MySQL Assignment

Topic: DDL, DML, and JOIN Queries for Categories, Suppliers, and Products

Part A: DDL & DML

Question:

1) DDL: Create the three tables: categories, suppliers, and products.

Answer (SQL):

```
CREATE TABLE categories (
category_id INT AUTO_INCREMENT PRIMARY KEY,
category_name VARCHAR(50) NOT NULL UNIQUE,
description TEXT
);

CREATE TABLE suppliers (
supplier_id INT AUTO_INCREMENT PRIMARY KEY,
supplier_name VARCHAR(100) NOT NULL,
contact_email VARCHAR(100),
phone VARCHAR(20)
);

CREATE TABLE products (
product_id INT AUTO_INCREMENT PRIMARY KEY,
product_name VARCHAR(100) NOT NULL,
category_id INT,
supplier_id INT,
price DECIMAL(10,2) CHECK (price >= 0),
stock_quantity INT CHECK (stock_quantity >= 0),
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
CONSTRAINT fk_products_category FOREIGN KEY (category_id) REFERENCES
categories(category_id),
CONSTRAINT fk_products_supplier FOREIGN KEY (supplier_id) REFERENCES
suppliers(supplier_id)
);
```

Question:

2) DML: Insert sample data (3 categories, 2 suppliers, 5 products).

Answer (SQL):

```
-- Insert 3 categories
INSERT INTO categories (category_name, description) VALUES
('Electronics', 'Devices and gadgets'),
('Books', 'Printed and digital books'),
('Clothing', 'Apparel and accessories');

-- Insert 2 suppliers
INSERT INTO suppliers (supplier_name, contact_email, phone) VALUES
('Best Supplier Inc.', 'contact@bestsupplier.com', '123-456-7890'),
('Global Goods', 'sales@globalgoods.com', '987-654-3210');

-- Insert 5 products
INSERT INTO products (product_name, category_id, supplier_id, price, stock_quantity)
VALUES
('Smartphone Model X', 1, 1, 699.99, 50),
('Wireless Headphones', 1, 2, 199.99, 30),
('Mystery Novel', 2, 1, 14.99, 100),
```

```
('T-shirt Classic', 3, 2, 9.99, 200),
('E-reader', 1, 1, 129.99, 10);
```

Part B: Queries (DDL + DML)

Question:

(a) Retrieve a list of all products with their category name and supplier name.

Answer (SQL):

```
SELECT
p.product_id,
p.product_name,
c.category_name,
s.supplier_name,
p.price,
p.stock_quantity
FROM products p
JOIN categories c ON p.category_id = c.category_id
JOIN suppliers s ON p.supplier_id = s.supplier_id;
```

Question:

(b) Find all products where stock quantity is below 5.

Answer (SQL):

```
SELECT *
FROM products
WHERE stock_quantity < 5;
```

Question:

(c) Add a new column discount_percent to the products table with a default value of 0.

Answer (SQL):

```
ALTER TABLE products
ADD COLUMN discount_percent DECIMAL(5,2) NOT NULL DEFAULT 0;
```

Question:

(d) Reduce the price of all products in the "Electronics" category by 15%.

Answer (SQL):

```
UPDATE products p
JOIN categories c ON p.category_id = c.category_id
SET p.price = p.price * 0.85
WHERE c.category_name = 'Electronics';
```

Part C: JOIN Queries

Question:

(a) List all products with their corresponding category name using an INNER JOIN.

Answer (SQL):

```
SELECT
p.product_id,
p.product_name,
c.category_name
FROM products p
INNER JOIN categories c
ON p.category_id = c.category_id;
```

Question:

(b) List all products with their category name, including products that do not belong to any category (LEFT JOIN).

Answer (SQL):

```
SELECT
p.product_id,
p.product_name,
c.category_name
FROM products p
LEFT JOIN categories c
ON p.category_id = c.category_id;
```

Question:

(d) List all products along with their supplier names, including products that have no supplier assigned (LEFT JOIN).

Answer (SQL):

```
SELECT
p.product_id,
p.product_name,
s.supplier_name
FROM products p
LEFT JOIN suppliers s
ON p.supplier_id = s.supplier_id;
```

Question:

(e) List all suppliers and the products they supply, including suppliers who supply no products (RIGHT JOIN).

Answer (SQL):

```
SELECT
s.supplier_id,
s.supplier_name,
p.product_name
FROM products p
RIGHT JOIN suppliers s
ON p.supplier_id = s.supplier_id;
```

Question:

(f) Find all products that do not have a supplier assigned.

Answer (SQL):

```
SELECT *
FROM products
WHERE supplier_id IS NULL;
```

Question:

(g) Get all products with their category name and supplier name using multiple JOINs.

Answer (SQL):

```
SELECT
p.product_id,
p.product_name,
c.category_name,
s.supplier_name
FROM products p
LEFT JOIN categories c ON p.category_id = c.category_id
LEFT JOIN suppliers s ON p.supplier_id = s.supplier_id;
```

Question:

(h) Get a list of all suppliers and categories, even if there are no products linking them (FULL OUTER JOIN between suppliers and categories).

Answer (SQL):

```
-- MySQL does NOT support FULL OUTER JOIN directly.
-- Use UNION of LEFT JOIN and RIGHT JOIN.

SELECT s.supplier_name, c.category_name
FROM suppliers s
LEFT JOIN products p ON s.supplier_id = p.supplier_id
LEFT JOIN categories c ON p.category_id = c.category_id

UNION

SELECT s.supplier_name, c.category_name
FROM categories c
LEFT JOIN products p ON c.category_id = p.category_id
LEFT JOIN suppliers s ON p.supplier_id = s.supplier_id;
```

Question:

(i) Find products where the supplier's contact email is not NULL using a join.

Answer (SQL):

```
SELECT
p.product_name,
s.supplier_name,
s.contact_email
FROM products p
JOIN suppliers s
ON p.supplier_id = s.supplier_id
WHERE s.contact_email IS NOT NULL;
```

Question:

(j) Find categories that have products supplied by supplier named 'Global Goods'.

Answer (SQL):

```
SELECT DISTINCT
c.category_name
FROM categories c
JOIN products p
ON c.category_id = p.category_id
JOIN suppliers s
ON p.supplier_id = s.supplier_id
WHERE s.supplier_name = 'Global Goods';
```