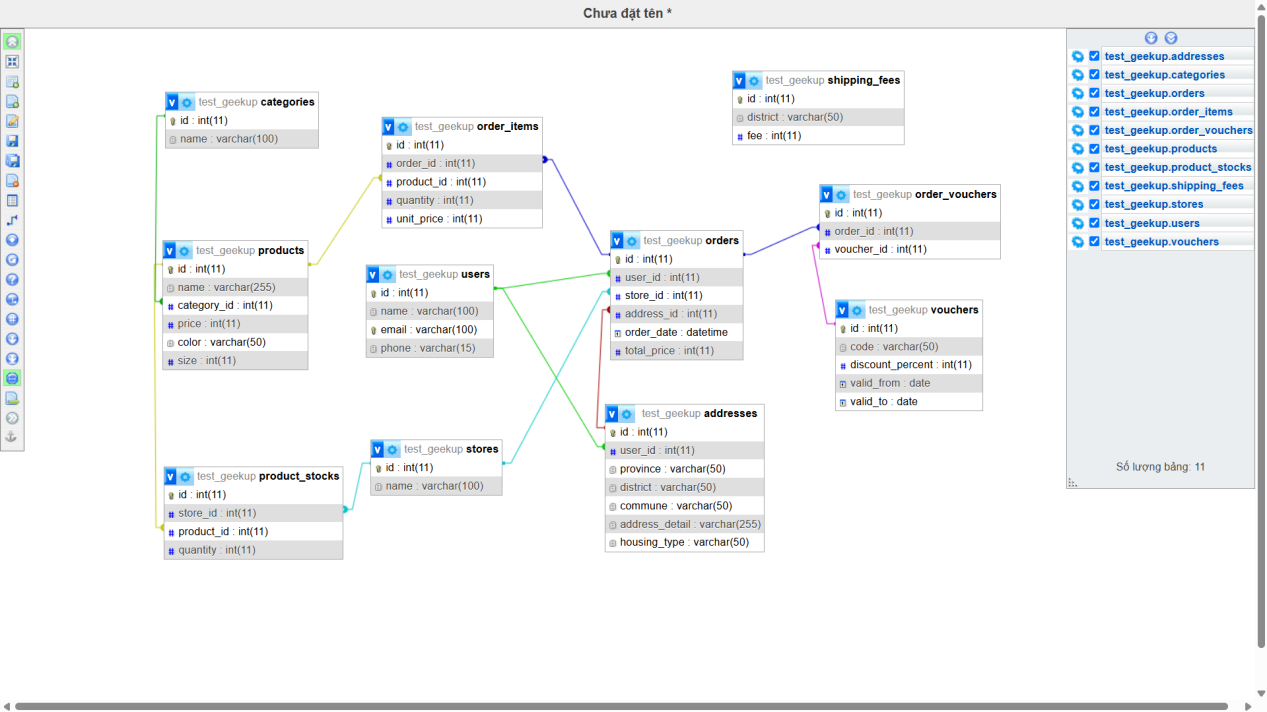
**Câu 1:**

Mục tiêu: Tạo các bảng ( create table )



**1. Người dùng**

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(15)

);

**2. Địa chỉ người dùng**

CREATE TABLE addresses (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT NOT NULL,

province VARCHAR(50),

district VARCHAR(50),

commune VARCHAR(50),

address\_detail VARCHAR(255),

housing\_type VARCHAR(50),

FOREIGN KEY (user\_id) REFERENCES users(id)

);

**3. Danh mục sản phẩm**

CREATE TABLE categories (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL

);

**4. Cửa hàng**

CREATE TABLE stores (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL

);

**5. Sản phẩm**

CREATE TABLE products (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

category\_id INT,

price INT,

color VARCHAR(50),

size INT,

FOREIGN KEY (category\_id) REFERENCES categories(id)

);

**6. Tồn kho theo cửa hàng**

CREATE TABLE product\_stocks (

id INT AUTO\_INCREMENT PRIMARY KEY,

store\_id INT NOT NULL,

product\_id INT NOT NULL,

quantity INT,

FOREIGN KEY (store\_id) REFERENCES stores(id),

FOREIGN KEY (product\_id) REFERENCES products(id)

);

**7. Đơn hàng**

CREATE TABLE orders (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT NOT NULL,

store\_id INT,

address\_id INT,

order\_date DATETIME DEFAULT CURRENT\_TIMESTAMP,

total\_price INT,

FOREIGN KEY (user\_id) REFERENCES users(id),

FOREIGN KEY (store\_id) REFERENCES stores(id),

FOREIGN KEY (address\_id) REFERENCES addresses(id)

);

**8. Chi tiết đơn hàng (nhiều sản phẩm)**

CREATE TABLE order\_items (

id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT NOT NULL,

product\_id INT NOT NULL,

quantity INT,

unit\_price INT,

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

FOREIGN KEY (product\_id) REFERENCES products(id)

);

**9. Voucher**

CREATE TABLE vouchers (

id INT AUTO\_INCREMENT PRIMARY KEY,

code VARCHAR(50) NOT NULL,

discount\_percent INT,

valid\_from DATE,

valid\_to DATE

);

**10. Gán nhiều voucher vào một đơn hàng**

CREATE TABLE order\_vouchers (

id INT AUTO\_INCREMENT PRIMARY KEY,

order\_id INT NOT NULL,

voucher\_id INT NOT NULL,

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

FOREIGN KEY (voucher\_id) REFERENCES vouchers(id)

);

**11. Phí vận chuyển theo quận/huyện**

CREATE TABLE shipping\_fees (

id INT AUTO\_INCREMENT PRIMARY KEY,

district VARCHAR(50),

fee INT

);

**Câu 2:**

Đề có thể viết câu truy vấn như trong cầu hỏi thì phải insert dữ liệu vào trong các bảng:

INSERT INTO users (name, email, phone)

VALUES ('assessment', 'gu@gmail.com', '3283555333');

INSERT INTO addresses (user\_id, province, district, commune, address\_detail, housing\_type)

VALUES (1, 'Bắc Kạn', 'Ba Bể', 'Phúc Lộc', '73 tân hòa 2', 'nhà riêng');

INSERT INTO categories (name)

VALUES ('Shoes');

INSERT INTO stores (name)

VALUES ('Main Store');

INSERT INTO products (name, category\_id, price, size, color)

VALUES ('KAPPA Women''s Sneakers', 1, 980000, 36, 'yellow');

INSERT INTO product\_stocks (store\_id, product\_id, quantity)

VALUES (1, 1, 5);

INSERT INTO orders (user\_id, store\_id, address\_id, total\_price)

VALUES (1, 1, 1, 980000);

INSERT INTO order\_items (order\_id, product\_id, quantity, unit\_price)

VALUES (1, 1, 1, 980000);

**Câu 3:**

**Câu truy vấn:**

SELECT

MONTH(order\_date) AS month,

AVG(total\_price) AS average\_order\_value

FROM

orders

WHERE

YEAR(order\_date) = YEAR(CURDATE()) -- chỉ lấy năm hiện tại

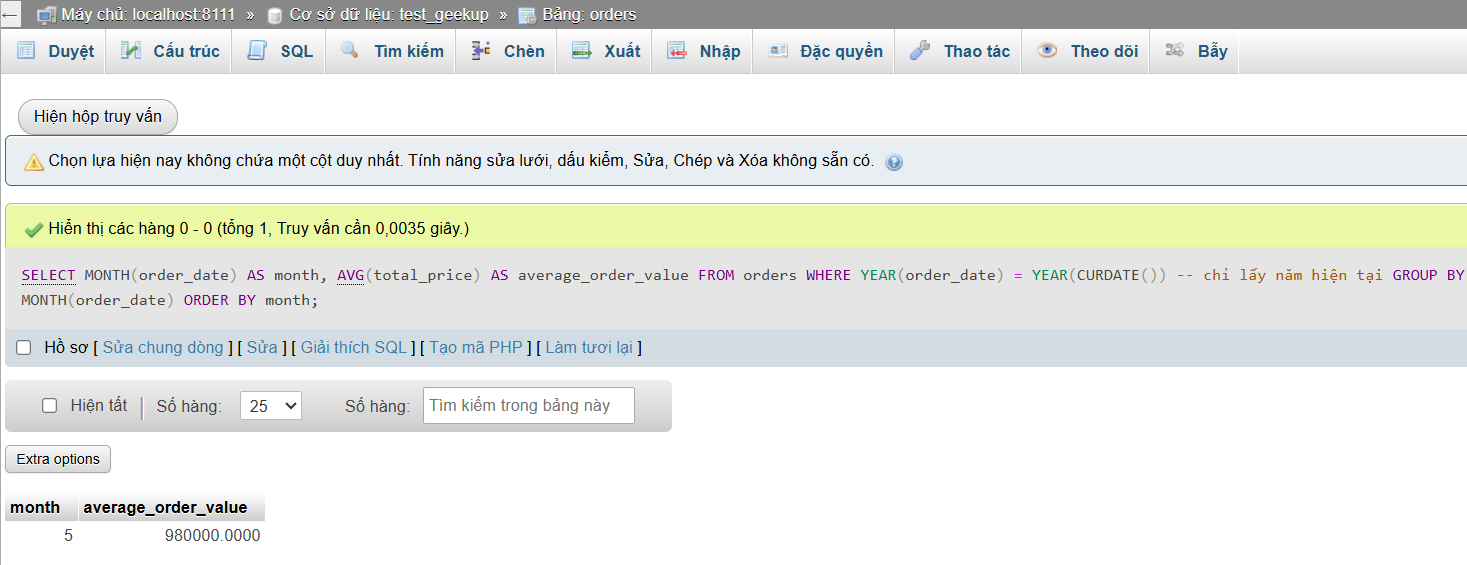
GROUP BY

MONTH(order\_date)

ORDER BY

month;

**Kết quả:**



**Câu 4:**

1. **Danh sách người dùng đã mua trong khoảng 6–12 tháng trước**

WITH past\_users AS (

SELECT DISTINCT user\_id

FROM orders

WHERE order\_date BETWEEN DATE\_SUB(CURDATE(), INTERVAL 12 MONTH)

AND DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

),

1. **Danh sách người dùng đã mua trong 6 tháng gần nhất**

recent\_users AS (

SELECT DISTINCT user\_id

FROM orders

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

)

1. **Tính churn rate**

SELECT

(SELECT COUNT(\*) FROM past\_users

WHERE user\_id NOT IN (SELECT user\_id FROM recent\_users)

) AS churned\_users,

(SELECT COUNT(\*) FROM past\_users) AS total\_past\_users,

ROUND(

(SELECT COUNT(\*) FROM past\_users

WHERE user\_id NOT IN (SELECT user\_id FROM recent\_users)

) / (SELECT COUNT(\*) FROM past\_users) \* 100,

2) AS churn\_rate\_percent;