Sinh viên thực hiện: 18126008 – Lê Bá Quốc

GV phụ trách: CSDL Nâng cao

Đồ án/bài tập môn học  - CƠ SỞ DỮ LIỆU NÂNG CAO

HỌC KỲ I – NĂM HỌC 2019-2020

**môn cơ sở dữ liệu nâng cao**

**BỘ MÔN HỆ THỐNG THÔNG TIN – KHOA CÔNG NGHỆ THÔNG TIN**

**ĐẠI HỌC KHOA HỌC TỰ NHIÊN THÀNH PHỐ HỒ CHÍ MINH, ĐẠI HỌC QUỐC GIA TP HCM**

**BẢNG THÔNG TIN CHI TIẾT NHÓM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mã nhóm:** | 03 | | | |
| **Tên nhóm:** | A Little Hope | | | |
| **Số lượng:** | **4** | | | |
| **MSSV** | **Họ tên** | **Email** | **Điện thoại** | **Hình ảnh** |
| 18126008 | Lê Bá Quốc | 18126008@student.hcmus.edu.vn | 0398661333 | A picture containing clipart  Description automatically generated |
| 18126032 | Đỗ Nguyễn Minh Thành | 18126032@student.hcmus.edu.vn |  |  |
| 18126017 | Trần Thị Ngọc Huyền | 18126017@student.hcmus.edu.vn |  |  |
| 18126006 | Thái Hoàng Nhân | 18126006@student.hcmus.edu.vn |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Bảng phân công & đánh giá hoàn thành công việc** | | | |
| **Công việc thực hiện** | **Người thực hiện** | **Mức độ hoàn thành** | **Đánh giá của nhóm** |
| Đặc tả đồ án | 18126008 – Lê Bá Quốc  18126017 – Trần Thị Ngọc Huyền  18126032 – Đỗ Nguyễn Minh Thành  18126006 – Thái Hoàng Nhân | 100% | 9/10 |
| Quy trình nghiệp vụ | 18126008 – Lê Bá Quốc | 100% | 10/10 |
| Toàn vẹn dữ liệu | 18126017 – Trần Thị Ngọc Huyền  18126008 – Lê Bá Quốc | 100% | 10/10 |
| Yêu cầu chức năng | 18126032 – Đỗ Nguyễn Minh Thành  18126008 – Lê Bá Quốc | 100% | 10/10 |
| Xác định yêu cầu dữ liệu | 18126017 – Trần Thị Ngọc Huyền | 100% | 10/10 |
| Chuẩn hóa dữ liệu | 18126008 – Lê Bá Quốc  18126017 – Trần Thị Ngọc Huyền  18126032 – Đỗ Nguyễn Minh Thành  18126006 – Thái Hoàng Nhân |  |  |
| Layout | 18126006 – Thái Hoàng Nhân  18126017 – Trần Thị Ngọc Huyền | 100% | 9/10 |
| Xác định chỉ mục | 18126008 – Lê Bá Quốc  18126017 – Trần Thị Ngọc Huyền | 100% | 10/10 |
| Cài đặt chỉ mục | 18126006 – Thái Hoàng Nhân | 100% | 10/10 |
| Phân tích | 18126008 – Lê Bá Quốc  18126017 – Trần Thị Ngọc Huyền | 100% | 10/10 |
| Kiểm chứng hiệu quả của chỉ mục | 18126008 – Lê Bá Quốc  18126006 – Thái Hoàng Nhân | 100% | 10/10 |
| Partition | 18126008 – Lê Bá Quốc  18126017 – Trần Thị Ngọc Huyền | 100% | 10/10 |
| Chức năng | | | |
| * Trang admin * Xem doanh thu * Xem danh sách user trong hệ thống, tìm kiếm theo tên * Xem các nhà cung cấp, xem danh sách đối tác giao dịch trong hệ thống, đơn vị vận chuyển | 18126032 – Đỗ Nguyễn Minh Thành | 100% | 10/10 |
| Chức năng của nhà cung cấp:   * Xem thông tin của nhà cung cấp và chỉnh sửa * Thêm sản phẩm vào hệ thống * Cập nhật giá | 18126017 – Trần Thị Ngọc Huyền | 100% | 10/10 |
| Chức năng của nhà cung cấp:   * Xem danh sách đơn hàng của nhà cung cấp và tùy chỉnh * Xem doanh thu ngày/tháng/năm, từ ngày này sang ngày khác | 18126006 – Thái Hoàng Nhân | 100% | 10/10 |
| Giỏ Hàng:   * Ước tính giá ship, thay đổi số lượng hoặc xóa * Chức năng thanh toán đơn hàng * Chọn đơn vị vận chuyển, thời gian hình thức thanh toán, tiến hành thanh toán đơn hàng | 18126008 – Lê Bá Quốc | 100% | 10/10 |
| Trang chủ:   * Xem danh sách sản phẩm trên trang chủ * Hiển thị theo danh sách , tìm kiếm theo tên sản phẩm * Đăng nhấp , đăng ký | 18126032 – Đỗ Nguyễn Minh Thành  18126008 – Lê Bá Quốc | 100% | 10/10 |
| User:   * Xem thông tin các nhân, địa chỉ giao hàng, đơn vừa đặt, thay đổi thông tin cá nhân * Thay đổi địa chỉ giao hàng | 18126017 – Trần Thị Ngọc Huyền | 100% | 10/10 |
| User :   * Kiểm tra order theo mã order và email * Xem danh sách tất cả order và đang đặt | 18126006 – Thái Hoàng Nhân | 100% | 10/10 |
|  |  |  |  |
|  |  |  |  |

**YÊU CẦU ĐỒ ÁN- BÀI TẬP**

|  |  |
| --- | --- |
| **Loại bài tập** | **🗹 Lý thuyết Thực hành Đồ án 🗹 Bài tập** |
| **Ngày bắt đầu** | **24/06/2021** |
| **Ngày kết thúc** | **13/08/2021** |

Contents

[**Yêu cầu của Đồ án/Bài tập** 4](#_Toc80858561)

[**1.** **Mô Tả:** 4](#_Toc80858562)

[a. Quy trình nghiệp vụ 4](#_Toc80858563)

[*1)* *Quy trình cho người mua:* 4](#_Toc80858564)

[2) *Quy trình cho người bán:* 4](#_Toc80858565)

[*3)* *Quy trình giao nhận hàng vận chuyển* 4](#_Toc80858566)

[*4)* *Quy trình thanh toán:* 4](#_Toc80858567)

[**Kết quả** 4](#_Toc80858568)

[**1.** **Thiết kế dữ liệu quan niệm** 4](#_Toc80858569)

[a. Đặc tả hệ thống 4](#_Toc80858570)

[b. Ràng buộc dữ liệu 6](#_Toc80858571)

[2. Thiết kế dữ liệu logic 11](#_Toc80858572)

[a. Mô hình ER 11](#_Toc80858573)

[b. Mô hình quan hệ 12](#_Toc80858574)

[c. Chuẩn hóa dữ liệu: 13](#_Toc80858575)

[**3.** **Thiết kế vật lý (thông tin tần suất, cài chỉ mục, phân tích, kiểm chứng hiệu quả chỉ mục…)** 16](#_Toc80858576)

[a. Thông tin tần suất 16](#_Toc80858577)

[b. Chỉ mục 16](#_Toc80858578)

[c. Phân tích 18](#_Toc80858579)

[d. Kiểm chứng hiệu quả chỉ mục 19](#_Toc80858580)

[e. Trigger và Procedure 36](#_Toc80858581)

[4. Xác định yêu cầu chức năng: 42](#_Toc80858582)

[**5.** **Layout của giao diện:** 45](#_Toc80858583)

[**6.** **Partiton** 56](#_Toc80858584)

[**I.** **Partition:** 56](#_Toc80858585)

[**II.** **Lợi ích của việc sử dụng Partition:** 57](#_Toc80858586)

[**III.** **Các kiểu phân chia dữ liệu:** 57](#_Toc80858587)

[**IV.** **Phân chia dữ liệu bảng Order (Hoá đơn) theo chiều ngang** 57](#_Toc80858588)

# **Yêu cầu của Đồ án/Bài tập**

1. **Mô Tả:**
2. Quy trình nghiệp vụ
3. *Quy trình cho người mua:*

* Đăng nhập tài khoản Shopee;
* Tìm kiếm, tham khảo thông tin sản phẩm, dịch vụ, khuyến mại và các Người Bán mà Người Mua đang quan tâm.
* Tham khảo thông tin giá và chính sách hỗ trợ của bên bán sản phẩm, dịch vụ mà Người Mua đang có nhu cầu mua.
* Người Mua đưa ra quyết định đặt hàng trực tuyến bằng cách click vào “Đặt hàng”.
* Người Mua lựa chọn cách thức giao dịch tuy theo chính sách của từng gian hàng đặt ra.
* Đơn hàng của Người Mua sẽ được chuyển thông tin đến Người Bán.
* Người Mua nhận sản phẩm, dịch vụ;

2) *Quy trình cho người bán:*

* Đăng ký tài khoản Shopee;
* Shopee xác nhận và kích hoạt tài khoản; ( có hoặc không )
* Sau khi đăng nhập, Người Bán tiến hành đăng tải thông tin bán hàng:
* Chuẩn bị tin bài bằng chữ và hình ảnh;
* Các tin bài cần đăng phải được chia thành 02 phần thông tin sản phẩm và hình ảnh sản phẩm.
* Các nội dung bằng chữ bị giới hạn 3.000 ký tự cho phần mô tả sản phẩm
* Các nội dung bằng hình ảnh được định dạng theo dạng ảnh jpg, bmp, hoặc png. Số lượng ảnh tối đa cho 01 lần đăng tin là 09 ảnh.
* Đưa nội dung lên Sàn giao dịch TMĐT Shopee;
* Shopee kiểm duyệt thông tin sản phẩm, dịch vụ của Người Bán khi đưa lên Sàn giao dịch TMĐT Shopee.
  1. *Quy trình giao nhận hàng vận chuyển*
* Sau khi nhận được đơn hàng từ Người Mua,Shopee sẽ chuyển thông tin đơn hàng đó đến Người Bán và xác nhận hình thức giao dịch.
* Shopee sẽ cử đơn vị vận chuyển tới chỗ Người Bán và vận chuyển sản phẩm đó tới địa chỉ đăng ký của Người Mua.
* Thời gian giao hàng được bắt đầu tính từ lúc đơn hàng được Người Bán giao cho đơn vị vận chuyển thành công tới khi đơn vị vận chuyển liên hệ lần đầu tiên với Người Mua để giao hàng.
* ***Thời gian giao hàng:***
* Trong phạm vi Nội thành Hà Nội/TP Hồ Chí Minh/Đà Nẵng: 01 - 03 ngày
* Ngoài khu vực này thì 4 – 7 ngày
  1. *Quy trình thanh toán:*
* Cách 1: Thanh toán bằng thẻ : thì tiền lập tức được thêm vào lịch sử giao dịch của cửa hàng và đơn vị vận chuyển và cả shoppe.
* Cách 2:  Sau khi khách đã nhận hàng, tiền hàng sẽ được cập nhật vào lịch sử giao dịch của cửa hàng và đơn vị vận chuyển + shoppe
* Sau khi đơn hàng được giao thành công, shoppePhake sẽ trích tiền thanh toán cho cửa hàng và nhà vận chuyển, sau đó thu hồi tiền ở những tài khoản nhận tiền điện tử, hoặc nếu là phương thức thanh toán tiền khi nhận hàng thì shoppePhake sẽ nhận lại tiền từ nhà vận chuyển( sau khi đã trừ phí vận chuyển ). Những lần giao dịch sẽ được ghi nhận lại.

1. Danh sách yêu cầu chức năng

* Đăng nhập/Đăng ký user
* Tìm lại password
* Hiển thị sản phẩm
* Thêm sản phẩm vào giỏ hàng
* Thêm/xóa/ cập nhật thông tin sản phẩm
* Order sản phẩm
* Chỉnh sửa thông tin user
* Sắp xếp hiển thị sản phẩm theo loại sản phẩm

1. Yêu cầu phi chức năng

* Toàn vẹn dữ liệu
* Khả năng tương tác
* Khả năng sử dụng
* Khả năng quản lí
* Khả năng bảo trì

# **Kết quả**

1. **Thiết kế dữ liệu quan niệm**
2. Đặc tả hệ thống

Công ty QTNN mở một trang thương mại điện tử (ShopePhaKe) chuyên bán đủ loại mặt hàng. Công ty cần xây dựng 1 trang web cho phép công ty và các cá nhân thực hiện việc kinh doanh qua mạng điện tử.

Sau đây là kết quả của việc phân tích yêu cầu ứng dụng:

* Trang ShopePhaKe bán đủ loại mặt hàng với nguồn hàng lấy từ các nhà cung cấp (**Mỗi shop trên trang ShopePhaKe được gọi là 1 nhà cung cấp**).
* Một **nhà cung cấp** có: *mã nhà cung cấp, tên, địa chỉ, và số điện thoại, email, tài khoản ngân hàng của nhà cung cấp*. Mỗi nhà cung cấp có thể cung ứng nhiều mặt hàng khác nhau và mỗi mặt hàng cũng có thể được cung cấp bởi nhiều nhà cung cấp khác nhau.
* Cần lưu trữ **thông tin khách hàng**, gồm : *mã khách hàng, tên khách hàng, số điện thoại, email, địa chỉ, ngày sinh, giới tính, tài khoản thẻ ( đối với sử dụng thanh toán online).* Mỗi khách hàng chỉ được sử dụng 1 email và 1 số điện thoại duy nhất để đăng ký.
* Cần lưu lại thông tin về tất cả các **sản phẩm** mà ShopePhaKe gồm có: *Mã sản phẩm, mã loại sản phẩm, tên sản phẩm, giá bán, số lượng tồn, mô tả sản phẩm,mã nhà cung cấp*. Mỗi sản phẩm thuộc 1 **loại sản phẩm** gồm : *mã loại sản phẩm, tên loại sản phẩm*. Mỗi sản phẩm chỉ có 1 mã sản phẩm duy nhất ứng với từng nhà cung cấp. Cần ghi nhận lại **sự thay đổi giá bán của sản phẩm theo thời gian** (nhằm mục đích phân tích), gồm các thông tin: *mã nhật ký thay đổi*,*mã sản phẩm, mã nhà cung cấp, giá cũ, giá mới, ngày cập nhật*.
* Khi khách hàng chọn sản phẩm, thì sản phẩm sẽ được đưa vào **giỏ hàng** của KHàng đó, có: *mã thứ tự giỏ hàng*, *mã sản phẩm*, *mã hóa đơn*,*đơn giá*, *số lượng*, *thành tiền*. Một Khách Hàng có thể mua được nhiều sản phẩm cùng một lúc. Khách hàng nhận được hóa **đơn đặt hàng** gồm : *mã đơn hàng*, *mã khách hàng*, *mã nhà cung cấp*, *ngày lập*,*trạng thái đơn hàng*, *phương thức giao dịch*, *địa chỉ nhận hàng*,*tổng tiền* – Ngay khi tiến hành đặt mua. Cần lưu lại **phí vận chuyển của đơn hàng** mà khách hàng vừa mua, gồm : *mã đơn hàng*, *mã nhà vận chuyển*, *phí vận chuyển*, *thời gian giao hàng*.
* ShoppePhake có nhiều **hình thức thanh toán** và được quản lí gồm những thông tin sau: *mã hình thức thanh toán, tên hình thức thanh toán*. Đối với mỗi hình thức thanh toán Shoppephake có nhiều **đối tác giao dịch** và các đối tác này được quản lí bởi : *mã đơn vị thanh toán, tên đơn vị thanh toán, mã hình thức thanh toán,ngày hợp tác, số tài khoản giao dịch*. Hình thức thanh toán và đối tác giao dịch phải được liên kết chặt chẽ, sau khi đơn hàng đã được người nhận thanh toán, thì ShoppePhake sẽ được nhận số tiền giao dịch trong tài khoản giao dịch (của shoppePhake).
* Sau khi đã đặt hàng thành công và được nhà cung cấp xác nhận. ShoppePhake sẽ chuyển thông tin đơn hàng đến **nhà vận chuyển**, gồm: *mã nhà vận chuyển, tên nhà vận chuyển, số điện thoại, địa chỉ, tài khoản giao dịch*. Trong quá trình vần chuyển, ShoppePhake cần được theo dõi quá trình một cách chính xác, **chi tiết quá trình vận chuyển** được ShoppePhake quản lí theo: *mã nhà vận chuyển,mã đơn hàng, địa chỉ giao hàng ,ngày bắt đầu giao hàng, thời gian giao, tình trạng giao hàng.*
* ShoppePhaKe cần lưu lại các thông tin giao dịch vào **nhật kí giao dịch của nhà cung cấp** và lưu trữ theo: *mã giao dịch, mã đơn hàng, mã nhà cung cấp, ngày lập giao dịch, trạng thái giao dịch, mã đánh giá, tổng số tiền*. *Tổng số tiền = 98% (tổng tiền của đơn hàng – phí vận chuyển lấy từ bảng “phí vận chuyển của đơn hàng”)*.
* ShoppePhaKe cần lưu lại các thông tin giao dịch vào **nhật kí giao dịch của khách hàng** và được lưu trữ theo: *mã giao dịch, mã đơn hàng, mã khách hàng,ngày lập giao dịch, trạng thái giao dịch, hình thức thanh toán, tổng số tiền*. *Tổng số tiền = tổng tiền của đơn hàng*.
* Đối với các đối tác giao dịch (ngoại trừ thanh toán khi nhận hàng), shoppePhake sẽ lưu trữ **thông tin giao dịch với tài khoản đối tác giao dịch** vào nhật kí bằng : *mã giao dịch, mã đơn hàng,mã đối tác giao dịch, ngày lập giao dịch,tổng số tiền thanh toán, trạng thái thanh toán*. Tổng số tiền thanh = tổng tiền của đơn hàng.
* Đối với các nhà vận chuyển, ShoppepPhake cũng sẽ lưu lại **lịch sử giao dịch với nhà vận chuyển** vào nhật kí*: mã giao dịch, mã đơn hàng, mã nhà vận chuyển, mã chi tiết vận chuyển, ngày lập giao dịch,phí vận chuyển, trạng thái vận chuyển*. Phí vận chuyển = 98% Phí vận chuyển được lấy trong bảng “phí vận chuyển của hóa đơn”.
* Đối với shoppePhake cần lưu trữ lại **phí giao dịch mà shoppePhake nhận được trên mỗi đơn hàng**, được quản lí bởi: *mã giao dịch, mã đơn hàng, mã khách hàng, ngày lập giao dịch, tài khoản giao dịch, phương thức giao dịch, tổng tiền*. Tổng tiền = 2% (tổng tiền của đơn hàng – phí vận chuyển lấy từ bảng “phí vận chuyển của đơn hàng”) + 2% phí vận chuyển.
* Sau khi đơn hàng được giao thành công, shoppePhake sẽ trích tiền thanh toán cho cửa hàng và nhà vận chuyển, sau đó thu hồi tiền ở những tài khoản nhận tiền điện tử, hoặc nếu là phương thức thanh toán tiền khi nhận hàng thì shoppePhake sẽ nhận lại tiền từ nhà vận chuyển( sau khi đã trừ phí vận chuyển ). Những lần giao dịch sẽ được ghi nhận lại.
* Khi khác hàng nhận được đơn hàng, thì khách hàng sẽ tiến hành đánh giá cho đơn hàng đó ( để ghi nhận lại chất lượng sản phẩm ), khi **đánh giá sản phẩm** cần lưu lại các thông tin sau: *mã khách hàng, mã sản phẩm, mã nhà cung cấp, bình luận đánh giá, số lượng sao cho đơn hàng*. Khách hàng có thể đánh giá hoặc không, và số sao tối đa là 5 sao và tối thiểu 1 sao.

Ngoài ra cần lưu trữ thông tin đăng nhập của khách hàng bằng user: username, password.

1. Ràng buộc dữ liệu

* **Nhà cung cấp:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã nhà cung cấp | int | PRIMARY Key |
| Tên nhà cung cấp | nvarchar(50) | NOT NULL |
| Địa chỉ | nvarchar(50) | NOT NULL |
| Số điện thoại | nvarchar(10) | UNIQUE |
| Email | nvarchar(32) | …@... |
| Tài khoản ngân hàng | nvarchar(10) | Đăng ký bằng phone |

* **Khách hàng:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã khách hàng | int | PRIMARY Key |
| Tên khách hàng | nvarchar(32) | NOT NULL |
| Số điện thoại | nvarchar(10) | UNIQUE |
| Email | nvarchar(32) | UNIQUE |
| Địa chỉ | nvarchar(50) | NOT NULL |
| Ngày sinh | date | 1960<ns< ngày hiện tại |
| Giới tính | int(1) | 0: nam – 1: nữ |
| Tài khoản ngân hàng | nvarchar(10) | Đăng ký bằng phone |

* **Sản phẩm:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã sản phẩm | int | PRIMARY Key |
| Mã loại sản phẩm | int | FOREIGN Key |
| Giá bán | int(10) | NOT NULL |
| Số lượng tồn | int(5) | NOT NULL |
| Mô tả sản phẩm | nvarchar(100) | NOT NULL |
| Mã nhà cung cấp | nvarchar(10) | FOREIGN Key |

* **Sự thay đổi giá bán của sản phẩm theo thời gian:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã nhật ký thay đổi | int | PRIMARY Key |
| Mã sản phẩm | int | FOREIGN Key |
| Giá cũ | int(10) | NOT NULL |
| Giá mới | int(5) | NOT NULL |
| Ngày cập nhật | Date time | NOT NULL |
| Mã nhà cung cấp | nvarchar(10) | FOREIGN Key |

**Loại sản phẩm:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã loại sản phẩm | int | PRIMARY Key |
| Tên loại sản phẩm | nvarchar(50) | NOT NULL |

**Đánh giá:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| STT | Int | PRIMARY Key |
| Mã sản phẩm | int | FOREIGN Key |
| Mã nhà cung cấp | nvarchar(10) | FOREIGN Key |
| Bình luận đánh giá | nvarchar(100) | Allow null |
| Đánh giá (1-5 sao) | Int(5) | NOT NULL, 1-5 sao |

**Giỏ hàng:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã thứ tự giỏ hàng | int | PRIMARY Key |
| Mã sản phẩm | int | FOREIGN Key |
| Mã đơn hàng | Nvarchar(10) | Allow null |
| Đơn giá | int(10) | Bằng giá tiền sản phẩm, not null |
| Số lượng | int(5) | NOT NULL <= 100 |
| Thành tiền | int(15) | TT = đơn giá x số lượng |

**Đơn hàng:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã đơn hàng | int | PRIMARY Key |
| Mã khách hàng | nvarchar(50) | FOREIGN Key |
| Mã nhà cung cấp | nvarchar(50) | FOREIGN Key |
| Ngày lập | date | NOT NULL |
| Trạng thái đơn hàng | int(1) | NOT NULL  0/1/2 : Thành công/ Đang giao/ chờ xác nhận |
| Phương thức giao dịch | nvarchar(10) | FOREIGN Key |
| Địa chỉ nhận hàng | nvarchar(50) | NOT NULL |
| Tổng tiền | int(15) | = Tổng tiền tất cả sp trên Cart ứng với mã đơn hàng |

**Phí vận chuyển của đơn hàng:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã đơn hàng | int | PRIMARY Key |
| Mã nhà vận chuyển | nvarchar(10) | FOREIGN Key |
| Phí vận chuyển | int(10) | * HCM,HaNoi, Da Nang = 30k * Nơi khác = 50k |
| Thời gian giao hàng | nvarchar(10) | * HCM,HaNoi, Da Nang = 1-3 ngày * Nơi khác = 3-7 ngày |

**Hình thức thanh toán:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã hình thức thanh toán | int | PRIMARY Key |
| Tên hình thức thanh toán | nvarchar(50) | NOT NULL |

**Đối tác giao dịch:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã đơn vị thanh toán | int | PRIMARY Key |
| Tên đơn vị thanh toán | nvarchar(50) | NOT NULL |
| Mã hình thức thanh toán | nvarchar(10) | FOREIGN Key |
| Ngày hợp tác | date | NOT NULL |
| Số tài khoản giao dịch | nvarchar(10) | UNIQUE |

**Nhà vận chuyển:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã nhà vận chuyển | int | PRIMARY Key |
| Tên nhà vận chuyển | nvarchar(50) | NOT NULL |
| Số điện thoại | nvarchar(10) | NOT NULL |
| Địa chỉ | nvarchar(50) | NOT NULL |
| Phương thức GD | int | NOT NULL |
| Tài khoản giao dịch | nvarchar(10) | Đăng ký bằng phone |

**Chi tiết quá trình vận chuyển:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã đơn hàng | int | PRIMARY Key |
| Mã nhà vận chuyển | nvarchar(10) | FOREIGN Key |
| Địa chỉ giao hàng | nvarchar(50) | NOT NULL |
| Ngày bắt đầu giao | date | Ngày xuất kho |
| Thời gian giao hàng | nvarchar(10) | * HCM,HaNoi, Da Nang = 1-3 ngày * Nơi khác = 3-7 ngày |
| Tình trạng giao hàng | int(1) | NOT NULL  0/1/2 : Thành công/ Đang giao/ chờ xác nhận |

**Nhật ký giao dịch của nhà cung cấp:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã giao dịch | int | PRIMARY Key |
| Mã sản phẩm | int | Foreign Key |
| Mã nhà cung cấp | int | NOT NULL |
| Ngày lập giao dịch | date | NOT NULL |
| Trạng thái giao dịch | int(1) | 0/1 : thành công / thất bại |
| Mã đánh giá | nvarchar(10) | NOT NULL |
| Tổng số tiền | int(15) | *Tổng số tiền = 98% (tổng tiền của đơn hàng – phí vận chuyển lấy từ bảng “phí vận chuyển của đơn hàng* |

**Nhật ký giao dịch của khách hàng:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã giao dịch | int | PRIMARY Key |
| Mã đơn hàng | int | Foreign key |
| Mã khách hàng | int | NOT NULL |
| Ngày lập giao dịch | date | NOT NULL |
| Trạng thái giao dịch | int(1) | NOT NULL |
| Hình thức thanh toán | nvarchar(10) | FOREIGN Key |
| Tổng số tiền | int(10) | Tổng tiền đơn hàng |

**Thông tin giao dịch với tài khoản đối tác giao dịch:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã giao dịch | int | PRIMARY Key |
| Mã đơn hàng | int | Foreign Key |
| Mã đối tác giao dịch | int | FOREIGN Key |
| Ngày lập giao dịch | date | NOT NULL |
| Tổng số tiền thanh toán | int(15) | Tổng tiền đơn hàng ( tiền chuyển về tài khoản của shopee trên từng đối tác thanh toán ) |
| Trạng thái thanh toán | int(1) | 0/1: thanh toán/ chưa thanh toán |

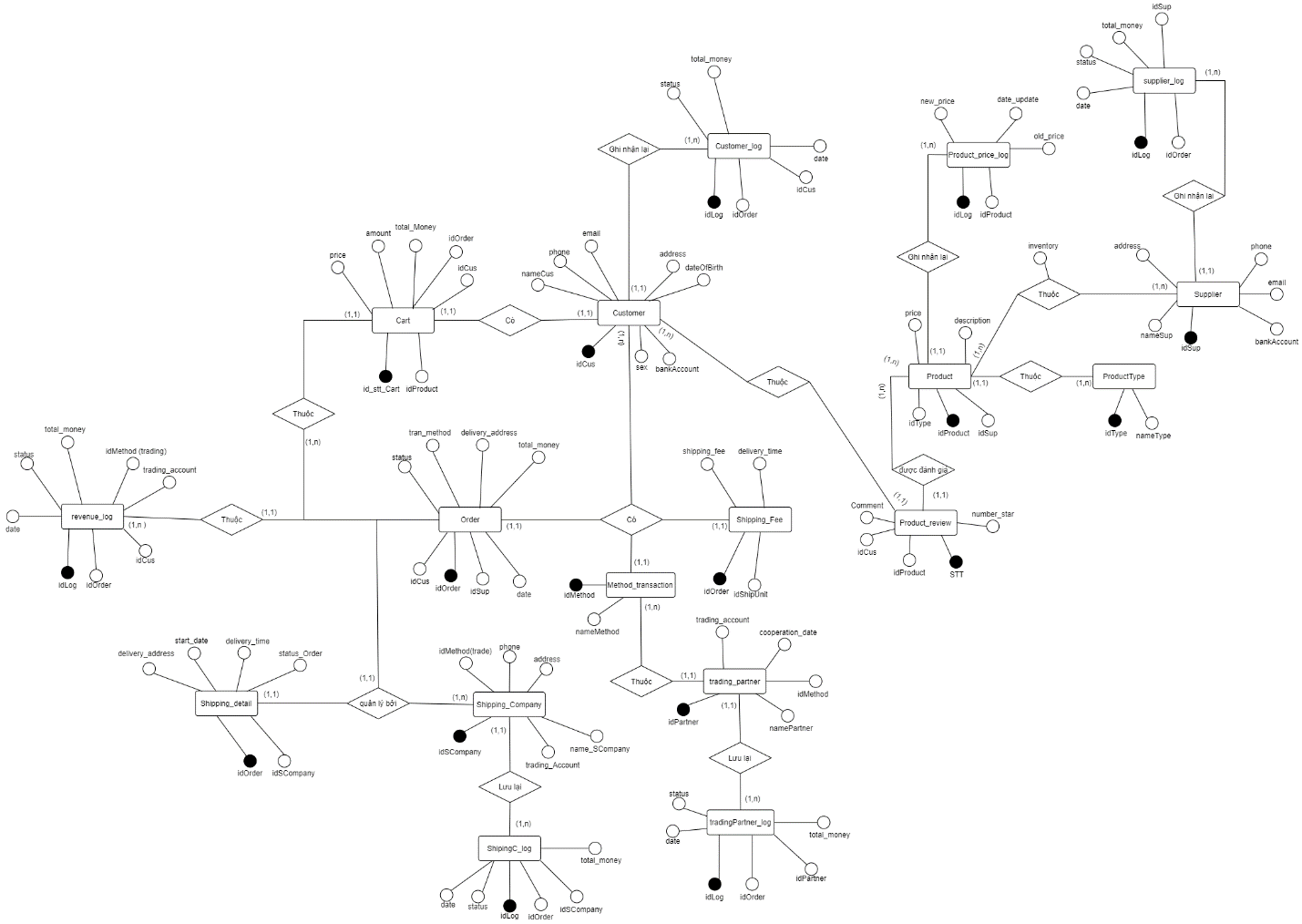
**Lịch sử giao dịch với nhà vận chuyển:**

|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã giao dịch | nvarchar(10) | PRIMARY Key |
| Mã đơn hàng | nvarchar(10) | PRIMARY Key |
| Mã nhà vận chuyển | nvarchar(10) | FOREIGN Key |
| Ngày lập giao dịch | date | NOT NULL |
| Phí vận chuyển | int(10) | Phí vận chuyển = 98% Phí vận chuyển được lấy trong bảng “phí vận chuyển của hóa đơn”. |
| Trạng thái vận chuyển | int(1) | 0/1: thanh toán/ chưa thanh toán |

**Phí giao dịch ShopeePhaKe nhận được trên mỗi đơn hàng:**

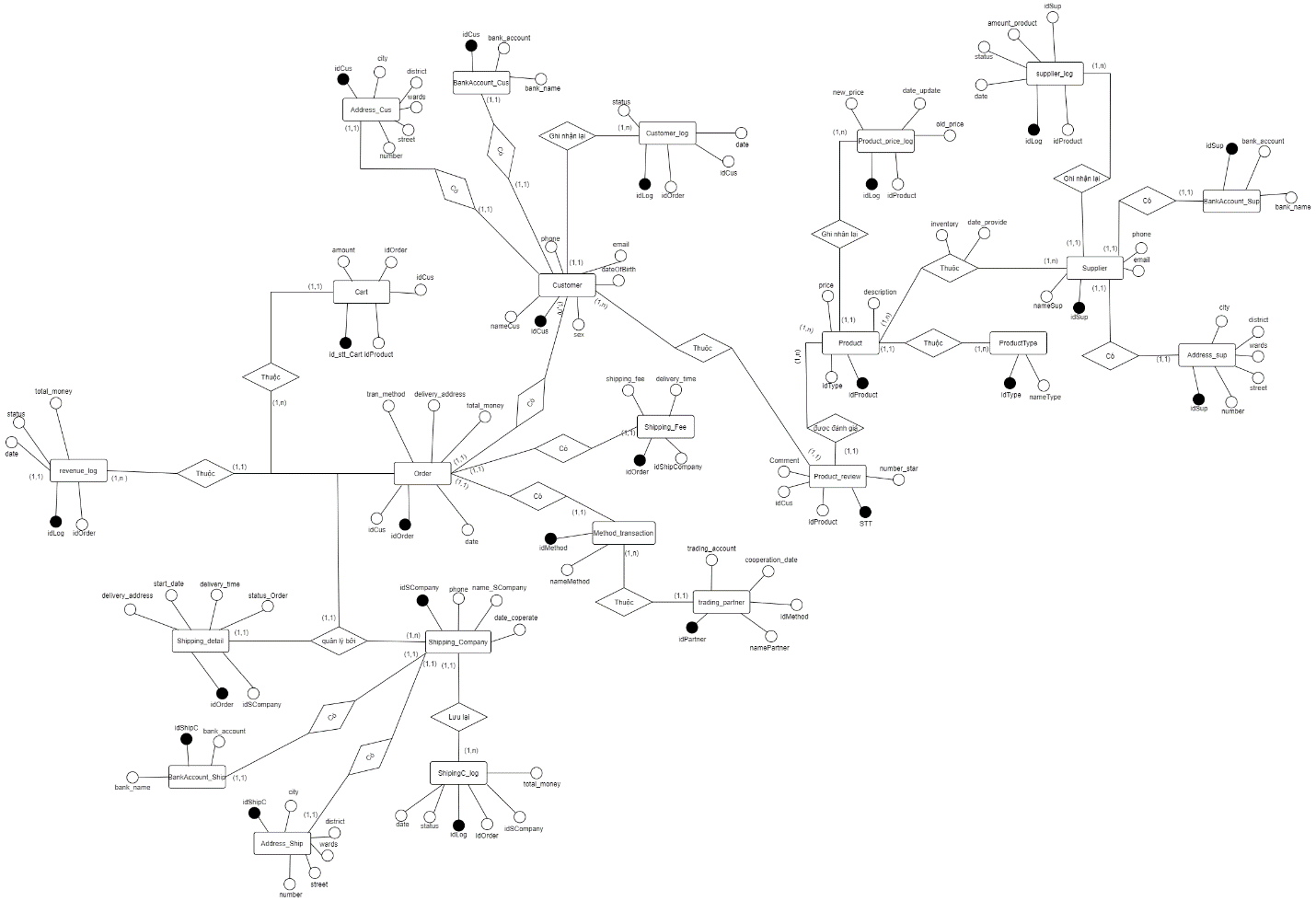
|  |  |  |
| --- | --- | --- |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Mã giao dịch | nvarchar(10) | PRIMARY Key |
| Mã đơn hàng | nvarchar(10) | PRIMARY Key |
| Mã khách hàng | nvarchar(10) | FOREIGN Key |
| Ngày lập giao dịch | date | NOT NULL |
| Tài khoản giao dịch | Nvarchar(50) | NOT NULL |
| Phương thức GD | Nvarchar(10) | NOT NULL |
| Tổng tiền | int(15) | Tổng tiền = 2% (tổng tiền của đơn hàng – phí vận chuyển lấy từ bảng “phí vận chuyển của đơn hàng”) |

1. Thiết kế dữ liệu logic
2. Mô hình ER



Mô hình ER được xây dựng dựa trên cách thức vết dầu loang, với trung tâm là Product, Supplier, Product type, Product review, Customer.

1. Mô hình quan hệ



1. Chuẩn hóa dữ liệu:

* Nhà cung cấp:

Supplier (idSup, nameSup, address, phone, email, bankAccount) không thuộc dạng chuẩn.

Trong bảng này có 2 trường là thuộc tính đa trị : address và bankAccount

* + Bảng mới sau khi nâng chuẩn sẽ là

Supplier (idSup, nameSup, phone, email)

Address\_Sup (idSup, number, street, wards, district, city )

BankAccount\_Sup(idSup, bank\_account\_number, bank\_name)

* + Bảng Supplier sẽ ở dạng chuẩn 3NF :
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Khách Hàng

Customer (idCus, nameCus, phone, email, address, dateOfBirth, sex, bankAccount) không thuộc dạng chuẩn.

Trong bảng này có 2 trường là thuộc tính đa trị : address và bankAccount

* + Bảng mới sau khi nâng chuẩn sẽ là

Customer (idCus, nameCus, phone, email, dateOfBirth, sex)

Address\_Cus (idCus, number, street, wards, district, city )

BankAccount\_Cus (idCus, bank\_account\_number, bank\_name)

* + Bảng Customer sẽ ở dạng chuẩn 3NF :
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Sản phẩm

Product (idProduct, nameProduct, price, description, idType)

* + Bảng đang ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Cung Cấp

Provide (idSup, idProduct, inventory)

* + Bảng Provide sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Sự thay đổi giá bán của sản phẩm

Product\_price\_log(idLog, idProduct, current\_price, new\_price, date\_update) ở dạng chuẩn 1.

* + Dữ liệu không cần nâng chuẩn
* Loại sản phẩm

ProductType (idType, nameType)

* + ở dạng 3NF
* Đánh giá

Product\_review (STT, idCus, idProduct ,comment, number\_star)

* + ở dạng chuẩn 3NF:
    - là 2NF
    - không có phụ thuộc hàm bắc cầu
* Nhật ký giao dịch của nhà cung cấp

Supplier\_log (idLog, amount\_product, date, status**, idSup, idProduct**)

* + thuộc chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc bắc cầu
* Nhật ký giao dịch của khách hàng

Customer\_log (idLog, status, date, **idCus, id Order)**

* + Dạng chuẩn 3NF:
    - là 2NF
    - không có phụ thuộc hàm bắc cầu
* Giỏ hàng

Cart (id\_stt\_card, idProduct, price, amount, total\_money, **idOrder, idCust)** chưa đạt dạng chuẩn.

* + Chuẩn hóa 1NF: bỏ đi total\_money bởi vì có thể tính toán được từ amount và price

Cart (id\_stt\_card, idProduct, price, amount, **idOrder, idCust)**

* + Chuẩn hóa 2NF:

{idProduct} -> {price}

Cart (id\_stt\_card, idProduct, amount, **idOrder, idCust)**

* + Chuẩn hóa 3NF:

Có quan hệ bắc cầu

{id\_stt\_card} -> {idOrder}

{idOrder} -> {idCus}

Cart (id\_stt\_card, idProduct, amount, **idOrder)**

* + **Vậy sau khi chuẩn hóa bảng Card có dạng 3NF**

Cart (id\_stt\_card, idProduct, amount, **idOrder)**

* Đơn hàng

Order (idOrder, date, delivery\_address, total\_money, **tran\_method ,idCus**) chuẩn 3NF, vì:

* + - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Phí vận chuyển

Shipping\_Fee(**idOrder**, idSCompany, shipping\_fee, delivery\_time) chuẩn 3NF, vì:

* + - Là 2NF
    - Không có phụ thuộc bắc cầu
* Hình thức thanh toán

Method\_payment(**idMethod**, nameMethod) chuẩn 3NF

* Đối tác giao dịch

trading\_partner(**idPartner**, trading\_account, cooperation\_date, namePartner, idMethod)

* + Dạng chuẩn 3NF
* Đơn vị vận chuyển

Shipping\_Company(**idSCompany**, phone, address, name\_Scompany, bankAccount)

Trong bảng này có 2 trường là thuộc tính đa trị : address và bankAccount

* + Bảng mới sau khi nâng chuẩn sẽ là

Shipping\_Company(**idSCompany**, phone, name\_Scompany)

Address\_ShipCompany (idSCompany, number, street, wards, district, city )

BankAccount\_ShipCompany (idSCompany, bank\_account\_number, bank\_name)

* + Bảng Shipping\_Company sẽ ở dạng chuẩn 3NF :
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Chi tiết quá trình vận chuyển

 Shipping\_detail(idOrder**,** idSCompany, delivery\_address, start\_date, delivery\_time, status\_Order)

* + Bảng chi tiết vận chuyển đạt chuẩn 3NF
    - Là 2NF
    - Không có phụ thuộc hàm bắt cầu
* Lịch sử giao dịch với nhà vận chuyển

ShippingC\_log(idLog, idSCompany, idOrder, total\_money, date, status) chuẩn 3NF

* Phí giao dịch shoppe nhận được trên mỗi đơn hàng

Revenue\_log (idLog, idOrder, idCus, idMethod, trading\_account, date, total\_money, status)

Chuẩn 1NF

* + Nâng chuẩn 2NF:

{idLog} -> {idOrder, idCus, idMethod, trading\_account, date, total\_money, status}

{idOrder} -> {idCus, idMethod}

Bảng sau khi loại bỏ dữ liệu thừa:

Revenue\_log (idLog, idOrder, trading\_account, date, total\_money, status)

Cus\_tran\_log (idOrder, idCus, method)

* + Bảng sau khi chuẩn hóa đạt dạng 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu

1. **Thiết kế vật lý (thông tin tần suất, cài chỉ mục, phân tích, kiểm chứng hiệu quả chỉ mục…)**
2. Thông tin tần suất

|  |  |  |
| --- | --- | --- |
| **Chức năng** | **Mục tiêu** | **Tuần suất (lượt truy vấn)** |
| Tìm kiếm thông tin nhà cung cấp | Tìm theo tên hoặc số điện thoại của nhà cung cấp | 1M + / ngày |
| Tìm kiếm thông tin khách hàng | Tìm theo tên hoặc số điện thoại của khách hàng | 1M+/ ngày |
| Tìm kiếm sản phẩm | Tìm theo tên sản phẩm hoặc tìm theo loại sản phẩm | 10M+/ ngày |
| Kiểm tra sự thay đổi giá của sản phẩm | Tìm theo ngày thay đổi hoặc mã sản phẩm | 500K/ tháng |
| Hiển thị đánh giá sản phẩm | Hiển thị đánh gia sản phẩm theo số sao, mã sản phẩm và mã khách hàng | 10M+/ ngày |
| Hiển thị thông tin giỏ hàng | Hiển thị giỏ hàng theo mã khách hàng | 500k/ ngày |
| Hiển thị đơn hàng | Hiển thị theo mã khách hàng hoặc ngày tạo order | 500k/ngày |
| Phí vận chuyển đơn hàng | Tìm theo mã đơn vị vận chuyển (idOrder làm khóa chính) | 500k/ngày |
| Hiển thị lịch sử của giao dịch NCC | Hiển thị theo mã nhà cung cấp và ngày tạo log | 500K/ ngày |
| Hiển thị lịch sử giao dịch của KH | Tìm theo mã khách hàng và ngày tạo log | 500K/ ngày |
| Hiển thị lịch sử của nhà vận chuyển | Tìm theo mã Nhà vận chuyển và ngày tạo log | 500K/ ngày |
| Hiển thị doanh thu mà shopee thu được | Tìm kiếm theo ngày tạo và mã Order | 500K/ ngày |

1. Chỉ mục

use ShopeeIndex

--- create index -----

-- NCC --

create index idx\_nameSup

on Supplier(nameSup)

create index idx\_phoneSup

on Supplier(phone)

-- Khach Hang --

create index idx\_nameCus

on Customer(nameCus)

create index idx\_phoneCus

on Customer(phone)

-- San pham ---

create index idx\_nameProduct

on Product(nameProduct)

create index idx\_idType

on Product(idType)

select \* from Product\_price\_log

-- Price Log --

create index idx\_update\_price\_log

on Product\_price\_log(date\_update)

-- loai sp --

create index idx\_nameTypeProduct

on ProductType(nameType)

-- danh gia --

create index idx\_idCus\_review

on Product\_review(idCus)

create index idx\_danh\_gia

on Product\_review(number\_star)

-- gio hang --

create index idx\_idCus\_cart

on Product\_review(idCus)

USE [ShopeeIndex]

GO

CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>]

ON [dbo].[Cart] ([idCus])

--- don hang --

create index idx\_idCus\_Order

on Order\_Table(idCus)

CREATE INDEX idx\_date\_order

ON Order\_Table(date);

USE [ShopeeIndex]

GO

CREATE NONCLUSTERED INDEX [<Card\_idOrder>]

ON [dbo].[Cart] ([idOrder])

-- Phi van chuyen don hang --

CREATE INDEX idx\_ShipUnit\_fee

ON Shipping\_fee (idSCompany);

USE [ShopeeIndex]

GO

CREATE NONCLUSTERED INDEX [<fee\_idOrder>]

ON [dbo].[Shipping\_Fee] ([idOrder],[idSCompany])

-- hinh thuc thanh toan --

CREATE INDEX idx\_nameMethod

ON Method\_payment(nameMethod);

-- doi tac giao dich -- DATA IT NEN KHONG CAN

--CREATE INDEX idx\_namePartner

--ON trading\_partner (namePartner);

--CREATE INDEX idx\_idMethod\_partner

--ON trading\_partner(idMethod);

-- nha van chuyen -- DATA NHO NEN KHONG CAN

CREATE INDEX idx\_nameSCompany

ON Shipping\_Company(name\_SCompany);

-- NCC log --

CREATE INDEX idx\_idSup\_SLog

on supplier\_log (idSup);

CREATE INDEX idx\_date\_Slog

On supplier\_log(date);

-- khach hang log --

CREATE INDEX idx\_idCus\_Clog

on Customer\_log(idCus);

CREATE INDEX idx\_date\_CLog

on Customer\_log(date);

-- nha vvan chuyen log --

CREATE INDEX idx\_Shipping\_Log

on ShippingC\_log(idSCompany);

CREATE INDEX idx\_date\_ShLog

on ShippingC\_log(date);

-- doanh thu --

CREATE INDEX idx\_date\_revenue

on revenue\_log(date);

Create index idx\_idOder\_revenue

on revenue\_log(idOrder)

select \* from revenue\_log

1. Phân tích
   * + Tìm kiếm thông tin nhà cung cấp theo tên hoặc số điện thoại: do tần suất tìm kiếm cửa hàng trên shopee là cao, nên ta cài index để cải thiện performance cho hệ thống
     + Tìm kiếm thông tin khách hàng theo tên hoặc số điện thoại: do tần suất tìm kiếm khách hàng để tạo order hoặc xem thông tin trên shopee là cao, nên ta cài index để cải thiện performance cho hệ thống.
     + Tìm kiếm sản phẩm là chức năng có lượt truy vấn nhiều nhất trên hệ thống, vì vậy phải cải thiện performance cho việc truy vấn liên quan đến sản phẩm
     + Việc kiểm tra sự thay đổi về giá của sản phẩm bằng ngày thay đổi và mã sản phẩm với mục đích quản lí doanh thu cho cửa hàng cũng như hệ thống, vậy chức năng này có thể thực hiện hằng ngày hoặc hàng tháng, năm. Vậy chúng ta cần cài index ở bảng này để cải thiện performance.
     + Cài đặt index cho bảng loại sản phẩm dùng để hiển thị sản phẩm theo tên loại sản phẩm, và chức năng này được thực hiện ngay khi khách hàng vào trang web, vì vậy khi cài index ở bảng này sẽ cải thiện được performance cho hệ thống.
     + Bảng đánh giá sản phẩm có tần suất truy vấn lớn đi kèm với tra thông tin sản phẩm , khi lấy thông tin sản phẩm thì đồng thời phải xuất hiện thông tin khách hàng đánh giá, comment , số sao đánh giá. Vậy cần phải thực hiện cài đặt index ở bảng này nhằm mục đích cải thiện performance đi kèm với tra thông tin sản phẩm.
     + Hiển thị thông tin giỏ hàng: cài index ở idCus để có thể kiểm tra khách nào đã thêm gì vào bảng, và dùng để đưa sản phẩm ra Order một cách nhanh chóng.
     + Hiển thị đơn hàng: hiển thị theo idCus hoặc ngày tạo order dùng để kiểm soát thông tin order của khách trong việc thanh toán hoặc kiểm tra doanh thu, lịch sử giao dịch của khách hàng.
     + Hiển thị phí vận chuyển của đơn hàng: tìm theo idShipCompany bởi vì cần xem thử đơn vị vận chuyển nào nhận chuyển hàng cho đơn hàng nào. Việc sử dụng này được sử dụng thường xuyên khi khách muốn xem thông tin chi tiết đơn hàng mà mình đã đặt.
     + Các bảng kiểm tra lịch sử giao dịch được admin và nhà cung cấp xem lại hàng ngày/tháng/năm để kiểm soát doanh thu và cải thiện việc bán hàng. Vì vậy cần cài đặt index ở các bảng này để việc truy suất dữ liệu được nhanh hơn (do dữ liệu lưu trữ rất lớn).
2. Kiểm chứng hiệu quả chỉ mục

Kích thước dữ liệu dùng kể kiểm chứng hiệu quả của chỉ mục

1. nhà cung cấp : 150.000 dòng dữ liệu

2. khách hàng : 500.000 khách hàng

3. sản phẩm : 1.000.000 sản phẩm

4. sự thay đổi giá : 750.000 sự thay đổi

5. loại sản phẩm: 16 loại

6. Đánh giá : 1.000.000 dòng đánh giá

7. giỏ hàng: 5.000.000 dòng với khách hàng khác nhau

8. Đơn hàng: 1 000 000 dòng

9. Phí vận chuyển: 1 000 000 dòng

10. Hình thức thanh toán: 3 hình thức

11. Đối tác giao dịch: 12 đối tác

12. nhà vận chuyển: 5 nhà vận chuyển

13. Chi tiết quá trình vận chuyển : 1 000 000 dòng

14. log nhà cung cấp: 1 000 000

15. log khách hàng: 1 000 000

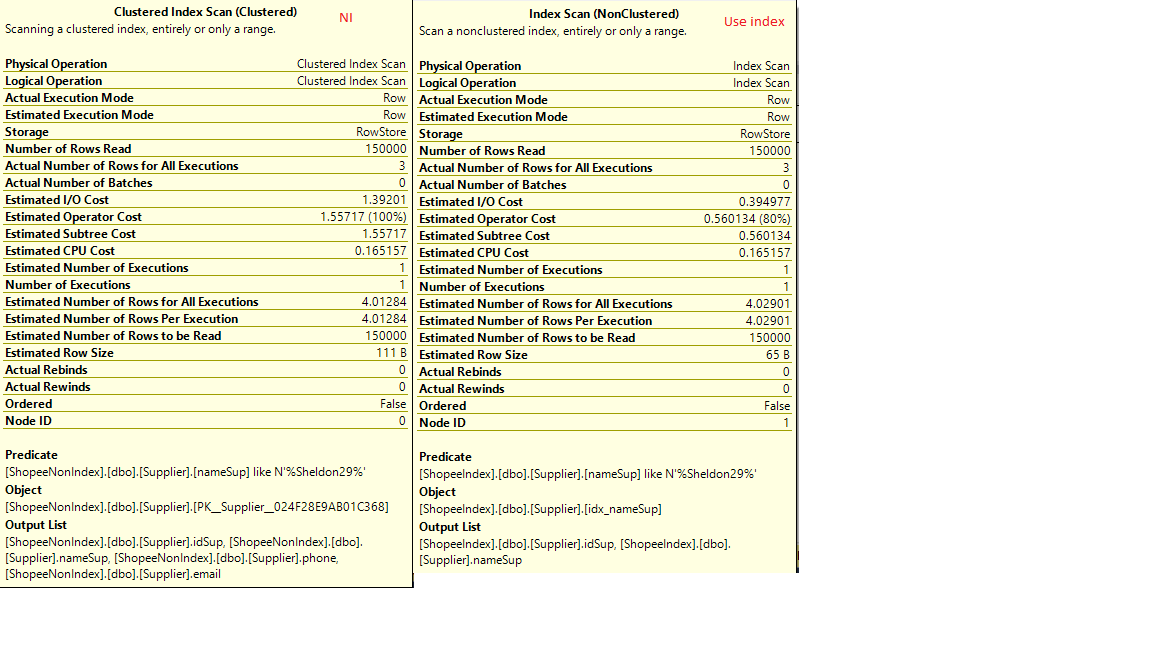
17. log nhà vận chuyển: 1 000 000

18. log phí giao dịch shoppe trên mỗi đơn hàng: 1 000 000

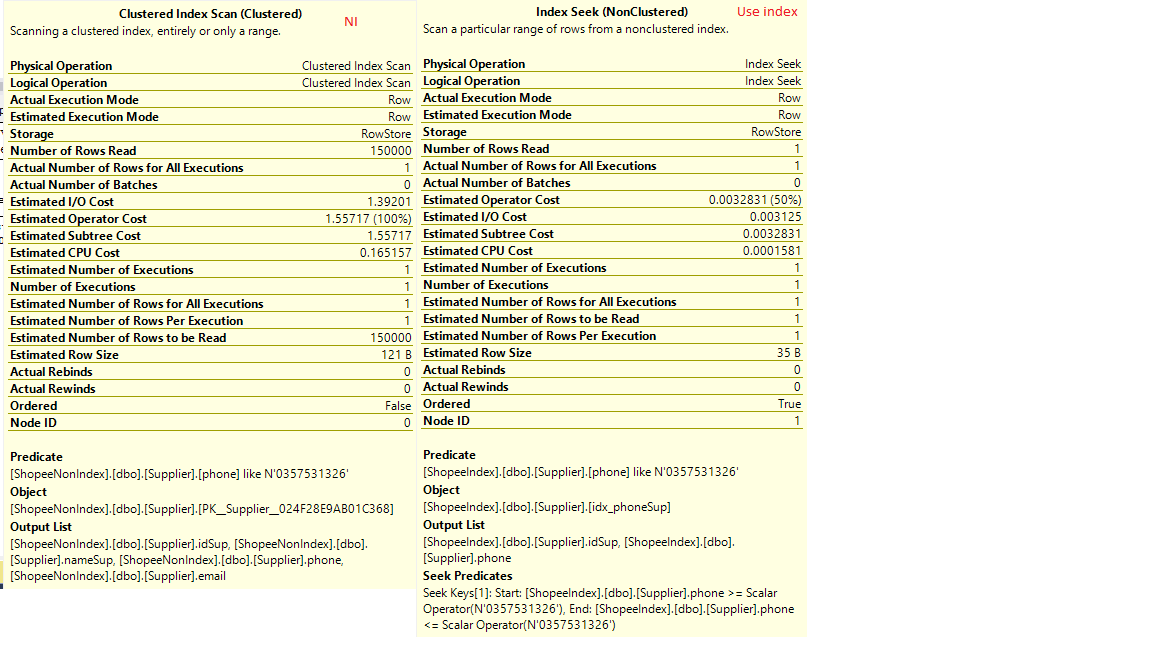
19. user : 650 000

20. Cung cấp: 1 000 000

* Nhà cung cấp

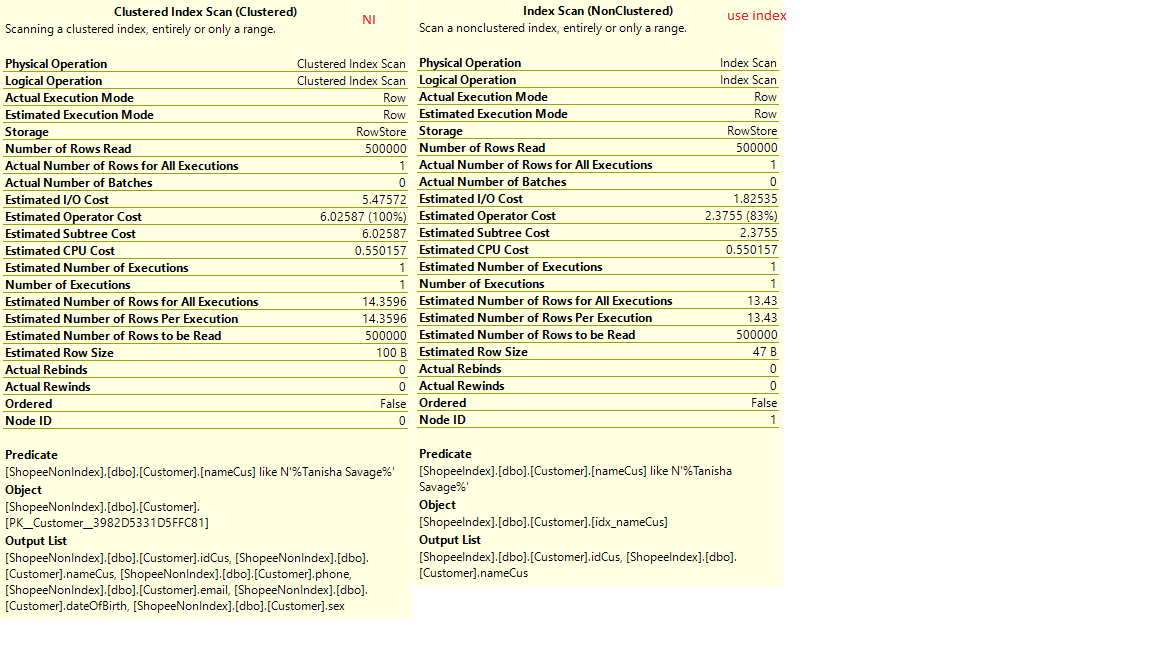
select \* from Supplier where nameSup like 'Sheldon29%'

select \* from Supplier where phone like '0357531326'

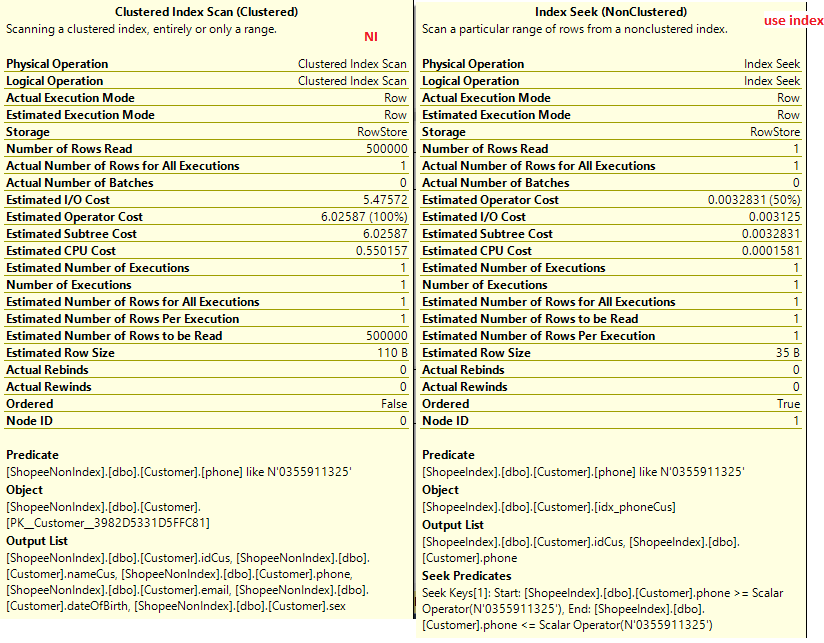


* Khách hàng

select \* from Customer where nameCus like 'Tanisha Savage%'

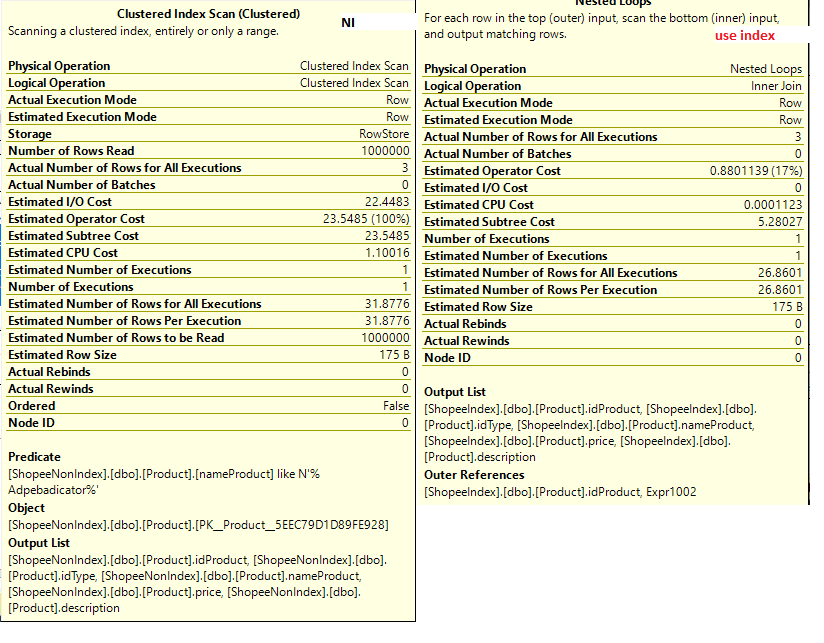


select \* from Customer where phone like '0355911325'



* Sản phẩm:

select \* from Product where nameProduct like 'Adpebadicator%'

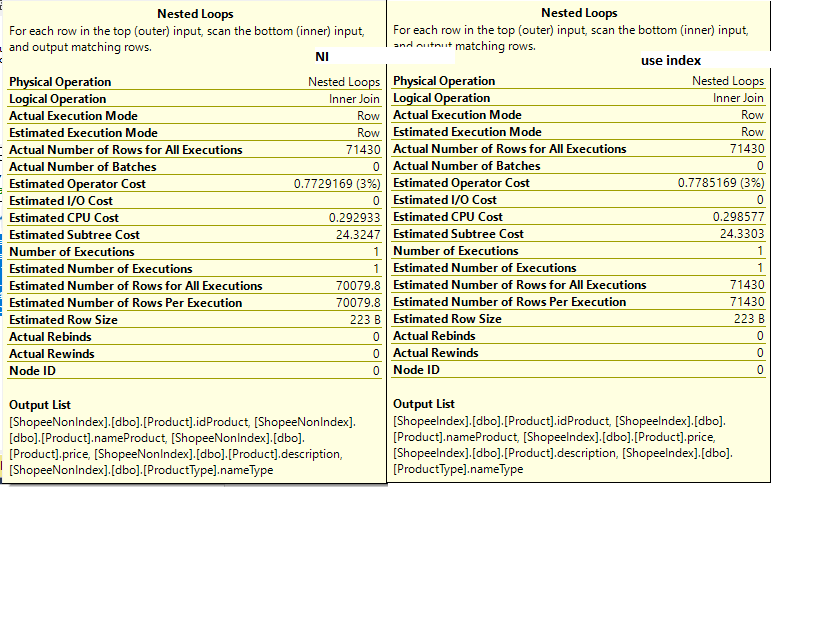


select pt.nameType,p.idProduct, p.nameProduct,p.price,p.description from Product p

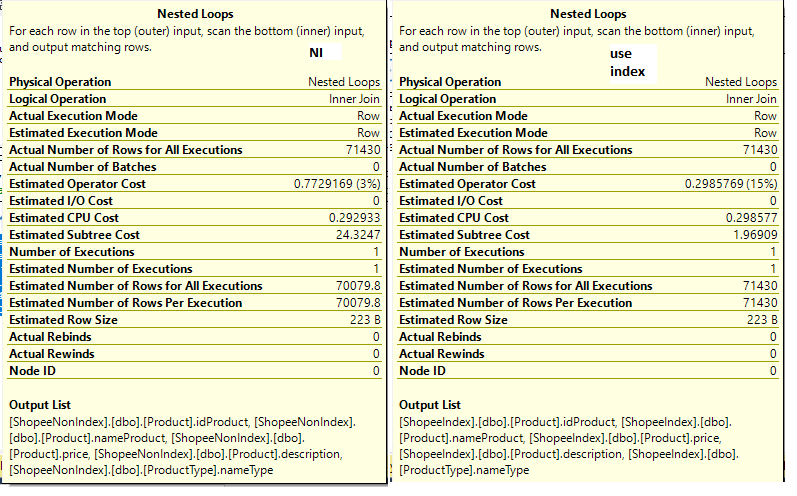
inner join ProductType pt

on p.idType = pt.idType

where p.idType = 7



Index recommend



* Kiểm tra sự thay đổi về giá sản phẩm

select s.idSup,s.nameSup,p.nameProduct,pl.old\_price,pl.new\_price,pl.date\_update

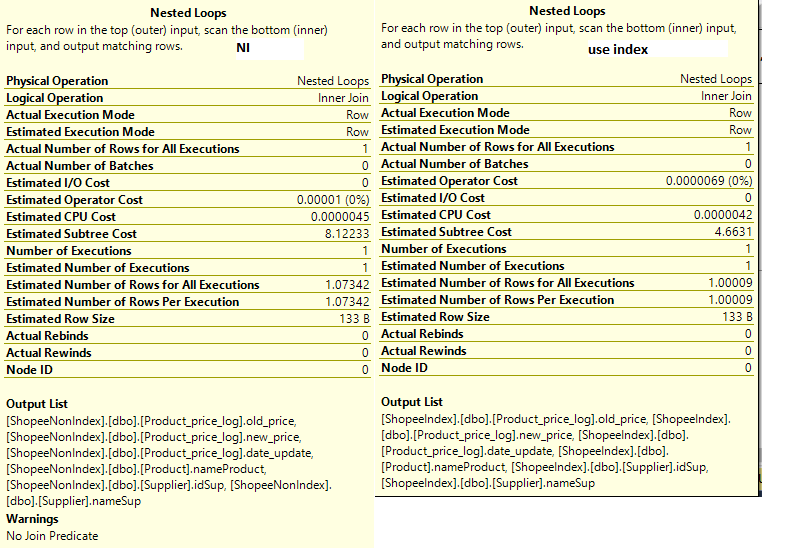
from Product\_price\_log pl

inner join Product p on p.idProduct = pl.idProduct

inner join Provide on Provide.idProduct = pl.idProduct

inner join Supplier s on s.idSup = Provide.idSup

where pl.idProduct = 620075



* Hiển thị đánh giá sản phẩm

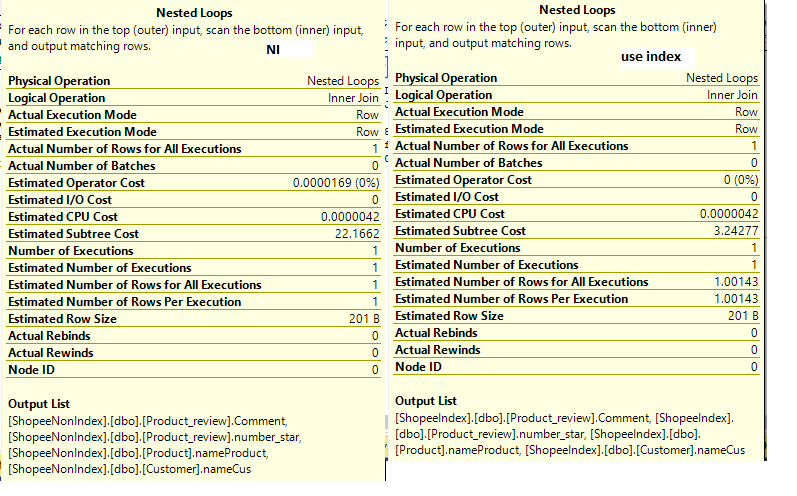
select p.nameProduct,c.nameCus, pr.Comment,pr.number\_star

from Product\_review pr

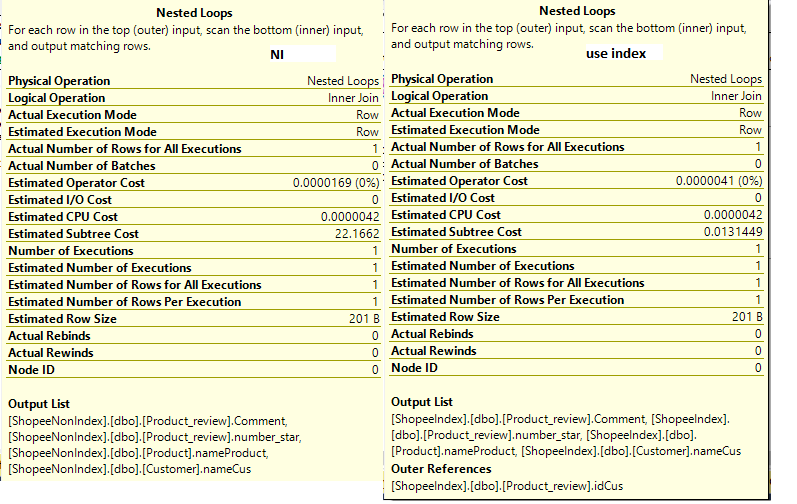
inner join Product p on p.idProduct = pr.idProduct

inner join Customer c on c.idCus = pr.idCus

where pr.idProduct = 164146



Recommend index



* Hiển thị thông tin giỏ hàng

select cus.nameCus, p.nameProduct, ca.amount from Cart ca

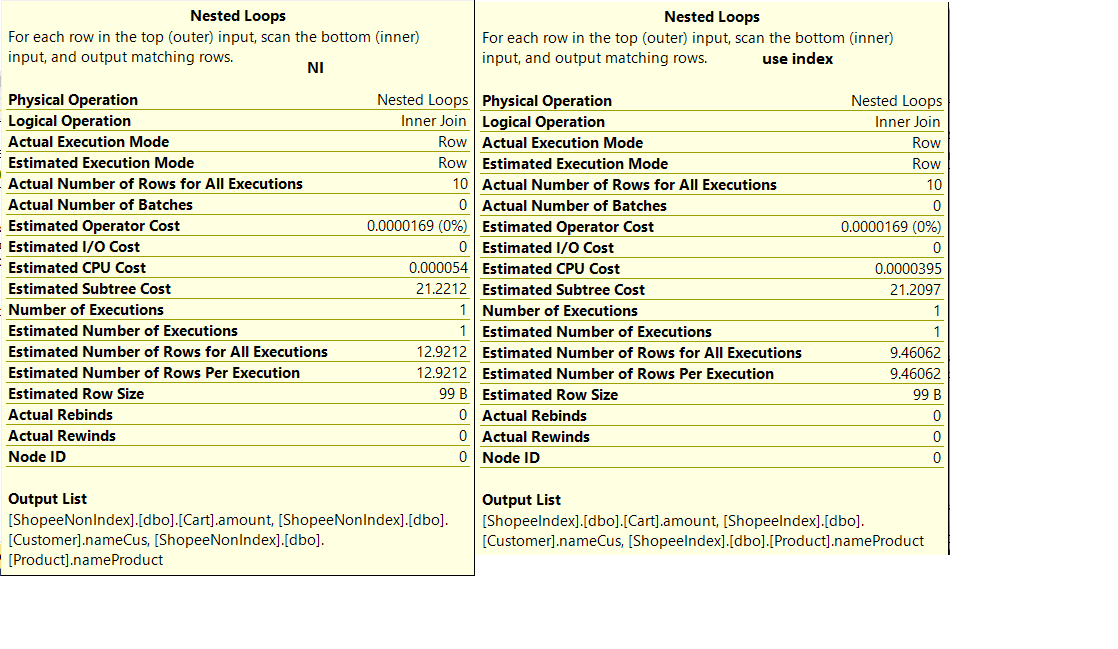
inner join Customer cus

on ca.idCus = cus.idCus

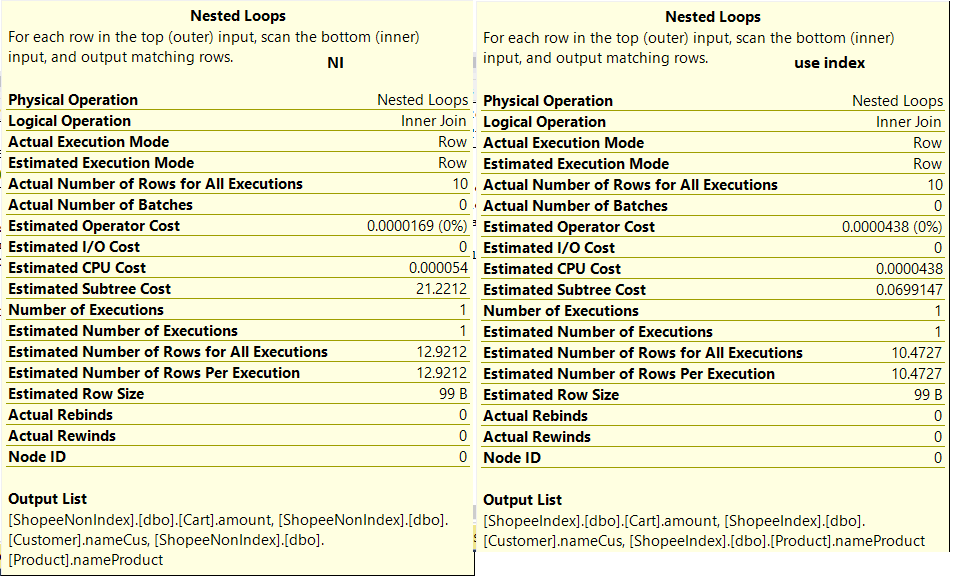
inner join Product p

on p.idProduct = ca.idProduct

where cus.idCus = 840



Recommend index:



* Hiển thị đơn hàng

select ot.idOrder, ot.date, cus.nameCus, p.nameProduct,

ot.delivery\_address,sc.name\_SCompany,sf.shipping\_fee ,

ot.total\_money + shipping\_fee as Total

from Order\_Table ot

inner join Customer cus

on cus.idCus = ot.idCus

inner join Shipping\_Fee sf

on sf.idOrder = ot.idOrder

inner join Cart ca

on ca.idOrder = ot.idOrder

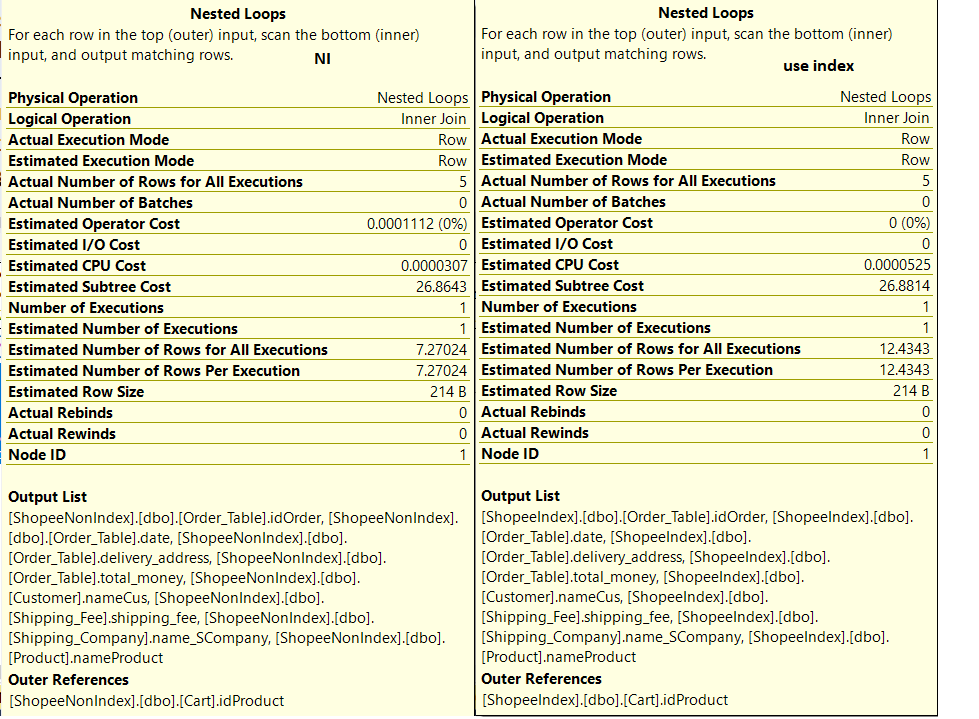
inner join Shipping\_Company sc

on sc.idSCompany = sf.idSCompany

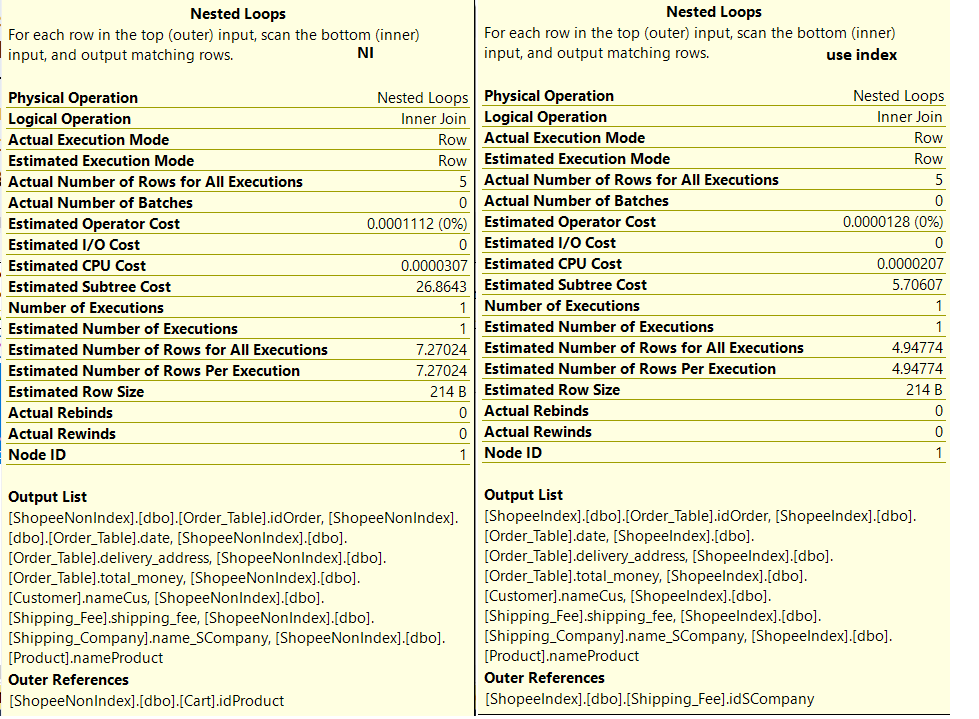
inner join Product p

on p.idProduct = ca.idProduct

where ot.idOrder = 264639



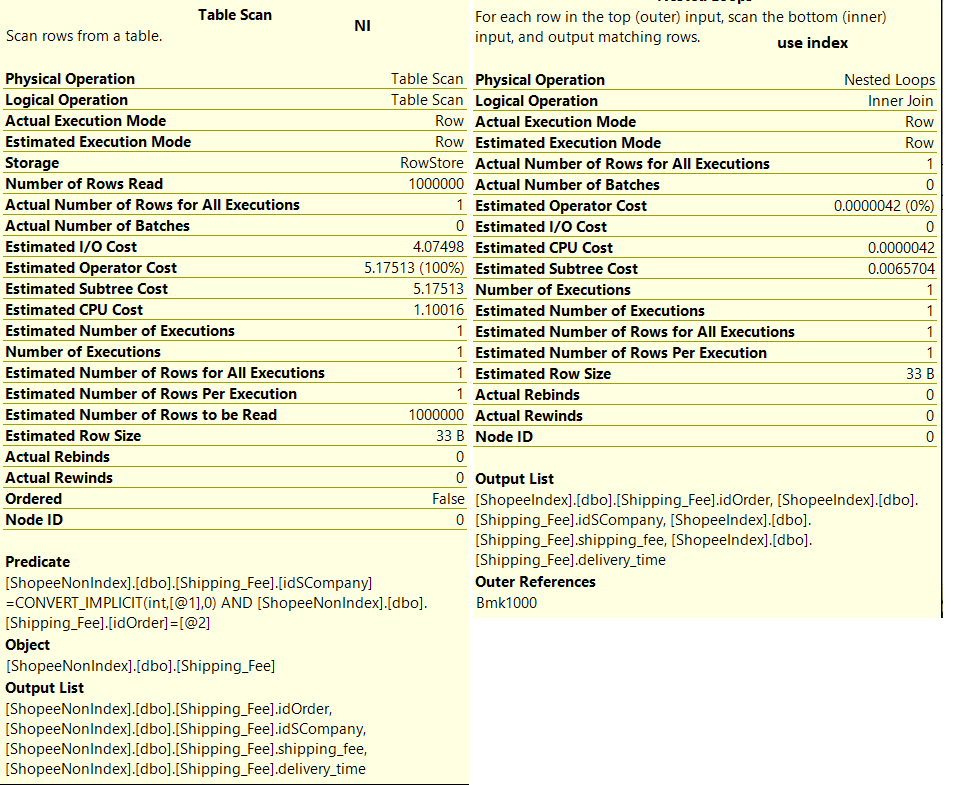
Recommend index



* Phí vận chuyển của đơn hàng

select \* from Shipping\_Fee

where idSCompany = 3 and idOrder = 897913



* Hiển thị lịch sử giao dịch của NCC

select p.nameProduct, sl.date,

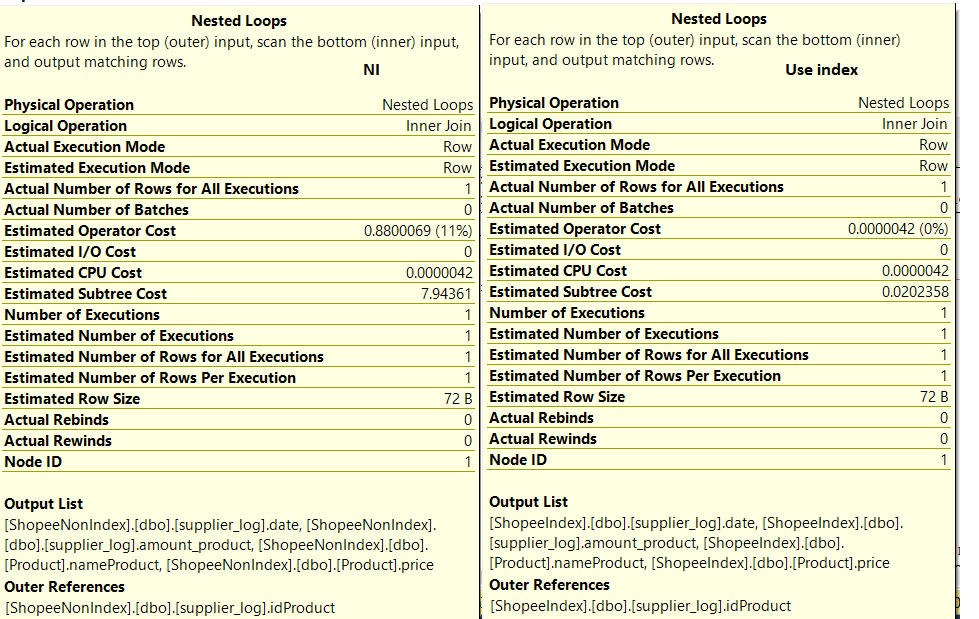
p.price \* amount\_product as revenue

from supplier\_log sl

inner join Product p

on p.idProduct = sl.idProduct

where date = '2020-01-21' and sl.idSup = 1440



* Hiển thị lịch sử giao dịch của khách hàng

select \* from Customer\_log cl

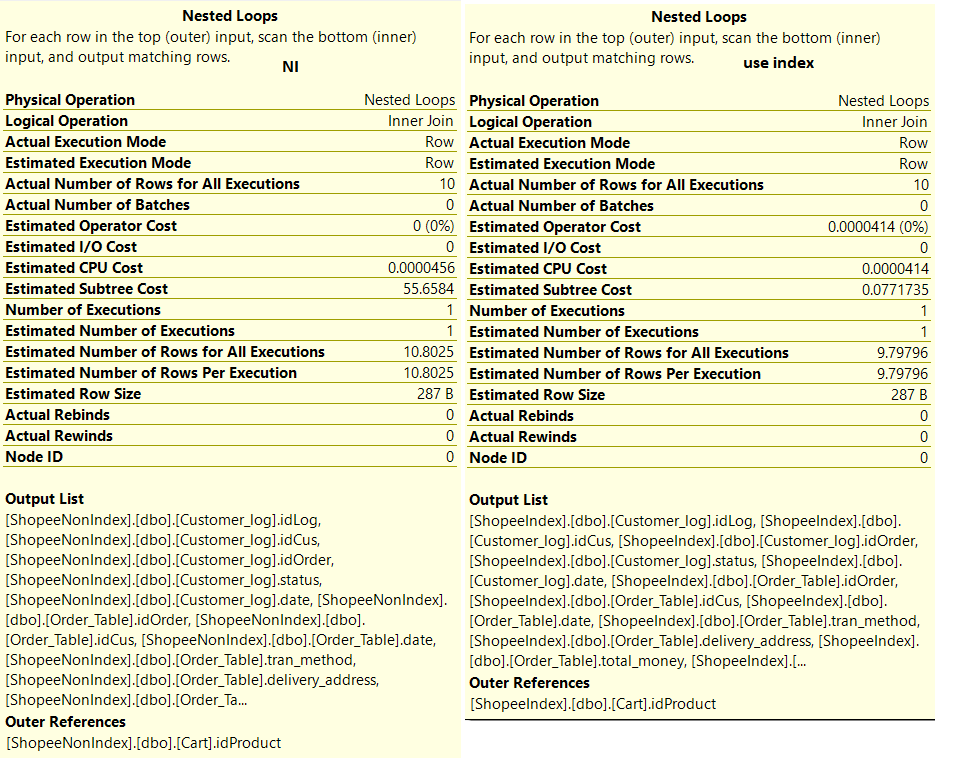
inner join Order\_Table ot

on cl.idOrder = ot.idOrder

inner join Cart ca on ca.idOrder = ot.idOrder

inner join Product p on p.idProduct = ca.idProduct

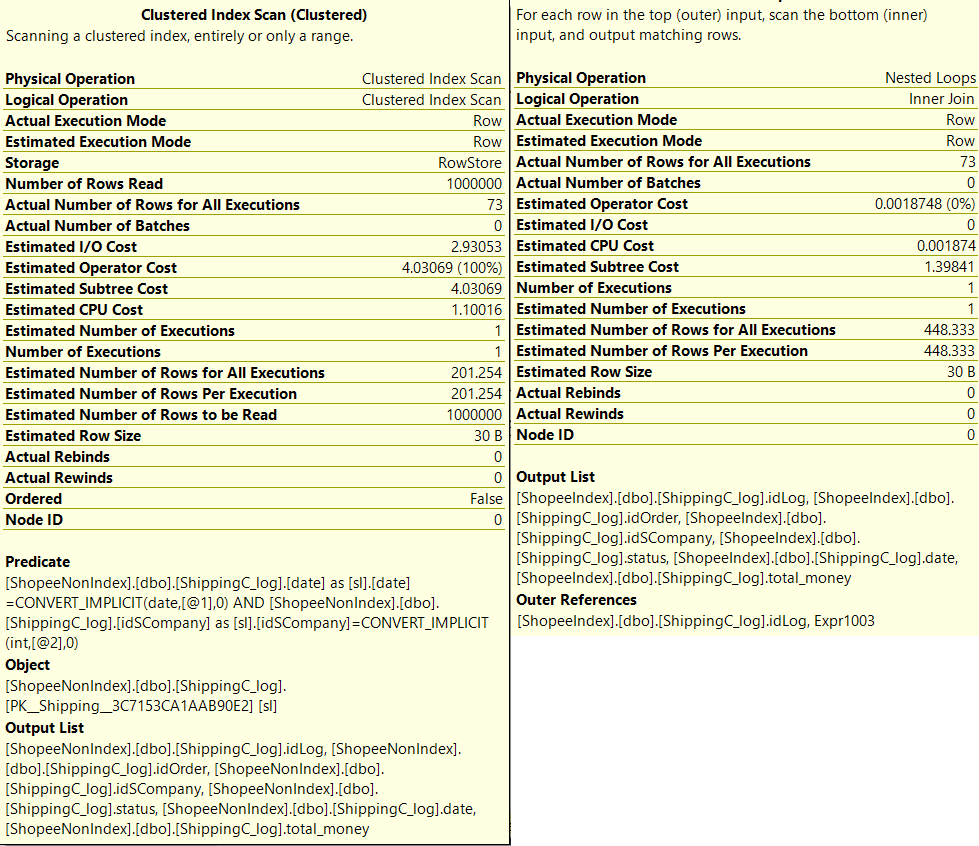
where cl.idCus = 282916



* Hiển thị lịch sử giao dịch với nhà vận chuyển

select \* from ShippingC\_log sl

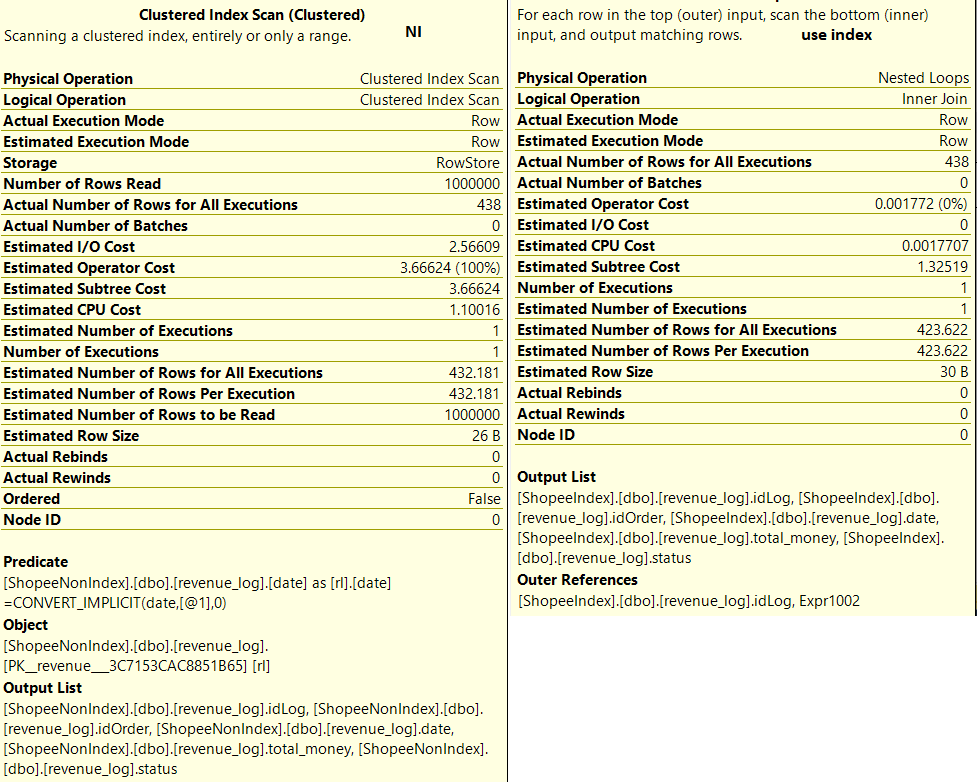
where sl.date = '2015-11-03' and sl.idSCompany = 2



* Hiển thị doanh thu mà shopee thu được

select \* from revenue\_log rl

where rl.date = '2020-10-21'



***Đánh giá tổng:***

* Sử dụng index cho kết quả truy vấn tốt hơn, nhanh hơn so với khi không dùng.
* Cải thiện tốc độ khi phải kết nhiều bảng với nhau
* Khi sử dụng index thì các chi phí bỏ ra để truy vấn kết quả đã được giảm đi 1 nửa (sub tree, )
* Những chỉ số khác như I/O, Operator, CPU Cost được giảm đi gần 9 lần (ví dụ ở bảng Hiển thị order của khách hàng: Operator cost được giảm đến 8.6 lần khi cài đặt index)
* Vậy việc sử dụng index là cần thiết trong việc vận hành hiệu quả ứng dụng, hệ thống, cải thiện tốc độ cũng như hiệu suất truy xuất dữ liệu
* Khi đánh index cho column có kiểu dữ liệu là string thì cẩn thận với toán tử like : ví dụ với index cho column name trong bảng Customer, nếu truy vấn dạng như WHERE name LIKE 'Xo%' hoặc 'X%n' thì index sẽ vẫn hoạt động, Nhưng, nếu với kiểu truy vấn WHERE name LIKE '%n' thì index không được sử dụng hiệu quả.
* Index sẽ không có nhiều tác dụng trong các trường hợp còn lại với toán tử <> (!=) hay NOT IN
* **Không nên** sử dụng các toán tử đối với các trường đánh index: ví dụ SELECT \* FROM Product WHERE price + 1 = 100 (lấy ra tất cả các giá sau 100) lúc này index sẽ không được apply, vì nó không đủ thông minh để hiểu được là cần phải tìm kiếm các records với age = 100
* Như đã nói để trên, khi insert cần mất thêm thời gian cho việc reIndex thế nên trong một vài trường hợp, bạn có thể bỏ index --> insert data --> sau khi insert hoàn thành thì chúng ta tạo lại index, đây cũng là một solution tốt trong một vài trường hợp nhất định.

1. Trigger và Procedure

* Trigger

create trigger [dbo].[insert\_update\_cart]

ON [dbo].[Cart]

for INSERT, UPDATE

AS

BEGIN

declare @idProduct int

declare @idOrder int

declare @amount int

declare @idCart int

select @idProduct = ProductList.idProduct from inserted ProductList

print @idProduct

select @idOrder = ProductList.idOrder from inserted ProductList

select @idCart = ProductList.idCart from inserted ProductList

select @amount = ProductList.amount from inserted ProductList

declare @priceProduct int

select @priceProduct = price from [dbo].[Product] where [dbo].[Product].idProduct = @idProduct

declare @idSup int

select @idSup = idSup from [dbo].Provide where idProduct = @idProduct

if @priceProduct = 0

BEGIN

print 'Product id dont existed in the database'

ROLLBACK TRAN

END

if @amount = 0

BEGIN

print 'System dont accept amount = 0'

ROLLBACK TRAN

END

update [dbo].[Cart]

set amount = @amount

where [dbo].[Cart].[idCart] = @idCart and idProduct = @idProduct;

if exists (select idOrder from [dbo].Order\_Table where idOrder = @idOrder)

BEGIN

update Order\_Table set total\_money =

(

select Sum(CAST(c.amount\*p.price AS BIGINT)) from Cart c

inner join Product p on c.idProduct = p.idProduct

where Order\_Table.idOrder = c.idOrder

)

update [dbo].Order\_Table

set date = GETDATE()

where idOrder = @idOrder

END

END

go

create trigger [dbo].[delete\_order]

ON [dbo].[Order\_Table]

for DELETE

AS

BEGIN

declare @idOrder int

select @idOrder = OrderList.idOrder from deleted OrderList

if @idOrder = 0

BEGIN

print 'Order id cannot = 0'

ROLLBACK TRAN

END

delete from [dbo].[Cart] where idOrder = @idOrder;

END

go

create trigger [dbo].[insert\_update\_order]

ON [dbo].[Order\_Table]

FOR INSERT, UPDATE

AS

BEGIN

declare @date date

select @date = OrderList.date from inserted OrderList

declare @idOrder int

select @idOrder = OrderList.idOrder from inserted OrderList

declare @total\_money int

select @total\_money = total\_money from Order\_Table where idOrder = @idOrder

declare @idCus int

select @idCus = OrderList.idCus from inserted OrderList

declare @shippingFee int

select @shippingFee = shipping\_fee from Shipping\_Fee where idOrder = @idOrder

declare @tran\_method nvarchar(10)

select @tran\_method = OrderList.tran\_method from inserted OrderList

print @total\_money

execute insertUpdateCustomer\_log @idCus , @idOrder ,1 , @total\_money , @date

-- ShippingC\_log

IF @total\_money != 0

BEGIN

print @total\_money

execute insertUpdateShippingC\_log @idOrder , @date , 1 , @shippingFee

execute insertUpdateRevenue\_log @total\_money , @shippingFee , @idOrder , @idCus , @date , @tran\_method

END

END

go

create trigger [dbo].[delete\_product]

on [dbo].[Product]

for delete

as

begin

declare @idProduct int

select @idProduct = ProductList.idProduct from deleted ProductList

delete from Cart where idProduct = @idProduct

declare @idCart1 int

select @idCart1 = idCart from Cart where idProduct = @idProduct

declare @temp int

declare @idOrder int

select @idOrder = idOrder from Cart where idCart = @idCart1

select @temp = total\_money from Order\_Table where idOrder = @idOrder

declare @productPrice int

select @productPrice = price from Product

declare @temp2 int

set @temp2 = @temp - @productPrice

update Order\_Table

set total\_money = @temp2

where idOrder = @idOrder

delete from Product\_price\_log where idProduct = @idProduct

delete from Product\_review where idProduct = @idProduct

end

go

create trigger [dbo].[update\_product]

on [dbo].[Product]

for UPDATE

AS

BEGIN

declare @idProduct int

declare @newPrice int

select @idProduct = ProductList.idProduct from inserted ProductList

if @idProduct = 0

BEGIN

print 'Product id cannot = 0'

ROLLBACK TRAN

END

declare @oldPrice int

execute @oldPrice = getOldPrice @idProduct

print @oldPrice

select @newPrice = ProductList.price from inserted ProductList

EXEC procedure\_on\_updateProduct\_price\_log @newPrice1 = @newPrice, @oldPrice1 = @oldPrice, @idProduct1 = @idProduct

update Cart

set idProduct = @idProduct

where idProduct = @idProduct

END

* Procedure

create PROCEDURE [dbo].[getLatestIdLog]

AS

BEGIN

IF not exists (select max(idLog)+1 from Product\_price\_log)

BEGIN

return 1

END

ELSE

BEGIN

return (select max(idLog)+1 from Product\_price\_log)

END

END

go

create PROCEDURE [dbo].[getLatestIdLogCustomer\_log]

AS

BEGIN

return (select max(idLog)+1 from Customer\_log)

END

go

create PROCEDURE [dbo].[getLatestIdLogRevenue\_log]

AS

BEGIN

return (select max(idLog)+1 from revenue\_log)

END

go

create PROCEDURE [dbo].[getLatestIdLogShippingC\_log]

AS

BEGIN

return (select max(idLog)+1 from ShippingC\_log)

END

go

create PROCEDURE [dbo].[getLatestIdLogSupplier\_log]

AS

BEGIN

IF not exists (select max(idLog)+1 from supplier\_log)

BEGIN

return 1

END

ELSE

BEGIN

return (select max(idLog)+1 from supplier\_log)

END

END

go

create PROCEDURE [dbo].[getOldPrice] @idProduct1 int

AS

BEGIN

if not exists (select new\_price from Product\_price\_log where idProduct = @idProduct1)

begin

return 0

end

else

begin

return (select new\_price from Product\_price\_log where idProduct = @idProduct1)

end

END

go

create PROCEDURE [dbo].[insertUpdateCustomer\_log] @idCus int, @idOrder int,@status int, @date date

AS

BEGIN

IF not exists (select idLog from Customer\_log where idCus = @idCus)

BEGIN

declare @idLogCus int

execute @idLogCus = getLatestIdLogCustomer\_log

IF @idLogCus = 0

BEGIN

set @idLogCus = 1

END

insert into Customer\_log(idCus, idLog, idOrder, status, date)

values (@idCus, @idLogCus, @idOrder,@status, @date)

END

ELSE

BEGIN

declare @idLogCus2 int

select @idLogCus2 = idLog from Customer\_log where idOrder = @idOrder and idCus = @idCus

update Customer\_log

set date = @date, status = @status

where idLog = @idLogCus2

END

END

go

create PROCEDURE [dbo].[insertUpdateRevenue\_log] @total\_money int, @shippingFee int, @idOrder int, @date date, @status int

AS

BEGIN

print @total\_money

declare @temp int

set @temp = 0.02\*@shippingFee +0.02\*(@total\_money - @shippingFee)

IF not exists (select idLog from revenue\_log where idOrder = @idOrder)

BEGIN

declare @idLog int

execute @idLog = getLatestIdLogRevenue\_log

IF @idLog = 0

BEGIN

set @idLog = 1

END

insert into revenue\_log(idLog, idOrder, date, status ,total\_money)

values (@idLog, @idOrder, @date, @status ,@temp )

END

ELSE

BEGIN

declare @idLog2 int

select @idLog2 = idLog from revenue\_log where idOrder = @idOrder

update revenue\_log

set date = @date, total\_money = @temp, status = @status

where idLog = @idLog2

END

END

go

create PROCEDURE [dbo].[insertUpdateShippingC\_log] @idOrder int, @date date, @status int, @shippingFee int

AS

BEGIN

IF not exists (select idLog from ShippingC\_log where idOrder = @idOrder)

BEGIN

declare @idLogShippingC\_log int

execute @idLogShippingC\_log = getLatestIdLogShippingC\_log

IF @idLogShippingC\_log = 0

BEGIN

set @idLogShippingC\_log = 1

END

declare @idSCompany int

select @idSCompany = idSCompany from Shipping\_Fee where idOrder = @idOrder

insert into ShippingC\_log(idLog, idOrder, idSCompany, status, date, total\_money)

values (@idLogShippingC\_log, @idOrder,@idSCompany ,@status, @date, 0.98\*@shippingFee)

END

ELSE

BEGIN

declare @idLogShippingC\_log2 int

select @idLogShippingC\_log2 = idLog from ShippingC\_log where idOrder = @idOrder

update ShippingC\_log

set date = @date, status = @status, total\_money =0.98\*@shippingFee

where idLog = @idLogShippingC\_log2

END

END

go

create PROCEDURE [dbo].[procedure\_on\_updateProduct\_price\_log] @newPrice1 int, @oldPrice1 int, @idProduct1 int

AS

BEGIN

IF not exists (SELECT idLog FROM Product\_price\_log WHERE idProduct = @idProduct1)

BEGIN

declare @tempIdLog int

execute @tempIdLog = getLatestIdLog;

IF @tempIdLog = 0

BEGIN

set @tempIdLog = 1

END

insert into Product\_price\_log(idProduct, idLog, new\_price, date\_update, old\_price)

values(@idProduct1,@tempIdLog ,@newPrice1, GETDATE(), @oldPrice1)

END

else

begin

update Product\_price\_log

set new\_price = @newPrice1, old\_price = @oldPrice1, date\_update = GETDATE()

where idProduct = @idProduct1

END

END

1. Xác định yêu cầu chức năng:

Chức năng:

1. Trang chủ:

* Xem danh sách sản phẩm trên trang chủ.
* Hiển thị sản phẩm theo danh sách, tìm kiếm sản phẩm theo tên.
* Đăng nhập + đăng ký.
* Trang thông tin cá nhân:
* Xem thông tin cá nhân + địa chỉ giao hàng + đơn hàng vừa đặt, thay đổi thông tin cá nhân.
* Thay đổi địa chỉ giao hàng.
* Kiểm tra order theo mã order và email
* Xem danh sách tất cả order đã và đang đặt.

1. Admin:

* Xem doanh thu tháng này/ doanh thu tổng của ShopePhake.
* Xem danh sách user có trong hệ thống (có thể thay đổi chức năng của người dùng ), tìm kiếm người dùng theo tên
* Xem các nhà cung cấp hiện có trong hệ thống, tìm kiếm nhà cung cấp theo tên.
* Xem danh sách các đối tác giao dịch có trong hệ thống, tìm kiếm đối tác theo tên.
* Xem danh sách các đơn vị vận chuyển, tìm kiếm đơn vị vận chuyển theo tên.

1. Nhà Cung cấp:

* Xem thông tin của nhà cung cấp, thay đổi thông tin.
* Thêm sản phẩm vào hệ thống.
* Cập nhật giá.
* Xem danh sách đơn hàng của nhà cung cấp và tùy chỉnh trạng thái đơn.
* Xem doanh thu theo ngày/ tháng/ năm, xem doanh thu từ ngày này sang ngày khác.

1. Giỏ hàng

* Ước tính giá ship, thay đổi số lượng hoặc xóa cả giỏ hàng hoặc xóa 1 mặt hàng.
  + Chức năng thanh toán đơn.
* Chọn đơn vị vận chuyển, thời gian, hình thức thanh toán, xong tiến hành thanh toán đơn hàng.

|  |  |  |
| --- | --- | --- |
|  | Đối tượng sử dụng | Mô tả |
| Chức năng tạo đơn hàng mới, lấy dữ liệu từ đơn vị vận chuyển. Chỉnh sửa đơn hàng nếu chưa thực hiện giao hàng. | Khách hàng | Khách hàng chọn sản phẩm cần mua, tạo mới đơn, chọn đơn vị vận chuyển phù hợp, xác nhận đơn hàng và tiền hành thanh toán( tiền mặt/ tiền điện tử). Có thể thay đổi số lượng hoặc thêm sản phẩm khác tồn tại trong shop. |
| Chức năng đặt hàng chung | Khách hàng | Khi tiến hành đặt hàng, nếu giỏ hàng chưa đủ chỉ tiêu để nhận khuyến mãi (VD: Đơn hàng > 500.000VNĐ để nhận mã giảm 50%), thì khách hàng có thể dùng code của đơn hàng (hệ thống sẽ tự tạo) để đưa cho bạn bè mình đặt thêm. Khi vận chuyển thì đơn sẽ được tách ra và giao về địa chỉ của khách hàng và bạn bè của khách. |
| Chức năng trang trí giao diện của trang bán | Chủ shop | Chủ shop có thể tùy chỉnh giao diện shop của mình cho bắt mắt, thu hút khách hàng. Các giao diện này có thể được trang trí theo template của hệ thống hoặc chủ shop kéo thả các mục tùy sao cho hợp lý (VD: mục tên shop, đánh giá nên để đầu trang, màu sắc sao cho đẹp, sản phẩm nên để giữa trang cho khách hàng dễ dàng quan sát). |
| Chức năng theo dõi doanh thu bán hàng, số lượng và tình trang đơn hàng đang giao | Chủ shop | Chủ shop có thể xem thống kê doanh thu của tháng này/ năm/ tổng doanh thu. Sản phẩm bán chạy nhất, sản phẩm bán ít nhất để dễ dàng tùy chỉnh số lượng bán trong tương lai. Xem được tình trạng của đơn hàng đang giao (giao chậm, đang giao, đã giao, đã hủy). |
| Chức năng theo dõi các thành viên trong hệ thống | Admin( nhà quản lý) | Admin có thể xem được các tài khoản đang có trong hệ thống, có thể là khách hàng, chủ shop và các nhà quản lý khác. |
| Chức năng theo dõi doanh thu/ số lượng tồn kho/ đánh giá của từng shop | Admin (nhà quản lý) | Admin có thể xem được tình trạng hoạt động của shop, số lượng đánh giá tiêu cực, tích cực của shop. Nếu nhận quá nhiều đánh giá tiêu cực sẽ bị cảnh cáo, và có thể cho đóng shop nếu còn tiếp diễn. |
| Chức năng theo dõi đơn hàng | Admin (nhà quản lý) | Admin có thể quản lý các đơn hàng đang giao của từng shop để theo dõi liệu đơn vị vận chuyển A có giao đúng thời hạn không, chủ shop có đưa hàng đúng thời hạn cho đơn vị vận chuyển không. Thống kê số lần giao hàng trễ của đơn vị vận chuyển/ shop để cảnh cáo. |
| Chức năng quản lý đơn vị vận chuyển | Admin ( nhà quản lý) | Admin có thể quản lý các đơn vị vận chuyển đã có hợp đồng với công ty. Các thông tin quản lý như: Tên đơn vị, giá trên 1km, điều kiện nhận khuyến mãi theo từng đơn vị vận chuyển, thời gian trung bình giao 1 đơn tùy theo vị trí địa lý. |

* Chức năng đăng nhập: phân quyền người dùng( khách hàng, chủ shop, admin).
* Quy trình: khách hàng đăng nhập có thể xem danh sách các shop đang hoạt động (sản phẩm bán chạy nhất/đang khuyến mãi ở trang chủ). “Khách hàng” khi đăng ký bán sản phẩm thì trở thành “Chủ shop”. Tình huống: khách hàng => đăng ký bán sản phẩm => hệ thống xác nhận => chủ shop.
* Chủ shop đăng ký tạo 1 gian hàng online => thêm/xóa/sửa/cập nhật dữ liệu của sản phẩm trên gian hàng.
* Admin khi đăng nhập vào trang ví dụ : /wp-admin thì sẽ chuyển hướng vào trang quản lý người dùng, quản lý các chủ shop (doanh thu, doanh số, sao đánh giá, …).

KHÁCH HÀNG

* Chức năng tạo đơn, lấy dữ liệu từ các đơn vị vận chuyển ( những thông tin cần lấy: tên đơn vị, giá ship hàng theo km tính từ vị trí của khách hàng đến kho của công ty, khuyến mãi nếu có), xác nhận đơn, hoàn tất đơn. Chức năng đánh giá sản phẩm sau khi hoàn tất đơn. Chức năng chỉnh sửa đơn hàng nếu đơn hàng chưa được vận chuyển. Chức năng gửi email quảng cáo/khuyến mãi (chỉ với những người dùng đăng ký).
* **Chức năng có thể có:** đặt hàng chung với bạn bè thông qua mã liên kết giỏ hàng. Tình huống: người A đặt xong sản phẩm tuy nhiên vẫn chưa đủ chỉ tiêu để nhận khuyến mãi, lúc này người A có thể lấy mã liên kết giỏ hàng cho bạn mình đặt, sau khi hoàn tất đặt hàng thì sẽ có 2 đơn vị vận chuyển (giá ưu đãi) thực hiện chuyển phát.

CHỦ SHOP:

* + Chủ shop ngoài thêm xóa sửa cập nhật còn có: thiết kế giao diện của gian hàng**(TƯƠNG TỰ CHỨC NĂNG TRANG TRÍ SHOP CỦA SHOPEE).**

VÍ DỤ: https://banhang.shopee.vn/edu/article/851 .

* + Chủ shop có thể thống kê doanh thu/ số lượng đơn hàng.

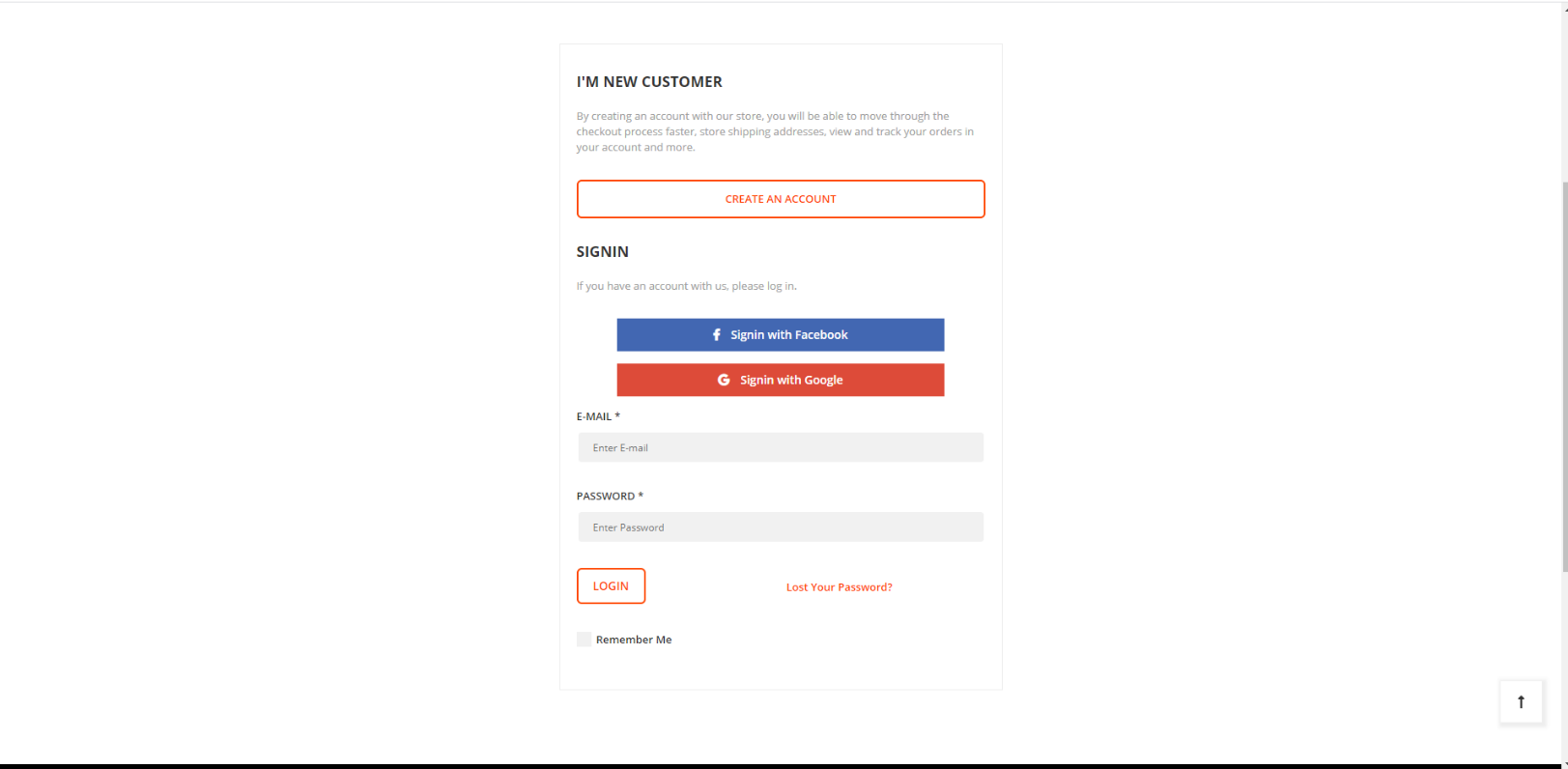


ADMIN ( NHÀ QUẢN LÝ )

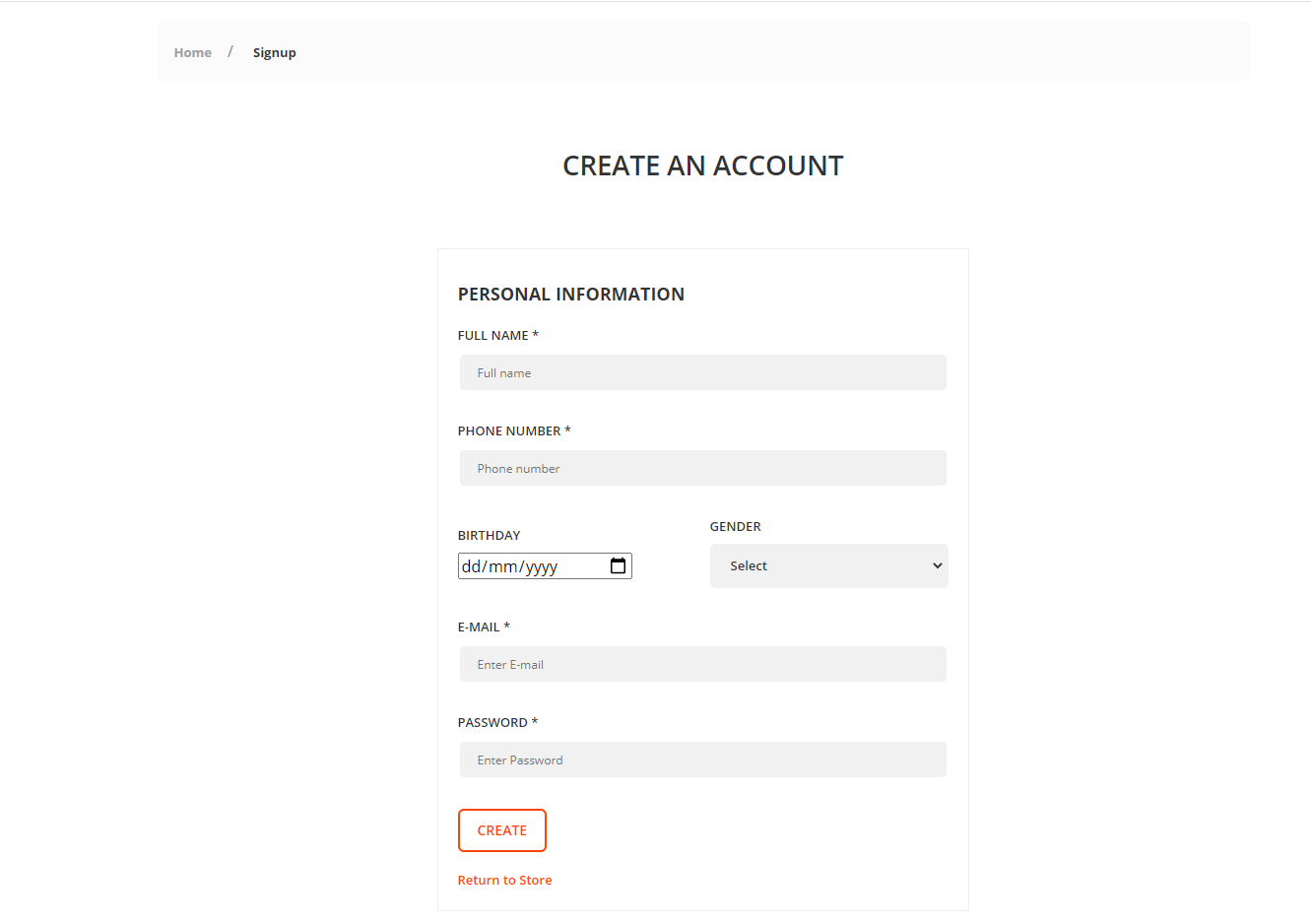
* Theo dõi số thành viên trong hệ thống - có thể là các quản lý, các khách hàng, các chủ shop- (tên tài khoản, email, vai trò, bài đăng/đánh giá, lần cuối đăng nhập)
* Chức năng theo dõi doanh thu/số lượng tồn của các shop, số lượng đánh giá tích cực/ tiêu cực. Nếu quá nhiều đánh giá tiêu cực/ tố cáo thì shop sẽ bị ngưng hoạt động.
* Theo dõi tình trạng đơn hàng đơn hàng đã giao, chưa giao, còn chờ giao, giao trễ của shop.
* Quản lý đơn vị vận chuyển (tên đơn vị, số tiền trên 1 km, điều kiện nhận khuyến mãi, thời gian trung bình giao 1 đơn tùy theo vị trí địa lý)

1. **Layout của giao diện:**

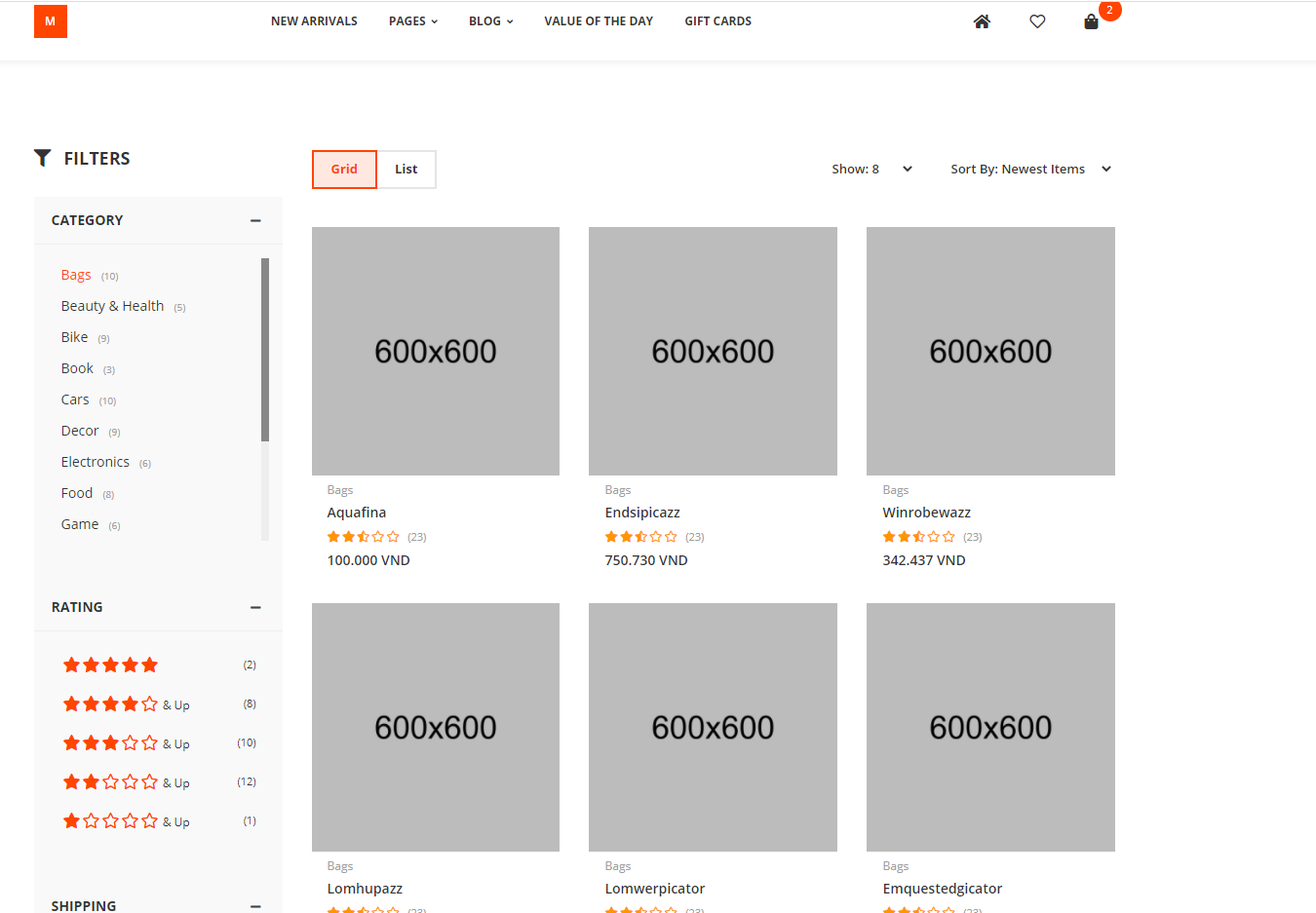
**Đăng nhập:**



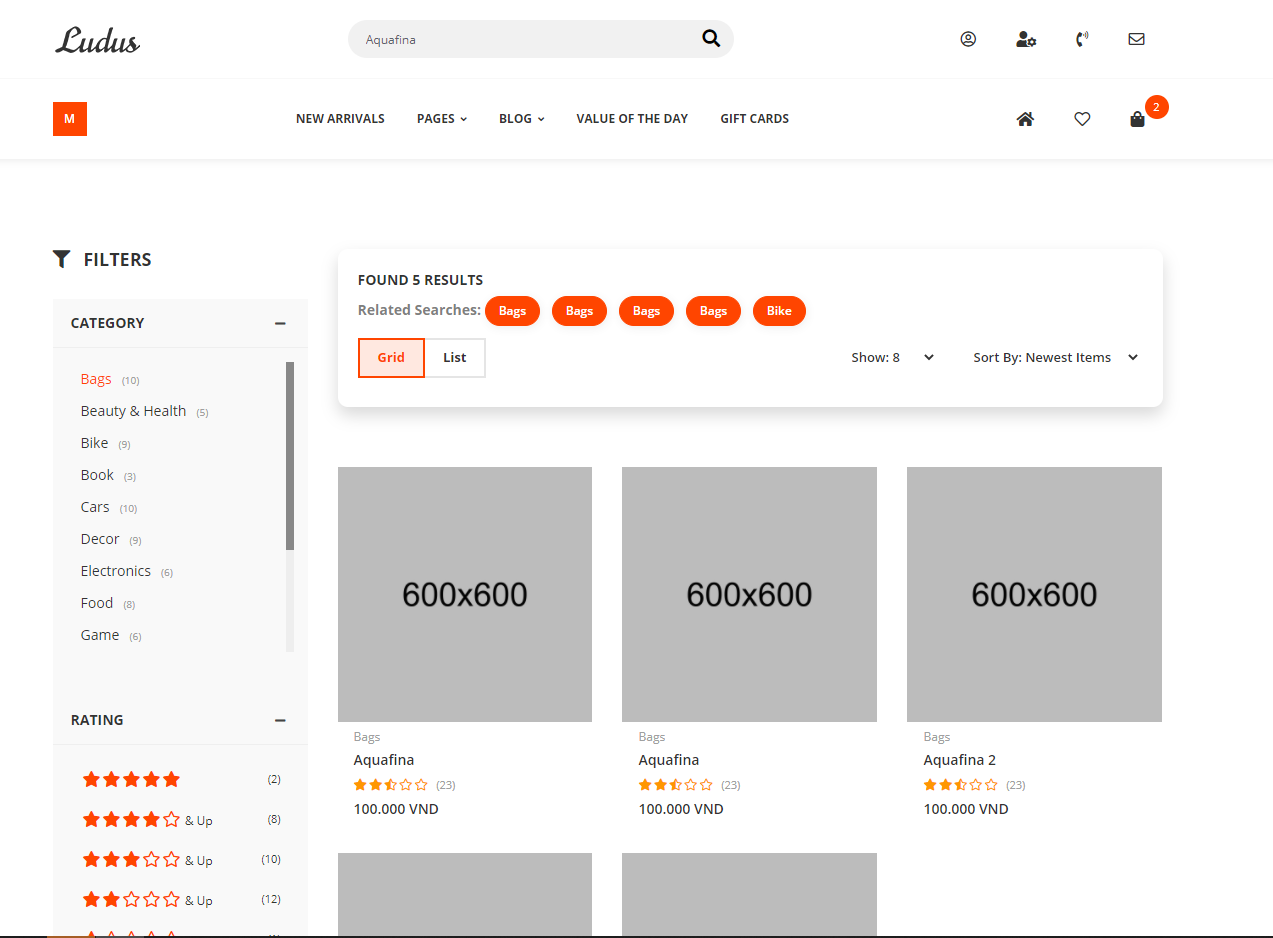
**Đăng ký:**



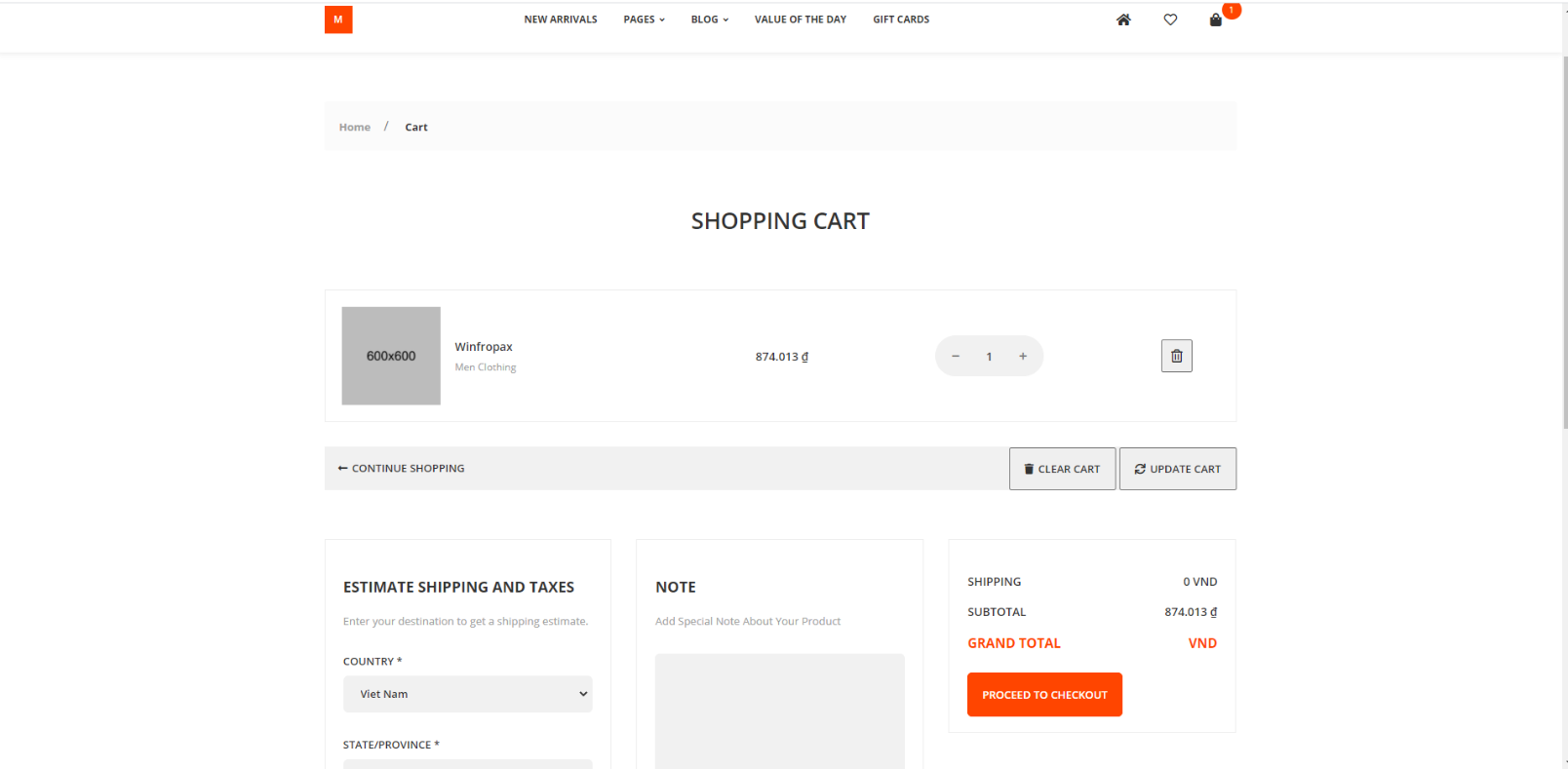
**Xem sản phẩm theo loại**



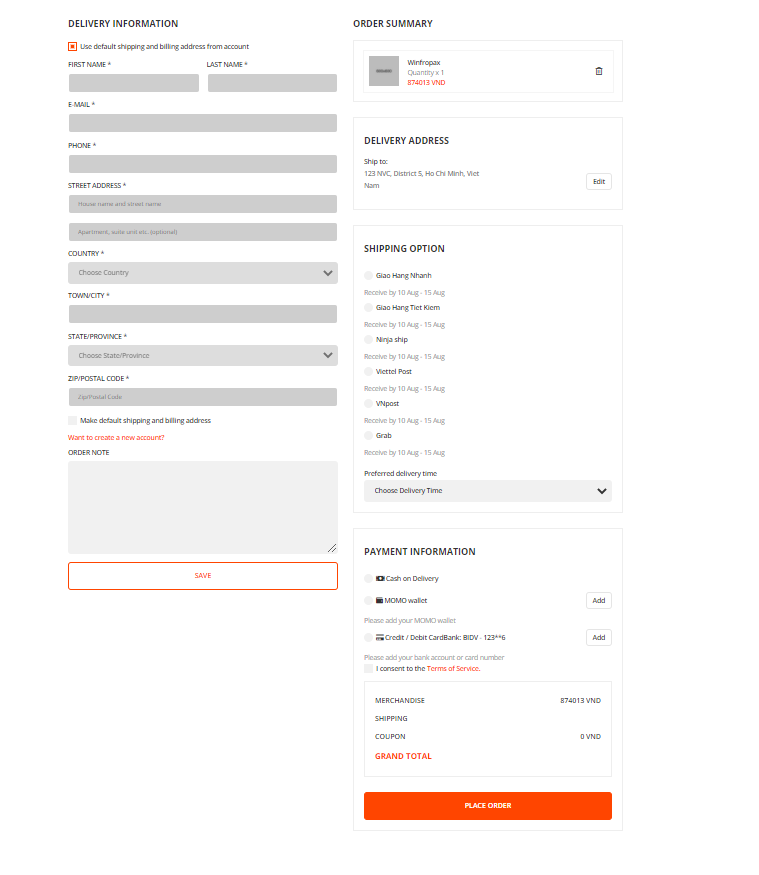
**Tìm kiếm sản phẩm(Aquafina):**



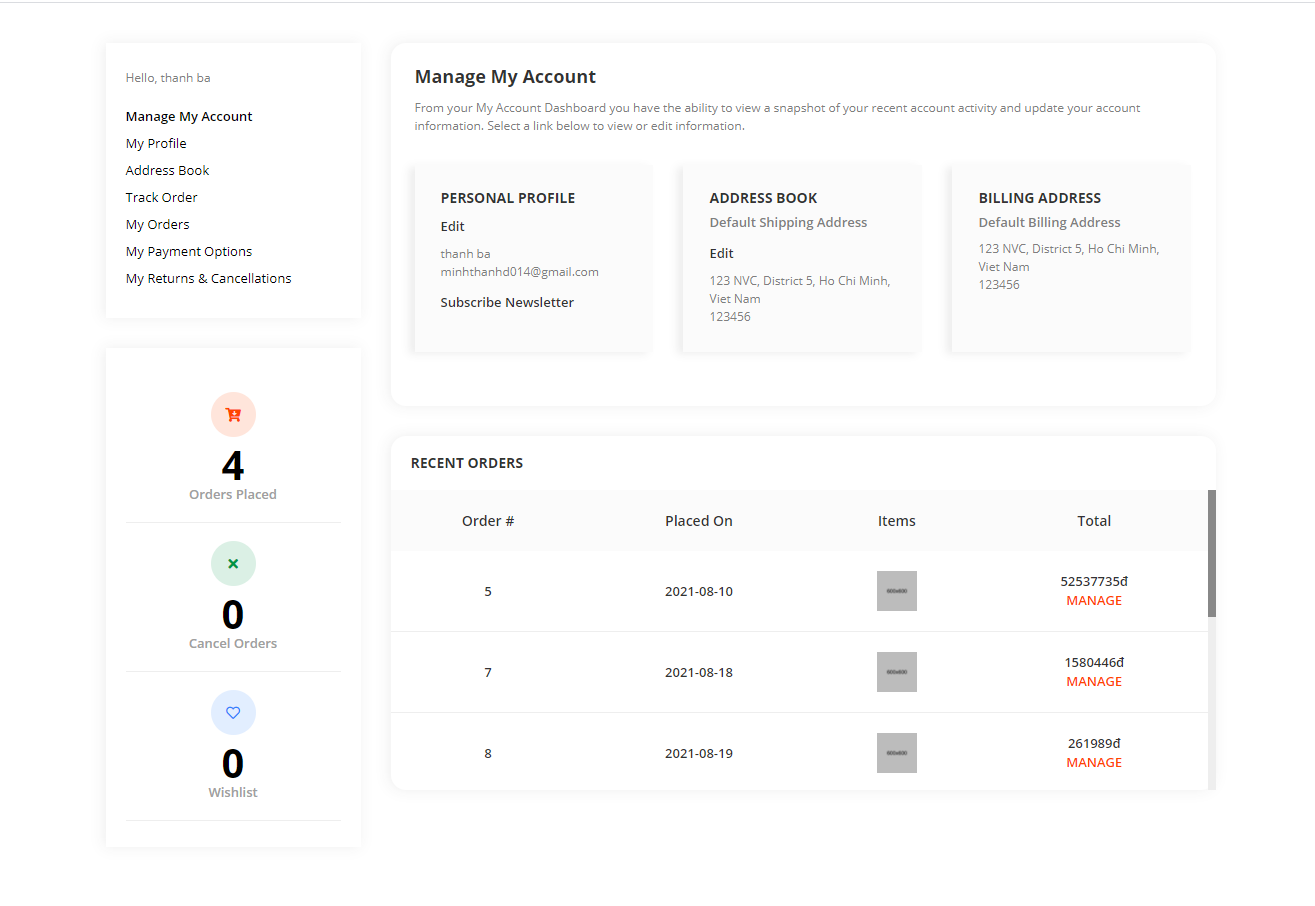
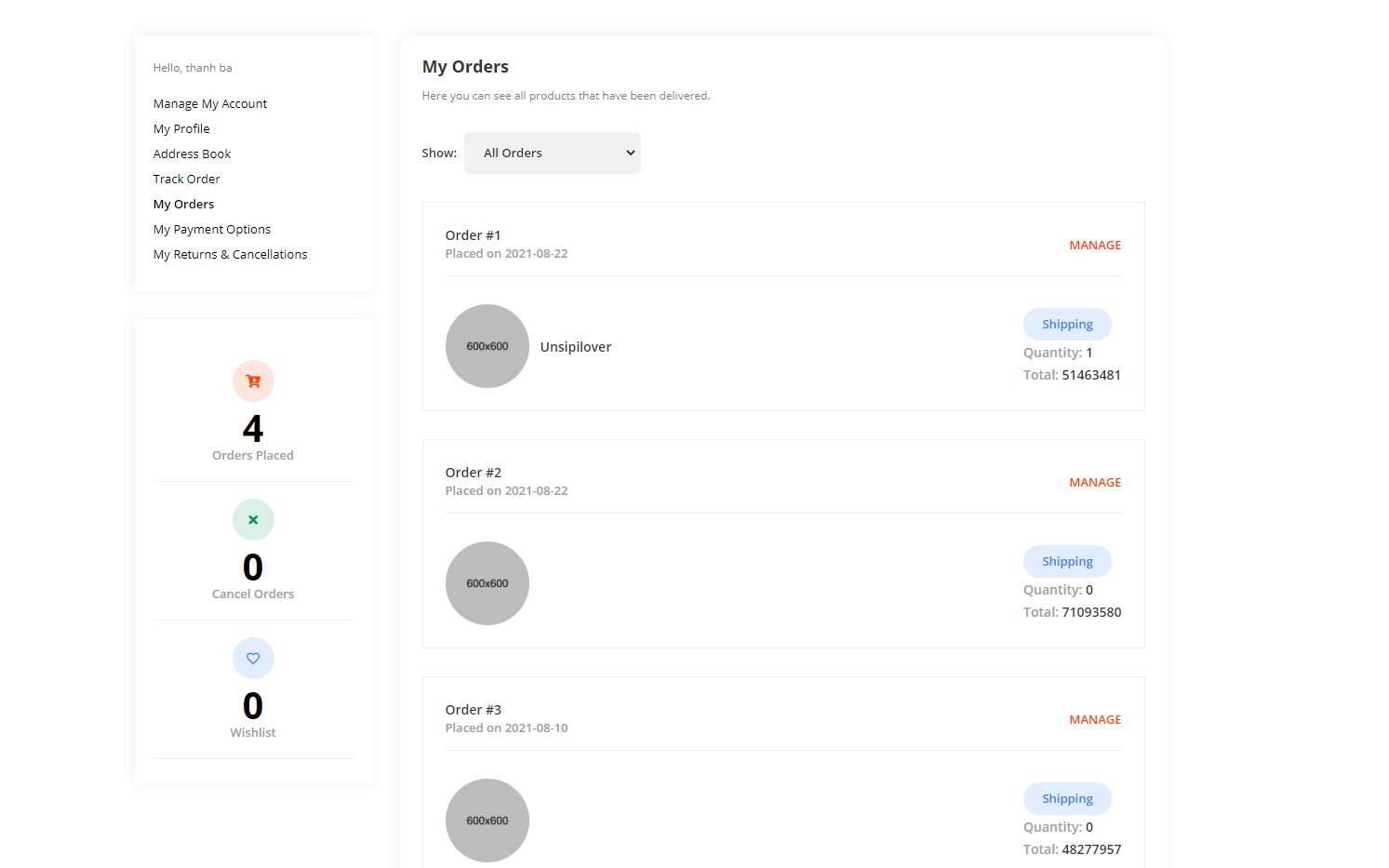
**Giỏ hàng:**



**Thanh toán:**

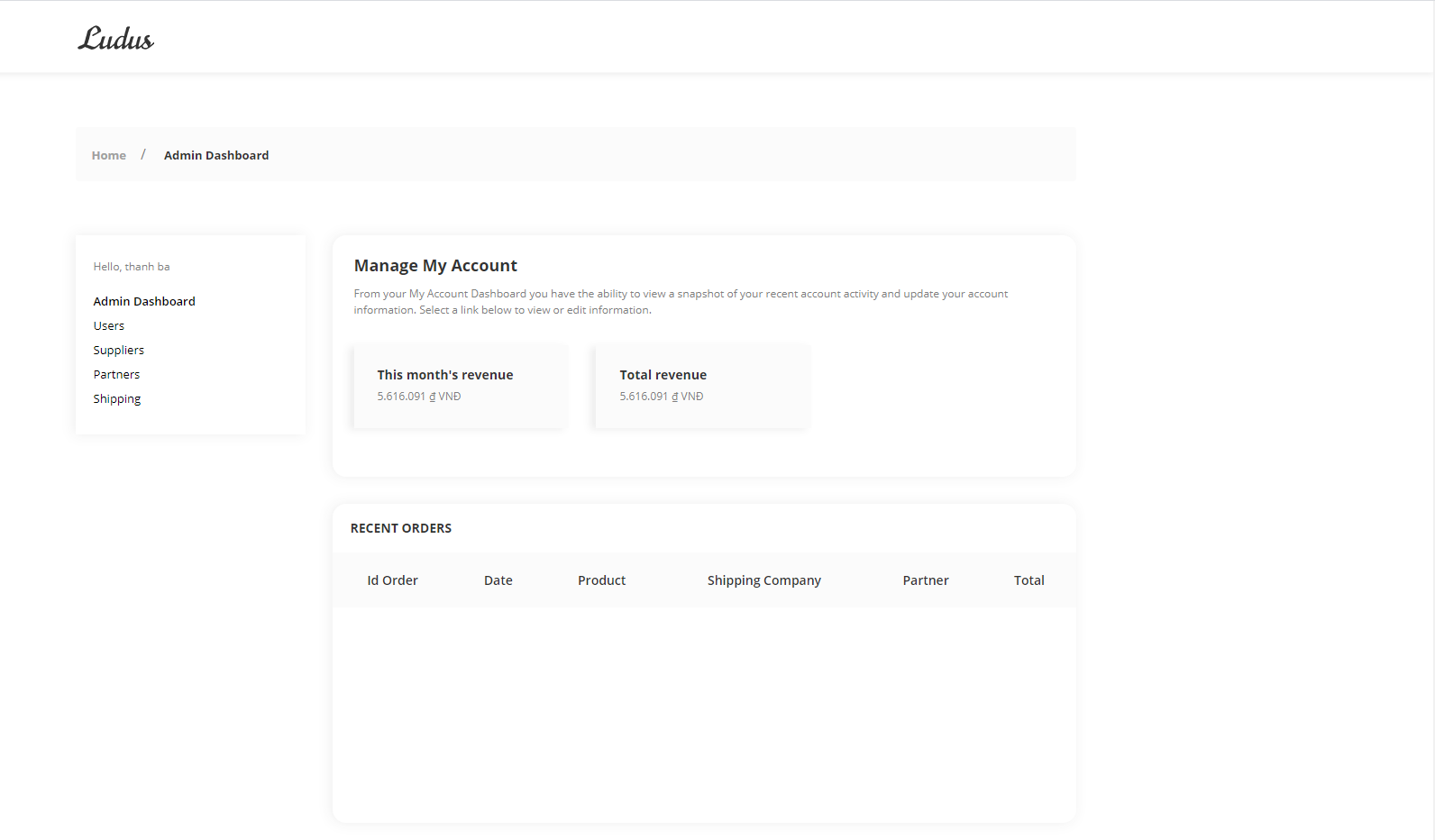


**Trang cá nhân:**

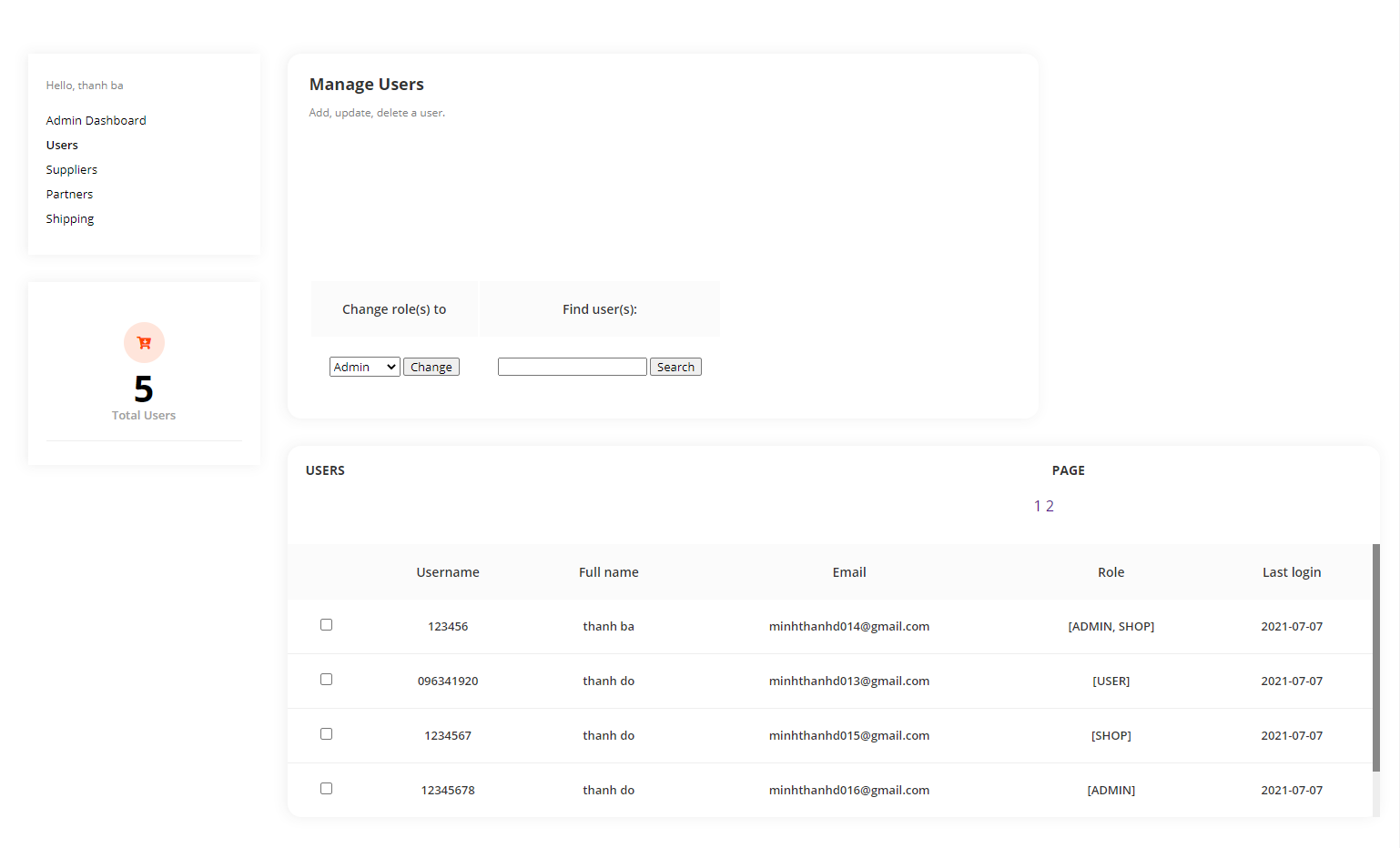
* Thông tin cá nhân:
* 
* Hiển thị danh sách order:
* 

**Admin:**

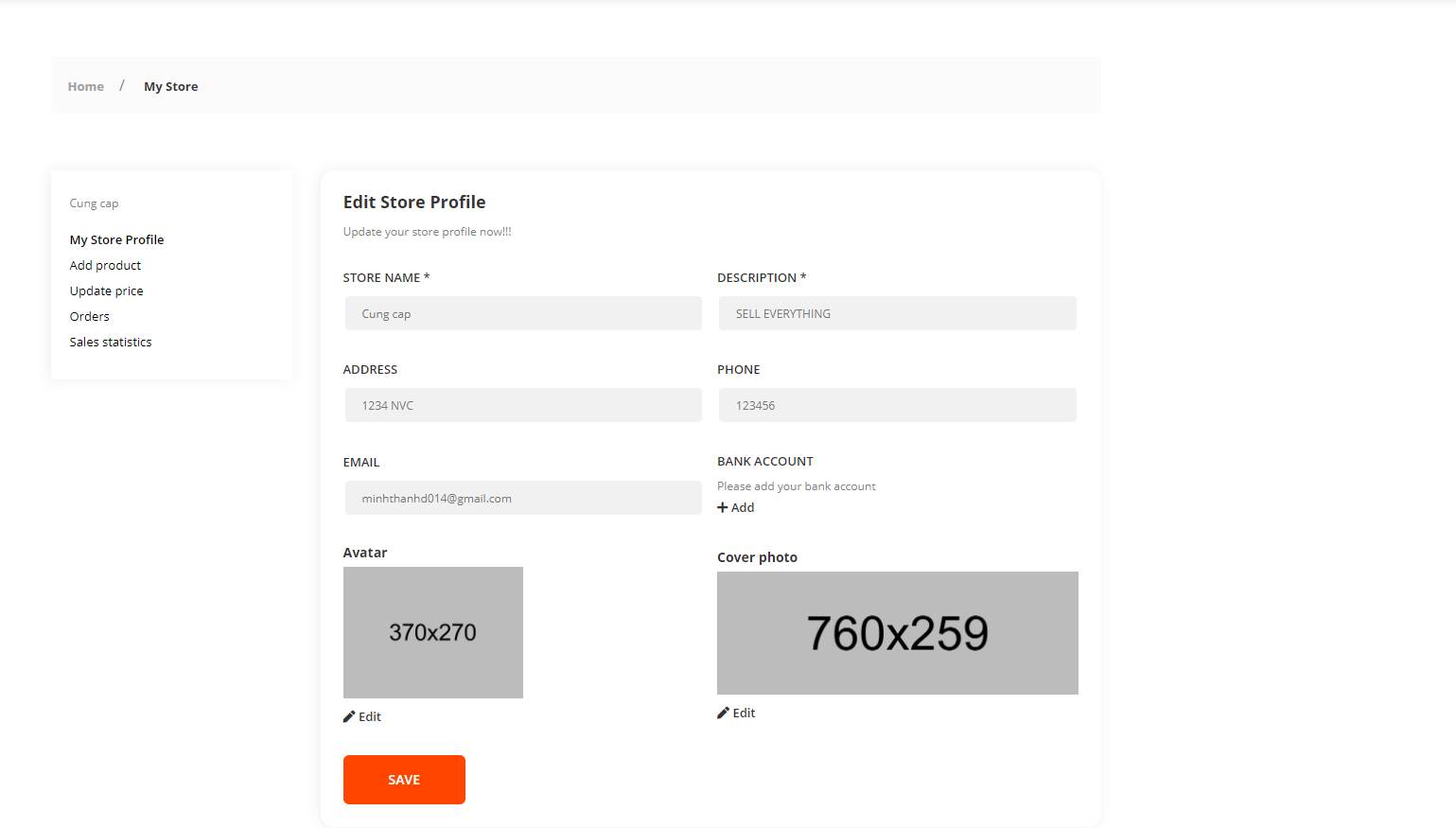
Admin dashboard:



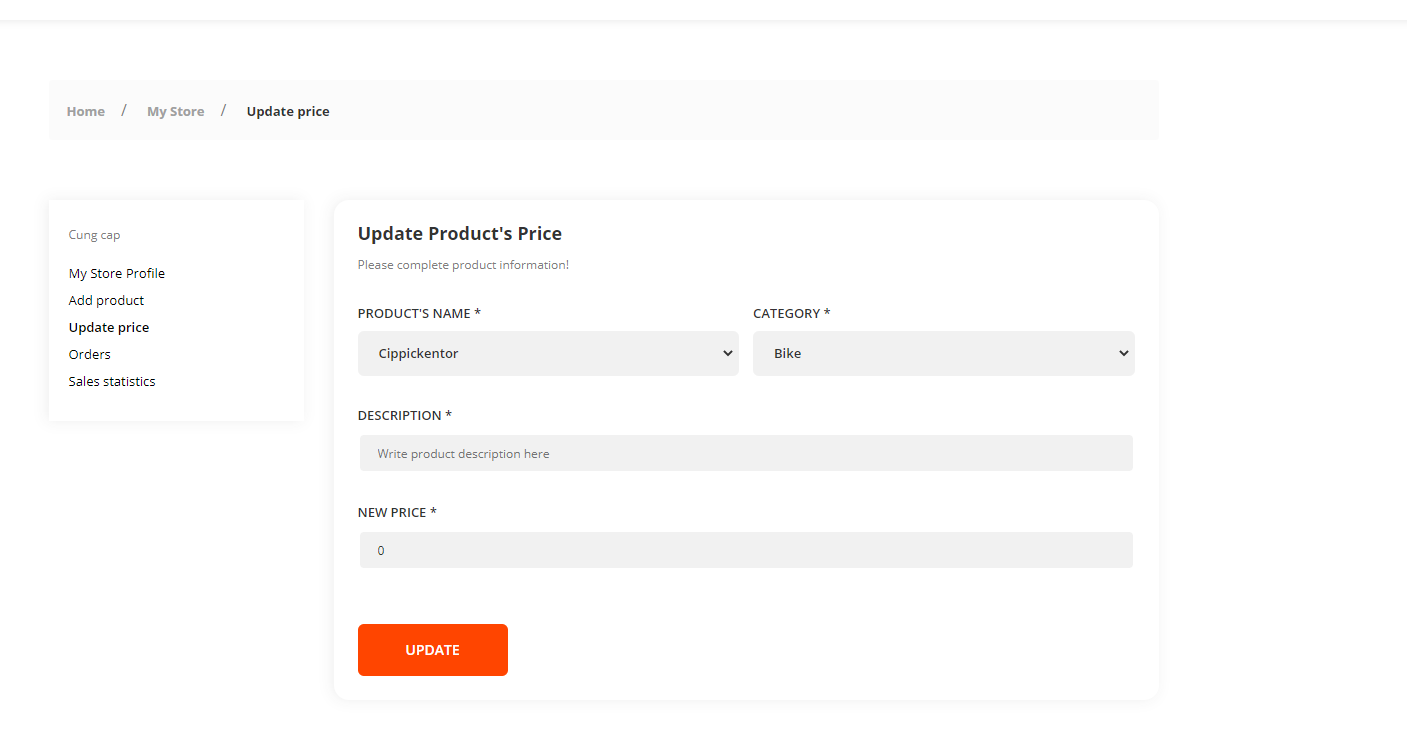
Danh sách user:



**Chủ shop:**



* Thay đổi giá:



1. **Partiton**

**Phân chia dữ liệu – Partition**

1. **Partition:**

* *Partition (Phân chia dữ liệu):* là một giải pháp chia nhỏ một Database lớn thành nhiều Database nhỏ, ta có thể phân tách từng bảng hoặc cả một DB ra nhiều phần nhỏ đặt ở nhiều máy chủ (server) khác nhau. Điều này sẽ giúp cho hệ thống DB của chúng ta đạt được các tính chất khả năng bảo trì, hiệu xuất, tính sẵn sàng, và cân bằng tải của ứng dụng. Và giải pháp này cũng giảm chi phí cũng như tính mở rộng để scale up DB bằng cách dùng nhiều server nhỏ gộp lại hơn là nâng cấp một server lớn.
* Partition function: Ánh xạ các dòng của bảng hoặc các chỉ mục trong phân mảnh dựa theo tiêu chí phân mảnh. Qui định giá trị biên cho các đoạn. Hệ thống dựa vào hàm này để xác định đoạn mà mỗi bản ghi thuộc vào.
* Partition Scheme: Ánh xạ các phân mảnh của bảng vào các filegroup (mỗi filegroup chứa một phân mảnh)

1. **Lợi ích của việc sử dụng Partition:**

* Cải thiện khả năng co giãn và khả năng quản lý của các bảng lớn (large table) và các bảng có các kiểu truy cập khác nhau.
* Khi các bảng và các chỉ mục quá lớn, việc phân mảnh giúp chia dữ liệu thành cá phần nhỏ hơn, có thể quản lý được.
* Nếu 1 bảng lớn tồn tại trên hệ thống nhiều CPUs, việc phân mảnh bảng sẽ giúp tăng hiệu suất khi thực hiện song song
* *Việc backup/restore* một đoạn mà không ảnh hưởng đến các đoạn còn lại.

🡺 Phân chia dữ liệu sẽ giúp cho hệ thống DB đạt được các tính chất khả năng bảo trì, hiệu xuất, tính sẵn sàng, và cân bằng tải của ứng dụng. Và giải pháp này cũng giảm chi phí cũng như tính mở rộng để scale up DB bằng cách dùng nhiều server nhỏ gộp lại hơn là nâng cấp một server lớn.

1. **Các kiểu phân chia dữ liệu:**

* Phân chia theo chiều ngang (horizontal partition):
* Phân chia các dòng trong một table thành nhiều table khác nhau.
* Tình huống áp dụng: khi nhiều người dùng khác nhau cần truy cập các dòng dữ liệu khác nhau.
* Ưu điểm: tối ưu hoá tốc độ truy cập dữ liệu
* Nhược điểm:

Phức tạp khi phải truy cập toàn bộ dữ liệu.

Nếu không phân chia cẩn thận sẽ dễ gây mất cân bằng giữa các DB.

* Phân chia theo chiều dọc (vertical partition):
* Phân chia một cấu trúc luận lý thành những cấu trúc lưu trữ vật lý khác nhau. Phân chia dựa vào thuộc tính (cột) của bảng thành nhiều bảng khác nhau.
* Tình huống áp dụng: chia bảng thành các loại cấu trúc có giá trị sử dụng khác nhau hoặc cấu trúc truy cập thường xuyên và không thường xuyên.
* Ưu điểm: Dễ thực hiện và không ảnh hưởng đến ứng dụng
* Nhược điểm: Khi hệ thống lớn thì dữ liệu cũng lớn theo, do đó phải thực hiện phân chia nhiều lần.

1. **Phân chia dữ liệu bảng Order (Hoá đơn) theo chiều ngang**

**Bước 1: Tạo các file group**

* Tạo các filegroup để chứa các phân mảnh sau khi đã phân chia.
* Group 1 (Primary): lưu các đơn hàng trước năm 2020 (ngày 31/12/2019 trở về trước)
* Group 2 (FG2): lưu các đơn hàng trong năm 2020 (từ ngày 1/1/2020 đến 31/12/2020)
* Group 3 (FG3): lưu các đơn hàng sau năm 2020 (từ ngày 1/1/2021 trở về sau)

**CREATE DATABASE** **DB\_Order\_Partition**

ON PRIMARY

(NAME= 'Order\_Partition\_1', FILENAME='D:\PartitionDB\FG1\Order\_Partition\_1.mdf', SIZE=2, MAXSIZE=100, FILEGROWTH=1 ),

FILEGROUP FG2

(NAME = 'Order\_Partition \_2', FILENAME = 'D:\PartitionDB\FG2\Order\_Partition\_2.ndf', SIZE = 2, MAXSIZE=100, FILEGROWTH=1 ),

FILEGROUP FG3

(NAME = 'Order\_Partition\_3', FILENAME='D:\PartitionDB\FG3\Order\_Partition\_3.ndf', SIZE = 2, MAXSIZE=100, FILEGROWTH=1 )

**GO**

**Bước 2: Tạo Partition Function**

* Tạo các giá trị biên cho mỗi bảng dựa vào cột Date trong bảng chính (Order)
* Hàm Partition function ‘OrderDatePartition’ tạo được 3 bảng nhỏ chia dữ liệu theo các mốc thời gian: trước ngày 1/1/2020; từ 1/1/2020 đến 31/12/2020; từ 1/1/2021 trở về sau.
* RANGE RIGHT nghĩa là lấy < or >=

LEFT lấy <= and >

|  |  |
| --- | --- |
| <1/1/2020 | Order\_Partition\_1 |
| >=1/1/2020 and <31/12/2020 | Order\_Partition\_2 |
| >=1/1/2021 | Order\_Partition\_3 |

**Use** DB\_Order\_Partition

**GO**

**CREATE PARTITION FUNCTION** OrderDatePartition (datetime)

AS

RANGE RIGHT

FOR VALUES ( '2020-01-01', '2021-01-01')

**GO**

**Bước 3: Tạo partition schema**

* Hàm Partition Scheme ‘OrderDatePartitionScheme’ dùng hàm ‘OrderDatePartition’ để “gắn” các đoạn vào từng filegroup

**Use** DB\_Order\_Partition

**GO**

**CREATE PARTITION SCHEME** OrderDatePartitionScheme

AS

PARTITION OrderDatePartition

TO ([PRIMARY], FG2, FG3 )

**GO**

**Bước 4: Tạo partitioned table**

* Tạo một bảng mới từ Partition Scheme, bảng này sẽ lọc ra những dữ liệu cần tìm kiếm nhiều khi truy vấn gồm mã đơn, ngày lập hoá đơn và thành tiền.
* Vì ngày lập hoá đơn không phải khoá chính nên sẽ phải tạo thêm một index cho bảng mới theo cột ngày lập (date) để truy vấn được dễ dàng và nhanh chóng hơn.

**Use** DB\_Order\_Partition

**GO**

**CREATE TABLE** Order\_Partition\_Table

(

**IdOrder** nvarchar(10) PRIMARY KEY,

**date** datetime,

**total\_money** int(15)

)

**O N** OrderDatePartitionScheme (date);

**GO**

**CREATE INDEX** idx\_date\_order\_partition

**ON** Order\_Partition\_Table (date);

**GO**