**Exercise 1 - SELECT Statement:**

**Scenario:**

**Create a table named employees with columns employee\_id, first\_name, last\_name, and salary. Insert 30 rows of sample data into the table. Write a query to select the first and last names of all employees.**

Solution:

-- Create employees table

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

salary DECIMAL(10, 2)

);

-- Insert 30 rows of sample data into the employees table

INSERT INTO employees (first\_name, last\_name, salary)

VALUES

('John', 'Doe', 60000.00),

('Jane', 'Smith', 55000.00),

('Michael', 'Johnson', 70000.00),

('Susan', 'Williams', 62000.00),

('Eva', 'Garcia', 59000.00),

('David', 'Brown', 64000.00),

('Emily', 'Miller', 61000.00),

('Daniel', 'Wilson', 68000.00),

('Olivia', 'Jones', 54000.00),

('William', 'Anderson', 58000.00),

('Sophia', 'Martinez', 65000.00),

('James', 'Taylor', 67000.00),

('Emma', 'Moore', 70000.00),

('Benjamin', 'Jackson', 59000.00),

('Ava', 'Lee', 62000.00),

('Mason', 'Gonzalez', 54000.00),

('Isabella', 'Rodriguez', 63000.00),

('Liam', 'Perez', 56000.00),

('Mia', 'Hernandez', 61000.00),

('Oliver', 'Lopez', 69000.00),

('Amelia', 'Garcia', 70000.00),

('Ethan', 'Martinez', 66000.00),

('Harper', 'Davis', 57000.00),

('Aiden', 'Rodriguez', 61000.00),

('Luna', 'Hernandez', 62000.00),

('Noah', 'Lopez', 59000.00),

('Layla', 'Gonzalez', 54000.00),

('Logan', 'Perez', 61000.00),

('Evelyn', 'Lee', 67000.00),

('Lucas', 'Jackson', 60000.00);

-- Query to select first and last names of all employees

SELECT first\_name, last\_name

FROM employees;

**Exercise 2 - INSERT Statement:**

**Scenario:**

**Create a table named students with columns student\_id, first\_name, last\_name, and gpa. Insert a new student with the following details: First Name - "Michael", Last Name - "Johnson", GPA - 3.85.**

Solution:

-- Create students table

CREATE TABLE students (

student\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

gpa DECIMAL(3, 2)

);

-- Insert a new student

INSERT INTO students (first\_name, last\_name, gpa)

VALUES ('Michael', 'Johnson', 3.85);

**Exercise 3 - UPDATE Statement:**

**Scenario:**

**You have a table named products with columns product\_id, product\_name, price, and stock\_quantity. Update the price of the product with product\_id 5 to $19.99.**

Solution:

-- Create products table

CREATE TABLE products (

product\_id INT AUTO\_INCREMENT PRIMARY KEY,

product\_name VARCHAR(100),

price DECIMAL(10, 2),

stock\_quantity INT

);

-- Insert 10 rows of sample data into the products table

INSERT INTO products (product\_name, price, stock\_quantity)

VALUES

('Widget', 10.99, 100),

('Gadget', 19.99, 75),

('Doodad', 7.49, 120),

('Thingamajig', 14.50, 90),

('Whatchamacallit', 8.75, 150),

('Doohickey', 6.25, 200),

('Contraption', 12.00, 50),

('Gizmo', 9.99, 80),

('Widgetron', 29.95, 30),

('Thingummy', 5.50, 110);

-- Update the price of product with product\_id 5

UPDATE products

SET price = 19.99

WHERE product\_id = 5;

**Exercise 4 - DELETE Statement:**

**Scenario:**

**Create a table named orders with columns order\_id, customer\_id, order\_date, and status. Delete all orders with a status of "Cancelled".**

Solution:

-- Create orders table

CREATE TABLE orders (

order\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT,

order\_date DATE,

status VARCHAR(50)

);

-- Insert 10 rows of sample data into the orders table

INSERT INTO orders (customer\_id, order\_date, status)

VALUES

(1, '2023-08-01', 'Pending'),

(2, '2023-08-02', 'Shipped'),

(3, '2023-08-03', 'Processing'),

(4, '2023-08-03', 'Shipped'),

(1, '2023-08-04', 'Pending'),

(5, '2023-08-05', 'Cancelled'),

(2, '2023-08-06', 'Shipped'),

(3, '2023-08-07', 'Processing'),

(4, '2023-08-08', 'Shipped'),

(5, '2023-08-09', 'Cancelled');

-- Delete orders with status "Cancelled"

DELETE FROM orders

WHERE status = 'Cancelled';

**Exercise 5 - SELECT with JOIN Statement:**

**Scenario:**

**Create two tables named orders1 and customers with columns order\_id, customer\_id, and order\_date in the orders table, and customer\_id, first\_name, and last\_name in the customers table.**

**Insert data into both tables**

**Write a query to retrieve the first and last names of customers who placed orders.**

-- Create customers table

CREATE TABLE customers (

customer\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50)

);

-- Create orders table

CREATE TABLE orders (

order\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT,

order\_date DATE

);

-- Insert data into customers table

INSERT INTO customers (first\_name, last\_name)

VALUES

('John', 'Doe'),

('Jane', 'Smith'),

('Michael', 'Johnson'),

('Susan', 'Williams'),

('Eva', 'Garcia');

-- Insert data into orders table

INSERT INTO orders (customer\_id, order\_date)

VALUES

(1, '2023-08-01'),

(2, '2023-08-02'),

(3, '2023-08-03'),

(4, '2023-08-04'),

(5, '2023-08-05');

-- Query to retrieve first and last names of customers who placed orders

SELECT c.first\_name, c.last\_name

FROM customers c

JOIN orders o ON c.customer\_id = o.customer\_id;

**Exercise 6 - INSERT...SELECT Statement:**

**Scenario:**

**Create two tables named source\_data and target\_data with columns value in both tables. Insert some sample data into the source\_data table. Write an INSERT...SELECT statement to copy data from source\_data to target\_data.**

-- Create source\_data table

CREATE TABLE source\_data (

value INT

);

-- Create target\_data table

CREATE TABLE target\_data (

value INT

);

-- Insert data into source\_data table

INSERT INTO source\_data (value)

VALUES (10), (20), (30), (40), (50);

-- Copy data from source\_data to target\_data

INSERT INTO target\_data (value)

SELECT value FROM source\_data;

**Exercise 7 - UPDATE with JOIN Statement:**

**Scenario:**

**Create two tables named employees and departments with columns employee\_id, first\_name, last\_name, department\_id in the employees table, and department\_id and department\_name in the departments table. Insert data into both tables and write a query to update the department of an employee based on the department name.**

-- Create departments table

CREATE TABLE departments (

department\_id INT AUTO\_INCREMENT PRIMARY KEY,

department\_name VARCHAR(100)

);

-- Create employees table

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

department\_id INT

);

-- Insert data into departments table

INSERT INTO departments (department\_name)

VALUES

('HR'),

('IT'),

('Finance');

-- Insert data into employees table

INSERT INTO employees (first\_name, last\_name, department\_id)

VALUES

('John', 'Doe', 1),

('Jane', 'Smith', 2),

('Michael', 'Johnson', 1),

('Susan', 'Williams', 3),

('Eva', 'Garcia', 2);

-- Update department of an employee based on department name

UPDATE employees e

JOIN departments d ON e.department\_id = d.department\_id

SET e.department\_id = 2

WHERE d.department\_name = 'IT';

**Exercise 8 - DELETE with JOIN Statement:**

**Scenario:**

**Create two tables named customers and orders with columns customer\_id, first\_name, last\_name in the customers table, and order\_id, customer\_id, and order\_date in the orders table. Insert data into both tables and write a query to delete all orders placed by customers whose last name is "Smith".**

-- Create customers table

CREATE TABLE customers (

customer\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50)

);

-- Create orders table

CREATE TABLE orders (

order\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT,

order\_date DATE

);

-- Insert data into customers table

INSERT INTO customers (first\_name, last\_name)

VALUES

('John', 'Doe'),

('Jane', 'Smith'),

('Michael', 'Johnson'),

('Susan', 'Williams'),

('Eva', 'Garcia');

-- Insert data into orders table

INSERT INTO orders (customer\_id, order\_date)

VALUES

(1, '2023-08-01'),

(2, '2023-08-02'),

(3, '2023-08-03'),

(2, '2023-08-04'),

(4, '2023-08-05');

-- Delete orders placed by customers with last name "Smith"

DELETE o

FROM orders o

JOIN customers c ON o.customer\_id = c.customer\_id

WHERE c.last\_name = 'Smith';

**Exercise 9 - INSERT...ON DUPLICATE KEY UPDATE Statement:**

**Scenario:**

**Create a table named users with columns user\_id, username, and email. Insert some sample data into the table. Write an INSERT...ON DUPLICATE KEY UPDATE statement to insert a new user or update the email address if the username already exists.**

-- Create users table

CREATE TABLE users (

user\_id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(50),

email VARCHAR(100)

);

-- Insert sample data into users table

INSERT INTO users (username, email)

VALUES ('john\_doe', 'john@example.com'),

('jane\_smith', 'jane@example.com'),

('mike\_jones', 'mike@example.com');

-- Insert a new user or update email if username exists

INSERT INTO users (username, email)

VALUES ('john\_doe', 'john.doe.updated@example.com')

ON DUPLICATE KEY UPDATE email = VALUES(email);

**Exercise 10 - REPLACE Statement:**

**Scenario:**

**Create a table named products with columns product\_id, product\_name, price. Insert some sample data into the table. Write a REPLACE statement to insert a new product or replace the existing product if the product\_id already exists.**

-- Create products table

CREATE TABLE products (

product\_id INT AUTO\_INCREMENT PRIMARY KEY,

product\_name VARCHAR(100),

price DECIMAL(10, 2)

);

-- Insert sample data into products table

INSERT INTO products (product\_name, price)

VALUES ('Widget', 10.99),

('Gadget', 19.99),

('Doodad', 7.49);

-- Insert a new product or replace existing product if product\_id exists

REPLACE INTO products (product\_id, product\_name, price)

VALUES (2, 'Updated Gadget', 24.99);

**Exercise 11 - ALTER TABLE Statement:**

**Scenario:**

**Create a table named students with columns student\_id, first\_name, last\_name, and grade. Insert some sample data into the table. Write an ALTER TABLE statement to add a new column gpa for storing the grade point average.**

-- Create students table

CREATE TABLE students (

student\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

grade CHAR(1)

);

-- Insert sample data into students table

INSERT INTO students (first\_name, last\_name, grade)

VALUES ('John', 'Doe', 'A'),

('Jane', 'Smith', 'B'),

('Michael', 'Johnson', 'A');

-- Add a new column gpa to the students table

ALTER TABLE students

ADD COLUMN gpa DECIMAL(3, 2);

**Exercise 12 - UPDATE with Subquery:**

**Scenario:**

**Create two tables named orders and order\_details with columns order\_id, customer\_id, and order\_date in the orders table, and order\_detail\_id, order\_id, product\_id, and quantity in the order\_details table. Insert data into both tables and write a query to update the quantity of a product in the order\_details table based on the order\_id.**

-- Create orders table

CREATE TABLE orders (

order\_id INT AUTO\_INCREMENT PRIMARY KEY,

customer\_id INT,

order\_date DATE

);

-- Create order\_details table

CREATE TABLE order\_details (

order\_detail\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT,

product\_id INT,

quantity INT

);

-- Insert data into orders table

INSERT INTO orders (customer\_id, order\_date)

VALUES

(1, '2023-08-01'),

(2, '2023-08-02');

-- Insert data into order\_details table

INSERT INTO order\_details (order\_id, product\_id, quantity)

VALUES

(1, 101, 5),

(1, 102, 3),

(2, 101, 2),

(2, 103, 4);

-- Update quantity of a product in order\_details table based on order\_id

UPDATE order\_details

SET quantity = quantity + 1

WHERE order\_id = 1;

**Exercise 13 - DELETE with Subquery:**

**Scenario:**

**Create two tables named authors and books with columns author\_id, author\_name in the authors table, and book\_id, book\_name, and author\_id in the books table. Insert data into both tables and write a query to delete all books written by a specific author.**

-- Create authors table

CREATE TABLE authors (

author\_id INT AUTO\_INCREMENT PRIMARY KEY,

author\_name VARCHAR(100)

);

-- Create books table

CREATE TABLE books (

book\_id INT AUTO\_INCREMENT PRIMARY KEY,

book\_name VARCHAR(100),

author\_id INT

);

-- Insert data into authors table

INSERT INTO authors (author\_name)

VALUES

('John Doe'),

('Jane Smith'),

('Michael Johnson');

-- Insert data into books table

INSERT INTO books (book\_name, author\_id)

VALUES

('Book 1', 1),

('Book 2', 2),

('Book 3', 3),

('Book 4', 1);

-- Delete all books written by author with author\_id 1

DELETE FROM books

WHERE author\_id = 1;

**Exercise 14 - SELECT with Aggregate Function:**

**Scenario:**

**Create a table named sales with columns sale\_id, product\_id, sale\_date, and amount. Insert data into the table and write a query to calculate the total sales amount for each product.**

-- Create sales table

CREATE TABLE sales (

sale\_id INT AUTO\_INCREMENT PRIMARY KEY,

product\_id INT,

sale\_date DATE,

amount DECIMAL(10, 2)

);

-- Insert data into sales table

INSERT INTO sales (product\_id, sale\_date, amount)

VALUES

(101, '2023-08-01', 50.00),

(102, '2023-08-02', 75.00),

(101, '2023-08-03', 30.00),

(103, '2023-08-03', 100.00),

(102, '2023-08-04', 60.00);

-- Calculate total sales amount for each product

SELECT product\_id, SUM(amount) AS total\_sales

FROM sales

GROUP BY product\_id;

**Exercise 15 - INSERT...SELECT with JOIN:**

**Scenario:**

**Create two tables named employees and departments with columns employee\_id, first\_name, last\_name in the employees table, and department\_id, department\_name in the departments table. Insert data into both tables and write an INSERT...SELECT statement to insert data from employees into a new table employee\_summary along with department names.**

-- Create departments table

CREATE TABLE departments (

department\_id INT AUTO\_INCREMENT PRIMARY KEY,

department\_name VARCHAR(100)

);

-- Create employees table

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

department\_id INT

);

-- Insert data into departments table

INSERT INTO departments (department\_name)

VALUES

('HR'),

('IT'),

('Finance');

-- Insert data into employees table

INSERT INTO employees (first\_name, last\_name, department\_id)

VALUES

('John', 'Doe', 1),

('Jane', 'Smith', 2),

('Michael', 'Johnson', 1),

('Susan', 'Williams', 3),

('Eva', 'Garcia', 2);

-- Insert data from employees into employee\_summary along with department names

INSERT INTO employee\_summary (employee\_id, first\_name, last\_name, department\_name)

SELECT e.employee\_id, e.first\_name, e.last\_name, d.department\_name

FROM employees e

JOIN departments d ON e.department\_id = d.department\_id;

**Exercise 16 - UPDATE with CASE Statement:**

**Scenario:**

**Create a table named orders with columns order\_id, order\_date, and order\_status. Insert data into the table and write a query to update the order\_status to "Shipped" for orders placed before a specific date.**

-- Create orders table

CREATE TABLE orders (

order\_id INT AUTO\_INCREMENT PRIMARY KEY,

order\_date DATE,

order\_status VARCHAR(50)

);

-- Insert data into orders table

INSERT INTO orders (order\_date, order\_status)

VALUES

('2023-08-01', 'Pending'),

('2023-08-02', 'Shipped'),

('2023-08-03', 'Pending'),

('2023-08-04', 'Processing'),

('2023-08-05', 'Shipped');

-- Update order\_status to "Shipped" for orders placed before '2023-08-03'

UPDATE orders

SET order\_status = CASE

WHEN order\_date < '2023-08-03' THEN 'Shipped'

ELSE order\_status

END;

**Exercise 17 - DELETE with LIMIT:**

**Scenario:**

**Create a table named logs with columns log\_id, log\_date, and description. Insert data into the table and write a query to delete the oldest 5 logs.**

-- Create logs table

CREATE TABLE logs (

log\_id INT AUTO\_INCREMENT PRIMARY KEY,

log\_date DATE,

description VARCHAR(200)

);

-- Insert data into logs table

INSERT INTO logs (log\_date, description)

VALUES

('2023-08-01', 'Log entry 1'),

('2023-08-02', 'Log entry 2'),

('2023-08-03', 'Log entry 3'),

('2023-08-04', 'Log entry 4'),

('2023-08-05', 'Log entry 5'),

('2023-08-06', 'Log entry 6'),

('2023-08-07', 'Log entry 7'),

('2023-08-08', 'Log entry 8'),

('2023-08-09', 'Log entry 9'),

('2023-08-10', 'Log entry 10');

-- Delete the oldest 5 logs

DELETE FROM logs

ORDER BY log\_date

LIMIT 5;

**Exercise 18 - UPDATE with Subquery and JOIN:**

**Scenario:**

**Create two tables named employees and salaries with columns employee\_id, first\_name, last\_name in the employees table, and employee\_id and salary in the salaries table. Insert data into both tables and write a query to update the salary of an employee in the salaries table based on the employee\_id.**

-- Create employees table

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50)

);

-- Create salaries table

CREATE TABLE salaries (

employee\_id INT PRIMARY KEY,

salary DECIMAL(10, 2)

);

-- Insert data into employees table

INSERT INTO employees (first\_name, last\_name)

VALUES

('John', 'Doe'),

('Jane', 'Smith'),

('Michael', 'Johnson'),

('Susan', 'Williams'),

('Eva', 'Garcia');

-- Insert data into salaries table

INSERT INTO salaries (employee\_id, salary)

VALUES

(1, 50000.00),

(2, 55000.00),

(3, 60000.00);

-- Update salary of an employee in salaries table based on employee\_id

UPDATE salaries s

JOIN employees e ON s.employee\_id = e.employee\_id

SET s.salary = s.salary \* 1.10

WHERE e.last\_name = 'Doe';

**Exercise 19 - DELETE with Self-Join:**

**Scenario:**

**Create a table named employees with columns employee\_id, first\_name, last\_name, and manager\_id. Insert data into the table and write a query to delete all employees who do not have a manager.**

-- Create employees table

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

manager\_id INT

);

-- Insert data into employees table

INSERT INTO employees (first\_name, last\_name, manager\_id)

VALUES

('John', 'Doe', NULL),

('Jane', 'Smith', 1),

('Michael', 'Johnson', NULL),

('Susan', 'Williams', 2),

('Eva', 'Garcia', 1);

-- Delete employees who do not have a manager

DELETE e

FROM employees e

LEFT JOIN employees m ON e.manager\_id = m.employee\_id

WHERE m.employee\_id IS NULL;

**Exercise 20 - SELECT...FOR UPDATE Statement:**

**Scenario:**

**Create a table named inventory with columns product\_id, product\_name, and stock\_quantity. Insert data into the table and write a query to select and lock rows for a specific product for updating.**

-- Create inventory table

CREATE TABLE inventory (

product\_id INT AUTO\_INCREMENT PRIMARY KEY,

product\_name VARCHAR(100),

stock\_quantity INT

);

-- Insert data into inventory table

INSERT INTO inventory (product\_name, stock\_quantity)

VALUES

('Widget', 100),

('Gadget', 50),

('Doodad', 75),

('Thingamajig', 120),

('Whatchamacallit', 90);

-- Select and lock rows for a specific product for updating

SELECT \*

FROM inventory

WHERE product\_id = 3

FOR UPDATE;