Product):

```
SELECT e.lname, d.name
FROM employee e CROSS JOIN department d;
```

* Outer Joins:

  * Left Outer Join:

```
SELECT d.dept_id, d.name, l.regional_group
FROM department d LEFT OUTER JOIN location
l
ON d.location_id = location_id;
```

  * RIGHT Outer Join:

```
SELECT d.dept_id, d.name, l.regional_group
FROM department d RIGHT OUTER JOIN
location l
ON d.location_id = l.location_id;
```

  * Full Outer Join:

```
SELECT d.dept_id, d.name, l.regional_group
FROM department d FULL OUTER JOIN location1
ON d.location_id = l.location_id;
```

  * Equi-Join vs. Non-Equi Join:

* Equi-Join:

```
SELECT s.name AS supplier_name, p.name AS
part_name
FROM supplier s JOIN part p
ON s.supplier_id = p.supplier_id;
```

* Non-Equi Join:

```
SELECT p.name AS part_name, c.inv_class AS inv_class
FROM part p JOIN inventory_class c
ON p.unit_cost BETWEEN c.low_cost AND
c.high_cost;
```

* Self Join:

  * Inner Self Join:

```
SELECT e.lname AS employee, m.lname AS
manager
FROM employee e JOIN employee m
ON e.manager_emp_id = m.emp_id;
```

  * Outer Self Join:

```
SELECT e.lname AS employee, m.lname AS
manager
FROM employee e LEFT OUTER JOIN employee m
ON e.manager_emp_id = m.emp_id;
```