**Exercise 1: One-to-One Relationship**

Consider two tables, **employees** and **employee\_details**, where each employee has exactly one corresponding detail record

**Table: employees**

employee\_id

employee\_name

employee\_salary

CREATE TABLE employees (

employee\_id INT PRIMARY KEY,

employee\_name VARCHAR(50),

employee\_salary DECIMAL(10, 2)

);

**Table: employee\_details**

employee\_detail\_id

employee\_id

hire\_date

CREATE TABLE employee\_details (

detail\_id INT PRIMARY KEY,

employee\_id INT UNIQUE,

hire\_date DATE,

FOREIGN KEY (employee\_id) REFERENCES employees(employee\_id)

);

Insert data into the tables to represent a one-to-one relationship between employees and their details.

(1, 'John Doe', 50000.00)

(2, 'Jane Smith', 60000.00)

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

(2, 2, '2023-02-01')

**Answer:**

-- Insert data into employees

INSERT INTO employees (employee\_id, employee\_name, employee\_salary)

VALUES

(1, 'John Doe', 50000.00),

(2, 'Jane Smith', 60000.00);

-- Insert data into employee\_details

INSERT INTO employee\_details (detail\_id, employee\_id, hire\_date)

VALUES

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

(2, 2, '2023-02-01');

**Exercise 2: One-to-Many Relationship**

Consider two tables, **departments** and **employees**, where each department can have multiple employees.

**Table: departments**

department\_id

department\_name

**Table: employees**

employee\_id

employee\_name

department\_id

CREATE TABLE departments (

department\_id INT PRIMARY KEY,

department\_name VARCHAR(50)

);

CREATE TABLE employees (

employee\_id INT PRIMARY KEY,

employee\_name VARCHAR(50),

department\_id INT,

FOREIGN KEY (department\_id) REFERENCES departments(department\_id)

);

Insert data into the tables to represent a one-to-many relationship between departments and employees.

(1, 'HR')

(2, 'IT')

(1, 'Alice Johnson', 1)

(2, 'Bob Smith', 1)

(3, 'Charlie Brown', 2)

**Answer:**

-- Insert data into departments

INSERT INTO departments (department\_id, department\_name)

VALUES

(1, 'HR'),

(2, 'IT');

-- Insert data into employees

INSERT INTO employees (employee\_id, employee\_name, department\_id)

VALUES

(1, 'Alice Johnson', 1),

(2, 'Bob Smith', 1),

(3, 'Charlie Brown', 2);

**Exercise 3: Many-to-Many Relationship**

Consider two tables, **students** and **courses**, where students can enroll in multiple courses, and each course can have multiple students.

**Table: students**

student\_id

student\_name

**Table: courses**

course\_id

course\_name

**Table: enrollments**

enrollment\_id

student\_id

course\_id

**Answers:**

CREATE TABLE students (

student\_id INT PRIMARY KEY,

student\_name VARCHAR(50)

);

CREATE TABLE courses (

course\_id INT PRIMARY KEY,

course\_name VARCHAR(50)

);

CREATE TABLE enrollments (

enrollment\_id INT PRIMARY KEY,

student\_id INT,

course\_id INT,

FOREIGN KEY (student\_id) REFERENCES students(student\_id),

FOREIGN KEY (course\_id) REFERENCES courses(course\_id)

);

Insert data into the tables to represent a many-to-many relationship between students and courses.

**Answer:**

-- Insert data into students

INSERT INTO students (student\_id, student\_name)

VALUES

(1, 'Tom Johnson'),

(2, 'Emily Davis');

-- Insert data into courses

INSERT INTO courses (course\_id, course\_name)

VALUES

(1, 'Mathematics'),

(2, 'History');

-- Insert data into enrollments

INSERT INTO enrollments (enrollment\_id, student\_id, course\_id)

VALUES

(1, 1, 1),

(2, 1, 2),

(3, 2, 1);

**Exercise 4: One-to-Many Relationship with Orders and Order Items**

Consider two tables, **orders** and **order\_items**, where each order can have multiple items.

**Table: orders**

order\_id

customer\_name

order\_date

**Table: order\_items**

item\_id

order\_id

product\_name

quantity

price

Insert data into the tables to represent a one-to-many relationship between orders and order items.

**Answers:**

CREATE TABLE orders (

order\_id INT PRIMARY KEY,

customer\_name VARCHAR(50),

order\_date DATE

);

CREATE TABLE order\_items (

item\_id INT PRIMARY KEY,

order\_id INT,

product\_name VARCHAR(50),

quantity INT,

price DECIMAL(10, 2),

FOREIGN KEY (order\_id) REFERENCES orders(order\_id)

);

-- Insert data into orders

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

VALUES

(1, 'John Doe', '2023-01-15'),

(2, 'Jane Smith', '2023-02-01');

-- Insert data into order\_items

INSERT INTO order\_items (item\_id, order\_id, product\_name, quantity, price)

VALUES

(1, 1, 'Laptop', 2, 1200.00),

(2, 1, 'Printer', 1, 200.00),

(3, 2, 'Tablet', 3, 500.00);

In this example, each order in the **orders** table can have multiple items associated with it in the **order\_items** table. This scenario is common in e-commerce or sales database setups. The **order\_id** column in the **order\_items** table establishes the one-to-many relationship with the **orders** table.