Day 1 -SQL : Topics covered

Domain Driven Design:

Educational Application

Student, Teacher, Course, Department, Subject

Banking Application

Account, Customer, Fund Transfer …

Retail Application

Product, Order, Cart, Payment

- Practised a simple query of creating student tables:

```
-- create
CREATE TABLE students(
  student_id INT PRIMARY KEY,
  name varchar(100),
  course varchar(100),
  join_date DATE
);
```

```sql
-- insert
INSERT INTO students VALUES
(1, 'kamali', 'Data Engineering', '2025-07-17'),
(2, 'Amirta', 'Data Science', '2025-07-16'),
(3, 'Ramya', 'Data Engineering', '2025-07-15');


-- fetch
SELECT * FROM Students;


SELECT name,course FROM students;


SELECT * FROM students WHERE course = 'Data Engineering';


SELECT * FROM students WHERE join_date > '2025-07-15';


SELECT * FROM students
WHERE course = 'Data Engineering' AND join_date > '2025-07-15';


SELECT * FROM students
WHERE course IN ('Data Science', 'AI');


SELECT * FROM students
WHERE join_date BETWEEN '2025-07-17' AND '2025-07-16';
```

Matching patterns:

```
SELECT * FROM students WHERE name LIKE 'A%';


SELECT * FROM students WHERE name LIKE '%a';


SELECT * FROM students WHERE name LIKE '%it%';
```

Update:

```
UPDATE students
SET course = 'Advanced Data Engineering'
WHERE student_id = 1;


UPDATE students
SET join_date = '2025-09-18'
WHERE name = 'Radha'
```

Delete:

```
DELETE FROM students
WHERE student_id = 2;


DELETE FROM students
WHERE join_date < '2025-09-18';
```

SQL Assignment: products Table

Task 1: Create a Table

```
CREATE TABLE products (
    product_id INT PRIMARY KEY,
    product_name VARCHAR(100),
    category VARCHAR(50),
    price DECIMAL(10,2),
    stock_quantity INT,
    added_date DATE
);
```

Task 2: Insert Records

```
INSERT INTO products (product_id, product_name, category, price, stock_quantity,
added_date) VALUES
(1, 'Wireless Headphones', 'Electronics', 2499.99, 25, '2025-07-15'),
(2, 'Ergonomic Mouse', 'Accessories', 799.50, 100, '2025-07-14'),
(3, 'Gaming Laptop', 'Computers', 75999.00, 10, '2025-07-10'),
(4, 'Smartwatch', 'Wearables', 11999.49, 35, '2025-07-12'),
(5, 'Office Chair', 'Furniture', 5499.00, 15, '2025-07-11');
```

Task 3: Write Queries

- 1. List all products.

```
SELECT * FROM products;
```

- 2. Display only product_name and price .

```
SELECT product_name, price FROM products;
```

- 3. Find products with stock_quantity less than 10.

```
SELECT * FROM products
WHERE stock_quantity < 10;
```

- 4. Find products with price between 500 and 2000.

```
SELECT * FROM products
WHERE price BETWEEN 500 AND 2000;
```

- 5. Show products added after 2023-01-01 .

```
SELECT * FROM products
WHERE added_date > '2023-01-01';
```

- 6. List all products whose names start with ‘S’.

```
SELECT * FROM products
WHERE product_name LIKE 'S%';
```

- 7. Show all products that belong to either Electronics or Furniture .

```
SELECT * FROM products
WHERE category IN ('Electronics', 'Furniture');
```

Task 4: Update & Delete

- 1. Update the price of one product.

```
UPDATE products
SET price = 899.50
WHERE product_id = 2;
```

- 2. Increase stock of all products in a specific category by 5.

```
UPDATE products
SET stock_quantity = stock_quantity + 5
WHERE category = 'Electronics';
```

- 3. Delete one product based on its product_id .

```
DELETE FROM products
WHERE product_id = 5;
```

- 4. Delete all products with stock_quantity = 0.

```
DELETE FROM products
WHERE stock_quantity = 0;
```

TASK 2:JOIN BASED QUESTION

```
CREATE TABLE departments (
   dept_id INT PRIMARY KEY,
   dept_name VARCHAR(100)
);

CREATE TABLE employees (
   emp_id INT PRIMARY KEY,
   emp_name VARCHAR(100),
   dept_id INT,
   salary INT);

INSERT INTO departments VALUES
(1, 'Human Resources'),
(2, 'Engineering'),
(3, 'Marketing');

INSERT INTO employees VALUES
(101, 'Amit Sharma', 1, 30000),
(102, 'Neha Reddy', 2, 45000),
(103, 'Faizan Ali', 2, 48000),
(104, 'Divya Mehta', 3, 35000),
(105, 'Ravi Verma', NULL, 28000);

SELECT e.emp_name, d.dept_name
FROM employees e
LEFT JOIN departments d ON e.dept_id = d.dept_id;

SELECT emp_name
```

```
FROM employees
WHERE dept_id IS NULL;

SELECT d.dept_name, COUNT(e.emp_id) AS total_employees
FROM departments d
LEFT JOIN employees e ON d.dept_id = e.dept_id
GROUP BY d.dept_name;

SELECT d.dept_name
FROM departments d
LEFT JOIN employees e ON d.dept_id = e.dept_id
WHERE e.emp_id IS NULL;

SELECT e.emp_name, d.dept_name, e.salary
FROM employees e
JOIN departments d ON e.dept_id = d.dept_id
WHERE e.salary > 40000;
```