

# **ĐẠI HỌC ĐÀ NẴNG**

## **TRƯỜNG ĐẠI HỌC KINH TẾ**



# BÁO CÁO CUỐI KỲ

# **ĐỀ TÀI: XÂY DỰNG HỆ THỐNG ĐỀ XUẤT NHÀ HÀNG Ở THÀNH PHỐ ĐÀ NẴNG Dựa TRÊN FOODY.VN**

<b>Học phần</b>	<b>: Quản trị cơ sở dữ liệu</b>
<b>Giảng viên</b>	<b>: TS Hoàng Nguyên Vũ</b>
<b>Nhóm</b>	<b>: 4</b>
<b>Thành viên nhóm</b>	<b>: Phan Thị Kim Ánh 48K29.1</b>
	<b>Phạm Thị Minh Thư 48K29.1</b>
	<b>Lê Thị Hà Vy 48K29.1</b>
	<b>Mai Thị Như Ý 48K29.1</b>

Đà Nẵng, tháng 12 năm 2024

## **MỤC LỤC**

### **PHẦN TRĂM ĐÓNG GÓP CỦA CÁC THÀNH VIÊN**

#### **Chương 1. GIỚI THIỆU ĐỀ TÀI**

1.1. Lý do chọn đề tài

1.2. Mục tiêu

1.3. Hướng giải quyết

#### **Chương 2. CƠ SỞ LÝ THUYẾT**

##### **2.1. Giới thiệu hệ thống đề xuất**

2.1.1. Khái niệm

2.1.2. Các thành phần của hệ thống đề xuất

2.1.3. Một số ứng dụng thực tiễn trong kinh doanh

##### **2.2. Giới thiệu các công cụ, nền tảng và kỹ thuật được sử dụng**

2.2.1. Phần mềm Microsoft Azure

2.2.2. Phương pháp gợi ý dựa theo nội dung (Content-based Filtering)

2.2.3. Một số công cụ và thư viện liên quan khác

##### **2.3 Giới thiệu website Foody thu thập dữ liệu**

#### **Chương 3. THU THẬP, XỬ LÝ VÀ TRỰC QUAN DỮ LIỆU**

##### **3.1. Thu thập dữ liệu từ website Foody.vn (R1)**

3.1.1. Thu thập thông tin từng cửa hàng

3.1.2. Thu thập reviews của từng cửa hàng

##### **3.2. Import dữ liệu vào cơ sở dữ liệu (SQL Server Management Studio) (R2)**

##### **3.3. Tiền xử lý dữ liệu bằng T-SQL (R3)**

3.3.1. Thiết kế và xây dựng cơ sở dữ liệu

3.3.1.1. Xây dựng mô hình thực thể

3.3.1.2. Thiết kế mô hình cơ sở dữ liệu

3.3.2. Thực hiện tiền xử lý dữ liệu trên SQL Server

3.4. Xây dựng cơ chế backup cho dữ liệu (R4)

3.4.1. Lý do lựa chọn loại backup

3.4.2. Quy trình quản lý backup

3.4.3. Xây dựng cơ chế backup cho dữ liệu trên Local

3.4.4. Xây dựng cơ chế backup cho dữ liệu trên Microsoft Azure

3.5. Phân quyền cho cơ sở dữ liệu (R5)

3.6. Trực quan dữ liệu (R6)

3.6.1 Trực quan dữ liệu bằng Tableau

3.6.2 Trực quan dữ liệu bằng Power BI

Chương 4. XÂY DỰNG MÔ HÌNH VÀ ỨNG DỤNG (R7)

4.1. Xây dựng mô hình đề xuất.

4.1.1. Thiết kế mô hình

4.1.2. Xây dựng mô hình

Chương 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Tổng kết

5.2. Hướng phát triển trong tương lai

TÀI LIỆU THAM KHẢO

### PHẦN TRĂM ĐÓNG GÓP CỦA CÁC THÀNH VIÊN

STT	Mã sinh viên	Họ và tên	Nhiệm vụ	Phần trăm
1	221124029104	Phan Thị Kim Ánh		25%
2	221124029143	Phạm Thị Minh Thư		25%
3	221124029153	Lê Thị Hà Vy		25%
4	221124029154	Mai Thị Như Ý		25%

## Chương 1. GIỚI THIỆU ĐỀ TÀI

### 1.1. Lý do chọn đề tài

Đà Nẵng được biết đến là một trong những thành phố du lịch nổi tiếng của Việt Nam, thu hút đông đảo du khách trong và ngoài nước nhờ vẻ đẹp thiên nhiên, sự hiện đại của cơ sở hạ tầng và nền văn hóa ẩm thực phong phú. Với sự phát triển nhanh chóng, thành phố hiện có nhiều cửa hàng, quán ăn, quán cà phê phục vụ nhu cầu ăn uống, giải trí của cả người dân địa phương lẫn khách du lịch.

Tuy nhiên, việc tìm kiếm các cửa hàng phù hợp với nhu cầu cá nhân lại không hề dễ dàng. Để chọn một địa điểm ưng ý, người dùng thường phải tốn nhiều thời gian tra cứu thông tin, đọc đánh giá, hoặc hỏi ý kiến từ bạn bè. Điều này trở nên phức tạp hơn khi mỗi người có những tiêu chí riêng như vị trí, mức giá, chất lượng dịch vụ, không gian hay đặc biệt là các đánh giá từ những người trải nghiệm trước đó.

Xuất phát từ nhu cầu thực tiễn, nhóm quyết định thực hiện đề tài “**Hệ thống đề xuất cửa hàng tại thành phố Đà Nẵng**.” Đề tài này không chỉ giúp gỡ rối bài toán tìm kiếm cửa hàng theo các tiêu chí đánh giá, mà còn tạo ra một công cụ hỗ trợ hữu ích cho cả người dân địa phương, khách du lịch và những người sáng tạo nội dung số như Vlogger.

### 1.2. Mục tiêu

Đề tài của nhóm đã được thực hiện thông qua các mục tiêu sau:

- Nghiên cứu về lý thuyết hệ thống đề xuất:
  - + Tìm hiểu các phương pháp và thuật toán để xuất dựng hệ thống đề xuất, đặc biệt là các thuật toán sử dụng đánh giá người dùng và dữ liệu về cửa hàng.
- Xây dựng mô hình gợi ý cửa hàng:
  - + Đầu ra của mô hình là danh sách các cửa hàng phù hợp với nhu cầu cụ thể của người dùng, được sắp xếp dựa trên các tiêu chí đánh giá như vị trí, mức giá, không gian, và chất lượng dịch vụ.
- Phát triển ứng dụng web:
  - + Xây dựng một ứng dụng web thân thiện với người dùng, nơi người dùng chỉ cần nhập thông tin như địa chỉ, tiêu chí cá nhân (giá cả, loại hình dịch vụ,...) để nhận được gợi ý các cửa hàng phù hợp.

### **1.3. Hướng giải quyết**

Nhóm đã tiến hành các bước giải quyết sau:

#### **a. Thu thập dữ liệu (R1)**

- Sử dụng kỹ thuật web scraping để lấy dữ liệu từ trang Foody.vn, một trang những nền tảng đánh giá cửa hàng phổ biến nhất hiện nay.
- Dữ liệu bao gồm thông tin về cửa hàng (tên, địa chỉ, loại hình, mức giá,...) và đánh giá từ người dùng (điểm số,...)

#### **b. Lưu trữ dữ liệu (R2)**

- Dữ liệu sau khi thu thập được đổ vào cơ sở dữ liệu SQL Server để quản lý và xử lý.
- Thiết kế bảng dữ liệu và các quan hệ phù hợp để tối ưu hóa việc truy vấn và phân tích

#### **c. Tiền xử lý dữ liệu (R3)**

- Sử dụng ngôn ngữ T - SQL và giao diện để thực hiện các thao tác làm sạch dữ liệu, bao gồm loại bỏ dữ liệu trùng lặp, xử lý các giá trị bị thiếu và chuẩn hóa thông tin.

#### **d. Xây dựng cơ chế backup cho dữ liệu (R4)**

- Thiết lập cơ chế backup định kỳ nhằm đảm bảo dữ liệu an toàn, tránh mất mát trong quá trình vận hành.

#### **e. Phân quyền người dùng trên cơ sở dữ liệu (R5)**

- Triển khai phân quyền người dùng trên cơ sở dữ liệu, đảm bảo chỉ những người ủy quyền mới có thể truy cập hoặc chỉnh sửa dữ

#### **f. Xây dựng hệ thống trực quan hóa dữ liệu (R6)**

- Sử dụng Microsoft Power BI để xây dựng các dashboard trực quan và kết hợp Python để tạo các biểu đồ xử lý dữ liệu.

#### **g. Phát triển mô hình học máy (R7)**

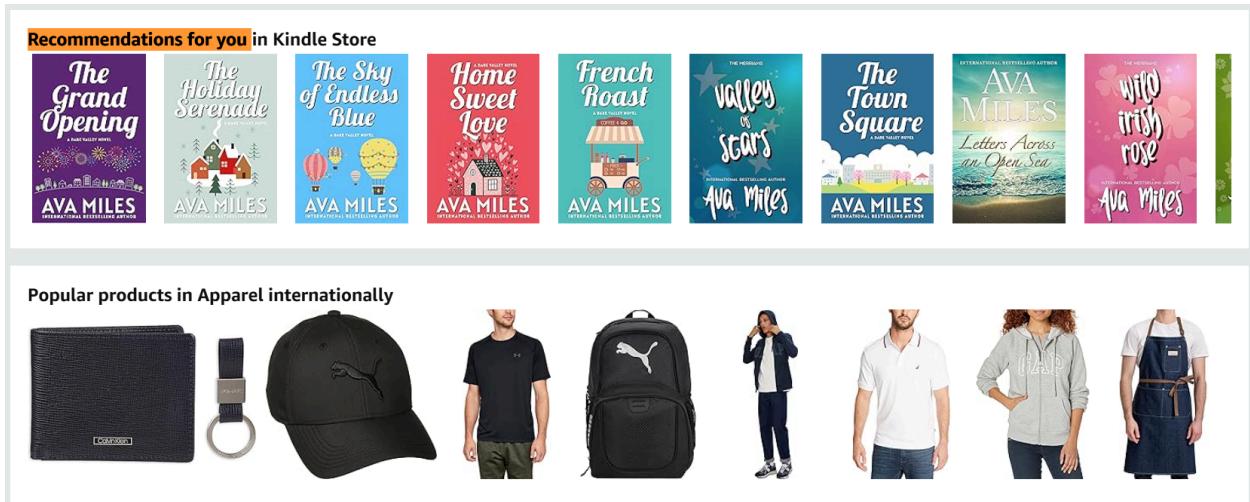
- Sử dụng ngôn ngữ Python và các thư viện học máy để xây dựng mô hình gợi ý cửa hàng.

## **Chương 2. CƠ SỞ LÝ THUYẾT**

### **2.1. Giới thiệu hệ thống đề xuất**

Hệ thống đề xuất (Recommendation System) là một hệ thống được phát triển nhằm mục đích gợi ý sản phẩm, dịch vụ hoặc nội dung phù hợp với nhu cầu và sở thích người dùng. Hệ thống này hoạt động dựa trên việc phân tích hành vi, thông tin người dùng hoặc đặc điểm của sản phẩm để đưa ra gợi ý chính xác và phù hợp nhất.

Ví dụ trong hệ thống bán hàng trực tuyến (chẳng hạn như Amazon), nhằm tối ưu hóa khả năng mua sắm của khách hàng (user), người ta quan tâm đến việc những khách hàng nào đã yêu thích những sản phẩm (item) nào bằng cách dựa vào dữ liệu quá khứ của họ (dữ liệu nào có thể là xếp hạng mà người dùng đã bình chọn trên sản phẩm, thời gian duyệt trên sản phẩm hay số lần click chuột trên sản phẩm,...) từ đó hệ thống sẽ dự đoán được người dùng có thể thích sản phẩm nào và đưa ra những gợi ý phù hợp cho họ. Hình 1 là một ví dụ minh họa về hệ thống gợi ý bán hàng của Amazon.



## 2.2. Giới thiệu các công cụ, nền tảng và kỹ thuật được sử dụng

### 2.2.1. Giới thiệu về Microsoft Azure

#### 2.2.1.1. Giới thiệu về Microsoft Azure

Microsoft Azure là một nền tảng điện toán đám mây công cộng với các giải pháp bao gồm Cơ sở hạ tầng dưới dạng dịch vụ (IaaS), Nền tảng dưới dạng dịch vụ (PaaS) và Phần mềm dưới dạng dịch vụ (SaaS). Microsoft Azure hỗ trợ nhiều ngôn ngữ cùng công cụ lập trình. Azure cung cấp các dịch vụ lưu trữ cơ sở dữ liệu, bao gồm cơ sở dữ liệu quan hệ không máy chủ Azure SQL và cơ sở dữ liệu không quan hệ NoSQL.

#### 2.2.1.2. Giới thiệu về Azure Blob Storage

Lưu trữ Azure Blob là giải pháp lưu trữ đối tượng của Microsoft cho đám mây. Bộ lưu trữ Blob được tối ưu hóa để lưu trữ một lượng lớn dữ liệu phi cấu trúc. "Blob" là viết tắt của "Binary Large Object" và dịch vụ này giúp ta quản lý các đối tượng dữ liệu lớn (có thể lên tới hàng terabyte hoặc petabyte). Các thành phần của Azure Blob Storage:

- Storage account (Tài khoản lưu trữ): nơi chứa tất cả các dịch vụ lưu trữ, bao gồm Blob Storage, Table Storage, Queue Storage, và File Storage. Dữ liệu trong này có thể truy cập được từ mọi nơi trên thế giới qua HTTP hoặc HTTPS và nó tính bền, an toàn và có khả năng mở rộng lớn.
- Container: Là một vùng chứa để tập hợp các blob, tương tự như một thư mục trong hệ thống tệp. Một tài khoản lưu trữ có thể không giới hạn số lượng container và một container có thể lưu trữ không giới hạn số lượng các blobs.
- Blob: Azure storage hỗ trợ 3 loại blob:
  - + Block blobs: dùng để lưu trữ text hoặc file có sẵn, dung lượng của 1 single block blob có thể lên đến 4.75 TB
  - + Append blobs: được tối ưu hóa cho việc thêm dữ liệu vào blob liên tục
  - + Page blobs: dung lượng có thể lên đến 1 TB, sử dụng hiệu quả cho cơ chế đọc/viết. Thường dùng để lưu dữ liệu ổ đĩa của máy ảo (VM).

### **2.2.2. Một số công cụ và thư viện sử dụng**

Giới thiệu về các thư viện sử dụng trong Python

#### **a. Selenium và Selenium WebDriver**

Selenium là thư viện được sử dụng để tự động hóa trình duyệt web, thường dùng trong web scraping (cào dữ liệu) hoặc kiểm thử ứng dụng web. Selenium cho phép điều khiển trình duyệt như một người dùng thực tế: truy cập trang web, tương tác với các phần tử (nhập dữ liệu, click chuột, cuộn trang,...) và trích xuất dữ liệu.

Ưu điểm của Selenium là xử lý được các trang web động (dùng JavaScript) mà các thư viện khác khó thực hiện. Hỗ trợ nhiều trình duyệt như Chrome, Firefox, Edge.

WebDriver là một giao diện lập trình ứng dụng (API) cho phép thực hiện thao tác trên các trình duyệt web như mở trang web, nhập dữ liệu, nhấn nút, kéo thả, chụp ảnh màn hình...

Selenium WebDriver là một dự án mã nguồn mở và là một thành phần của Selenium với khả năng tương tác trực tiếp với trình duyệt và thực hiện các thao tác như click, nhập liệu, kéo thả, chụp ảnh màn hình,... Selenium WebDriver hỗ trợ nhiều ngôn ngữ lập trình và nhiều trình duyệt khác nhau rất thân thiện và dễ sử dụng và nó giao tiếp trực tiếp với trình duyệt mà không cần qua bất kỳ lớp trung gian nào.

#### **b. Pandas**

Pandas là thư viện mạnh mẽ để xử lý và phân tích dữ liệu trong Python. Pandas hỗ trợ các cấu trúc dữ liệu chính như DataFrame và Series, giúp người dùng dễ dàng đọc, ghi, xử lý và thao tác dữ liệu có dạng bảng (giống Excel).

Các chức năng nổi bật bao gồm:

- Đọc/ghi dữ liệu từ nhiều định dạng (CSV, Excel, SQL...).
- Xử lý và làm sạch dữ liệu.
- Phân tích và tổng hợp dữ liệu nhanh chóng.

Đây là công cụ phổ biến trong khoa học dữ liệu, phân tích dữ liệu, và học máy.

#### c. pyodbc

pyodbc là một thư viện Python cho phép kết nối và tương tác với các hệ quản trị cơ sở dữ liệu (SQL Server, MySQL, PostgreSQL...) thông qua giao thức ODBC (Open Database Connectivity).

Tính năng chính:

- Kết nối với nhiều loại cơ sở dữ liệu.
- Thực hiện các truy vấn SQL để lấy dữ liệu hoặc cập nhật cơ sở dữ liệu.

#### d. Geopy

Geopy là một thư viện Python được sử dụng để làm việc với dữ liệu vị trí địa lý. Nó cung cấp các công cụ để geocoding (chuyển đổi địa chỉ thành tọa độ), reverse geocoding (chuyển đổi tọa độ thành địa chỉ), và tính khoảng cách giữa các điểm địa lý.

#### e. Scikit-learn

Scikit-learn là thư viện machine learning (học máy) mã nguồn mở. Scikit-learn cung cấp công cụ và thuật toán học máy để giải quyết nhiều bài toán như phân loại, hồi quy, phân cụm, giảm chiều dữ liệu, và tiền xử lý dữ liệu.

Sklearn.neighbors là một module trong Scikit-learn được sử dụng để giải quyết các bài toán học máy sử dụng các thuật toán dựa trên khoảng cách hoặc lân cận.

Các thuật toán trong module sklearn.neighbors chủ yếu dựa trên nguyên lý "hàng xóm gần nhất" (Nearest Neighbors), phù hợp cho cả bài toán phân loại (classification), hồi quy (regression), và tìm kiếm lân cận (neighbor search).

### 2.3. Giới thiệu website Foody.vn



Hình: Logo Foody.vn

Foody.vn là một trong những nền tảng tìm kiếm, đánh giá và đặt chỗ ăn uống trực tuyến lớn nhất tại Việt Nam, phục vụ nhu cầu khám phá ẩm thực và dịch vụ tiện ích của người dùng. Được ra mắt vào năm 2012, Foody đã nhanh chóng phát triển và trở thành “bản đồ ẩm thực” không thể thiếu cho người yêu thích ăn uống và trải nghiệm dịch vụ.

#### 2.3.1. Chức năng chính của Foody.vn:

Tìm kiếm địa điểm ăn uống: Foody cung cấp thông tin chi tiết về các quán ăn, nhà hàng, quán cà phê, quán bar, tiệm bánh, quán nhậu và các loại hình dịch vụ khác trên khắp Việt Nam. Người dùng có thể tìm kiếm dựa trên nhiều tiêu chí như khu vực, món ăn, giá cả, và loại hình dịch vụ.

Đánh giá và bình luận: Người dùng có thể chia sẻ cảm nhận (bình luận) và đánh giá về chất lượng món ăn, dịch vụ, không gian của các địa điểm. Điểm số đánh giá từ 1 đến 10 được tổng hợp dựa trên các yếu tố như Vị trí, Giá cả, Chất lượng, Không khí và Dịch vụ.

Đặt bàn và giao đồ ăn: Foody còn tích hợp tính năng đặt bàn tại các nhà hàng và quán ăn. Ngoài ra, với sự hợp tác cùng Now.vn (nay là ShopeeFood), người dùng có thể đặt giao đồ ăn tận nhà một cách nhanh chóng và tiện lợi.

Khuyến mãi và ưu đãi: Foody liên tục cập nhật thông tin khuyến mãi, voucher và mã giảm giá cho người dùng, giúp họ có những bữa ăn ngon với chi phí tiết kiệm.

#### 2.3.2. Đối tượng người dùng:

Người yêu thích ẩm thực: Foody.vn là điểm đến lý tưởng để khám phá món ngon, địa điểm ăn uống mới lạ. Những thực khách đam mê trải nghiệm các quán ăn sẽ tìm được mọi thông tin cần thiết trên nền tảng.

Chủ quán và doanh nghiệp: Các chủ quán có thể đăng tải thông tin về cửa hàng của mình trên Foody để tiếp cận nhiều khách hàng hơn, tăng cơ hội kinh doanh.

### **2.3.3. Điểm nổi bật của Foody.vn:**

Kho dữ liệu địa điểm phong phú: Foody sở hữu hàng trăm nghìn địa điểm ăn uống trải dài từ Bắc vào Nam, bao gồm quán ăn vỉa hè, nhà hàng sang trọng, quán cà phê và các dịch vụ ăn uống khác.

Giao diện thân thiện và dễ sử dụng: Website và ứng dụng Foody được thiết kế khoa học, giúp người dùng dễ dàng tìm kiếm, lọc thông tin và đọc đánh giá.

Tính minh bạch và khách quan: Đánh giá trên Foody được thực hiện bởi cộng đồng người dùng, giúp thông tin trở nên đáng tin cậy và minh bạch. Các quán ăn có thể nhận phản hồi trực tiếp để cải thiện chất lượng dịch vụ.

Hệ sinh thái đa dạng: Foody không chỉ dừng lại ở mảng đánh giá mà còn mở rộng sang giao đồ ăn, đặt bàn, đặt tiệc và cung cấp mã giảm giá. Điều này giúp người dùng có trải nghiệm trọn vẹn hơn khi sử dụng dịch vụ.

### **2.3.4. Tổng kết:**

Foody.vn là một nền tảng tiên phong trong lĩnh vực tìm kiếm, đánh giá và đặt dịch vụ ăn uống tại Việt Nam. Không chỉ giúp người dùng tìm kiếm địa điểm ăn uống nhanh chóng và chính xác, Foody còn tạo ra một cộng đồng chia sẻ kinh nghiệm ăn uống phong phú, góp phần nâng cao chất lượng dịch vụ ăn uống trên cả nước.

## **Chương 3. THU THẬP, XỬ LÝ VÀ TRỰC QUAN DỮ LIỆU**

### **3.1. Thu thập dữ liệu từ website Foody.vn (R1)**

#### **3.1.1. Thu thập thông tin từ cửa hàng**

Đầu tiên, nhóm sẽ thiết lập các tùy chọn cho Selenium WebDriver. Trong đó có

- no-sandbox: giúp bảo vệ và không cho các phần mềm độc hại xâm nhập vào máy tính để hạn chế hỏng hệ thống máy hoặc rò rỉ các thông tin cá nhân.
- disable-dev-shm-usage: Vô hiệu hóa việc sử dụng hệ thống tệp bộ nhớ chia sẻ /dev/shm, điều này có thể làm giảm mức sử dụng bộ nhớ trong quá trình chạy chương trình.

thiết lập các tùy chọn cho trình duyệt Chrome      Vô hiệu hóa chế độ sandbox của trình duyệt

```
# Thiết lập các tùy chọn cho Selenium
options = Options()
options.add_argument('--no-sandbox')
options.add_argument('--disable-dev-shm-usage')
```

Chuyển hướng dữ liệu sang bộ nhớ ổ đĩa thay vì dùng shared memory

Hình: Code thiết lập các tùy chọn cho Selenium WebDriver

Sau đây sẽ khởi tạo trình duyệt với Selenium WebDriver

```
# Khởi tạo trình duyệt
service = Service('D:/Chrome_for_Dev/chromedriver-win64/chromedriver.exe')
driver = webdriver.Chrome(service=service, options=options)
```

Hình: Code khởi tạo trình duyệt với Selenium WebDriver

Sau đó sẽ đăng nhập vào trang web vì trang web Foody.vn cần đăng nhập thì mới có thể xem toàn bộ các cửa hàng có trong trang web

```
# Hàm đăng nhập vào Foody
def login_to_foody(driver, username, password):
    driver.get("https://id.foody.vn/account/login?returnUrl")
    time.sleep(3) # Chờ trang tải

    # Nhập thông tin đăng nhập
    try:
        username_input = driver.find_element(By.XPATH, "//input[@name='Email']")
        password_input = driver.find_element(By.XPATH, "//input[@name='Password']")
        login_button = driver.find_element(By.XPATH, "//input[@type='submit']")

        username_input.send_keys(username)
        password_input.send_keys(password)
        login_button.click() # Nhấn nút đăng nhập
        time.sleep(3) # Chờ một chút để đăng nhập
    except Exception as e:
        print("Lỗi khi đăng nhập:", e)
```

login\_to\_foody(driver, '...', ...) # Thông tin đăng nhập

Tài khoản

Mật khẩu

Hình: Code đăng nhập vào Foody.vn

Sau khi đã đăng nhập thành công thì sẽ tiến hành thu thập các dữ liệu cần thiết trong trang web. Đầu tiên, nhóm sẽ lấy tên và link của cửa hàng dựa theo cấu trúc html của trang web và vì Foody.vn là một trang web có cấu trúc cuộn trang nên sẽ phải tìm và bấm vào nút “Xem thêm” để tiếp tục lấy dữ liệu

```

while True:
    try:
        # Lấy các liên kết và tên cửa hàng sau khi cuộn
        items = WebDriverWait(driver, 10).until(
            EC.presence_of_all_elements_located((By.CLASS_NAME, 'title'))
        )

        for item in items:
            try:
                link = item.find_element(By.TAG_NAME, 'a').get_attribute('href') → Lấy link và tên cửa hàng
                name = item.find_element(By.TAG_NAME, 'a').text
                if link not in restaurant_links: # Kiểm tra trùng lặp
                    restaurant_links.append(link)
                    restaurant_names.append(name)
            except Exception as e:
                print("Lỗi khi lấy tên hoặc liên kết:", e)

        # Tìm nút "Xem thêm" và nhấn vào
        load_more_button = WebDriverWait(driver, 5).until(
            EC.element_to_be_clickable((By.XPATH, "//a[contains(@class, 'fd-btn-more') and contains(., 'Xem thêm')]"))
        )
        load_more_button.click()
        time.sleep(3) # Chờ để nội dung mới được tải về
    except Exception as e:
        print("Không còn nút 'Xem thêm' hoặc có lỗi:", e)
        break # Nếu không còn nút "Xem thêm", thoát khỏi vòng lặp

```

Hình: Code thu thập thông tin cửa hàng

Tiếp theo, nhóm sẽ lấy các thông tin như: tổng số lượt đánh giá của cửa hàng, tên địa chỉ chi tiết, quận và thành phố, khoảng giá cửa hàng cung cấp, giờ mở cửa và loại cửa hàng dựa theo cấu trúc html tương ứng

```

# Lấy thông tin chi tiết về đánh giá
review_count = driver.find_element(By.XPATH, "//div[@itemprop='reviewCount']").text.strip()

# Lấy địa chỉ và quận
address_element = driver.find_element(By.CLASS_NAME, 'res-common-add')
street_address = address_element.find_element(By.XPATH, "./span[@itemprop='streetAddress']").text
district = address_element.find_element(By.XPATH, "./span[@itemprop='addressLocality']").text
city = address_element.find_element(By.XPATH, "./span[@itemprop='addressRegion']").text

# Lấy khoảng giá
price_range = driver.find_element(By.XPATH, "//div[@class='res-common-minmaxprice']/span[@itemprop='priceRange']").text.strip()

# Lấy giờ mở cửa
opening_hours_element = driver.find_element(By.XPATH, "//div[contains(@class, 'micro-timesopen')]/span[3]")
opening_hours = opening_hours_element.text.strip() # Giờ mở cửa

# Lấy loại cửa hàng
category_element = driver.find_element(By.XPATH, "//div[@class='category-items']/a")
category_name = category_element.text.strip()

```

Hình: Code thu thập thông tin cửa hàng

Sau khi đã lấy được các thông tin cần thiết thì sẽ lưu các thông tin đó vào một danh sách

```
# Lưu thông tin vào danh sách
details.append({
    'name': name,
    'link': link,
    'review_count': review_count,
    'street_address': street_address,
    'district': district,
    'city': city,
    'opening_hours': opening_hours,
    'price_range': price_range,
    'category_name' : category_name,
    'reviews': reviews
})
```

*Hình: Code lưu thông tin cửa hàng vào danh sách*

### 3.1.2. Thu thập các đánh giá chi tiết của từng cửa hàng

Để lấy các đánh giá chi tiết của từng cửa hàng thì sẽ phải vòi từng link cửa hàng, sau đó sẽ lấy các thông tin dựa trên html tương ứng. Đầu tiên sẽ lấy tên của người đánh giá và thời gian người đó đánh giá trong từng cửa hàng

```
# Lấy tên người đánh giá
user_name = review.find_element(By.CLASS_NAME, 'ru-username.ng-binding').text.strip()
rating = review.find_element(By.CLASS_NAME, 'review-points.ng-scope').text.strip()

# Lấy thời gian đánh giá và HTML của thời gian đánh giá
review_time = review.find_element(By.CLASS_NAME, 'ru-time.ng-binding').text.strip()
```

*Hình: Code thu thập các đánh giá chi tiết của cửa hàng*

Sau đó sẽ lấy thông tin về từng chỉ số của mỗi loại đánh giá và các chỉ số này bị ẩn và cần phải di chuyển chuột tới nó mới hiện ra. Các chỉ số đánh giá bao gồm chỉ số về vị trí, giá cả, chất lượng, dịch vụ và không khí của cửa hàng

```

# Tìm phần tử để di chuyển chuột tới
rating_element = review.find_element(By.CLASS_NAME, 'review-points.ng-scope')
actions.move_to_element(rating_element).perform()

# Đợi popup hiện ra
time.sleep(2)

# Lấy thông tin chi tiết từ popup bằng JavaScript
popup_info = driver.execute_script("""
    var popup = document.getElementById('fdDlgReviewRating');
    if (popup) {
        return {
            position: popup.querySelector('b[ng-bind="Model.Position"]').innerText || null,
            price: popup.querySelector('b[ng-bind="Model.Price"]').innerText || null,
            quality: popup.querySelector('b[ng-bind="Model.Food"]').innerText || null,
            service: popup.querySelector('b[ng-bind="Model.Services"]').innerText || null,
            atmosphere: popup.querySelector('b[ng-bind="Model.Atmosphere"]').innerText || null
        };
    }
    return null;
""")

```

*Hình: Code thu thập các đánh giá chi tiết của cửa hàng*

Vì Foody.vn là một trang web có dạng cuộn trang nên trong các đánh giá cũng phải tìm “Xem thêm bình luận” để có thể thu thập tất cả bình luận có thể có trong từng cửa hàng

```

# Tìm và nhấn nút "Xem thêm bình luận" nếu có
try:
    load_more_button = WebDriverWait(driver, 5).until(
        EC.element_to_be_clickable((By.XPATH, "//a[contains(@class, 'fd-btn-more') and contains(., 'Xem thêm bình luận')]"))
    )
    load_more_button.click()
    time.sleep(3) # Chờ bình luận tải thêm

```

*Hình: Code hiển thị thêm các đánh giá của cửa hàng*

Cuối cùng, sau khi đã lấy được tất cả thông tin thì sẽ lưu vào một danh sách

```

if popup_info:
    reviews.append({
        'user_name': user_name,
        'review_time': review_time,
        'rating': rating,
        'position': popup_info['position'],
        'price': popup_info['price'],
        'quality': popup_info['quality'],
        'service': popup_info['service'],
        'atmosphere': popup_info['atmosphere'],
    })

```

*Hình: Code lưu các đánh giá chi tiết của cửa hàng vào danh sách*

## 3.2. Import dữ liệu vào cơ sở dữ liệu (SQL Server Management Studio) (R2)

### 3.2.1 Tạo cơ sở dữ liệu

Sử dụng **SQL Server Management Studio (SSMS)** để tạo cơ sở dữ liệu lưu trữ thông tin cửa hàng và đánh giá từ dữ liệu crawl trên trang **Foody.vn**.

```

]create database Foody_CrawlData
use Foody_CrawlData

```

### 3.2.2 Tạo bảng dữ liệu

Dựa trên cấu trúc dữ liệu thu thập được, xây dựng 2 bảng chính:

- da\_nang: chứa thông tin cửa hàng
- reviews\_dn: chứa thông tin đánh giá

#### a. Tạo bảng da\_nang:

Bảng này chứa thông tin chi tiết về các cửa hàng tại Đà Nẵng.

```

CREATE TABLE da_nang (
    id_quan      INT PRIMARY KEY IDENTITY(1,1),
    [name]        NVARCHAR(255),
    link          NVARCHAR(500),
    review_count  INT,
    street_address NVARCHAR(255),
    district      NVARCHAR(100),
    city          NVARCHAR(100),
    opening_hours NVARCHAR(100),
    price_range   NVARCHAR(50),
    category_name NVARCHAR(100)
);

```

Giải thích các cột trong bảng da\_nang:

Cột	Ý nghĩa
id_quan	Mã định danh cửa hàng
name	Tên cửa hàng
link	URL liên kết tới thông tin cửa hàng
review_count	Tổng số đánh giá
street_address	Địa chỉ đường phố
district	Quận
city	Thành phố
opening_hours	Giờ mở cửa
price_range	Mức giá
category_name	Danh mục (nhà hàng, quán cafe,...)

## b. Tạo bảng reviews\_dn

Bảng này lưu trữ các đánh giá từ người dùng cho từng cửa hàng.

```
|CREATE TABLE reviews_dn (
    id_review    INT PRIMARY KEY IDENTITY(1,1),
    id_quan      INT,
    [user_name]   NVARCHAR(255),
    review_time  NVARCHAR(100),
    rating       float,
    position     float,
    price        float,
    quality      float,
    [service]    float,
    atmosphere   float,
    FOREIGN KEY (id_quan) REFERENCES da_nang(id_quan)
);
```

Giải thích các cột trong bảng reviews\_dn:

Cột	Ý nghĩa
id_review	Mã định danh của từng đánh giá
id_quan	Mã cửa hàng được đánh giá
user_name	Tên người dùng thực hiện đánh giá
review_time	Thời gian viết đánh giá
rating	Điểm đánh giá tổng thể
position	Điểm đánh giá vị trí của cửa hàng
price	Điểm đánh giá mức giá
quality	Điểm đánh giá chất lượng
service	Điểm đánh giá dịch vụ
atmospher	Điểm đánh giá không gian cửa hàng

### 3.2.3 Import dữ liệu vào database

Trong python, kết nối SQL Server, sau đó Ghi dữ liệu vào cơ sở dữ liệu bằng cách xây dựng hàm lưu dữ liệu. Dữ liệu được chia làm 2 phần:  
Thông tin cửa hàng lưu vào bảng da\_nang.

Đánh giá cửa hàng lưu vào bảng reviews\_dn và được liên kết với id\_quan của cửa hàng.

```

def save_to_sql_server(data):
    connection_string = "Driver={SQL Server};Server=127.0.0.1;Database=NTHU;UID=sa;PWD=123456;" 
    try:
        conn = pyodbc.connect(connection_string)
        cursor = conn.cursor()
        for record in data:
            # Lưu thông tin tổng quát của quán vào bảng thong_tin_quan
            cursor.execute("""
                INSERT INTO da_nang (name, link, review_count,
                                     street_address, district, city,
                                     opening_hours, price_range, category_name)
                OUTPUT INSERTED.id_quan
                VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)
            """, (
                record['name'], record['link'], record['review_count'],
                record['street_address'], record['district'], record['city'],
                record['opening_hours'], record['price_range'], record['category_name']
            ))
            # Lấy ID của quán vừa lưu bằng cách sử dụng SCOPE_IDENTITY()
            id_quan = cursor.fetchone()[0]
            # Lưu thông tin đánh giá vào bảng reviews
            for review in record['reviews']:
                cursor.execute("""
                    INSERT INTO reviews_dn (id_quan, user_name, review_time, rating, position, price, quality, service, atmosphere)
                    VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)
                """, (
                    id_quan, review['user_name'], review['review_time'], review['rating'], review['position'],
                    review['price'], review['quality'], review['service'], review['atmosphere']
                ))
        conn.commit()
    except pyodbc.Error as e:
        print(f"Error: {e}")

```

### 3.3. Tiền xử lý dữ liệu bằng T-SQL (R3)

#### 3.3.1. Thiết kế và xây dựng cơ sở dữ liệu

##### 3.3.1.1. Xây dựng mô hình thực thể - quan hệ (ERD)

Trong quá trình thiết kế và xây dựng cơ sở dữ liệu, mô hình thực thể quan hệ đã được xây dựng để biểu diễn các thực thể, thuộc tính của từng thực thể và mối quan hệ giữa chúng.

###### a. Các thực thể và thuộc tính

Bảng : Mô tả thực thể và thuộc tính

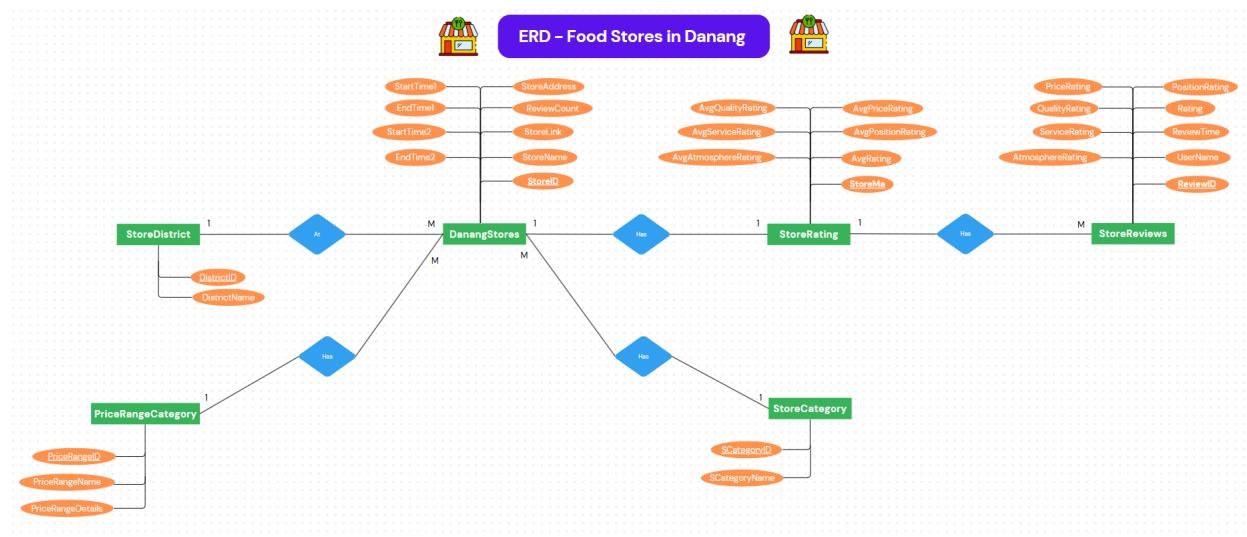
STT	Thực thể	Mô tả	Thuộc tính
1	StoreDistrict	Đại diện các quận nơi có các cửa hàng	<ul style="list-style-type: none"> <li>- DistrictID (Khóa chính)</li> <li>- DistrictName</li> </ul>
2	DanangStores	Đại diện cho các cửa hàng	<ul style="list-style-type: none"> <li>- StoreID (Khóa chính)</li> <li>- StoreName</li> <li>- StoreAddress</li> <li>- StoreLink</li> <li>- StartTime1, EndTime1 (Giờ mở cửa - ca 1)</li> </ul>

			<ul style="list-style-type: none"> <li>- StartTime2, EndTime2 (Giờ mở cửa - ca 2)</li> <li>- ReviewCount</li> </ul>
3	StoreRating	Tổng hợp các đánh giá trung bình của cửa hàng	<ul style="list-style-type: none"> <li>- StoreMa (Khóa chính, liên kết với StoreID)</li> <li>- AvgQualityRating</li> <li>- AvgServiceRating</li> <li>- AvgAtmosphereRating</li> <li>- AvgPriceRating</li> <li>- AvgPositionRating</li> <li>- AvgRating</li> </ul>
4	StoreReviews	Chi tiết đánh giá của khách hàng đối với cửa hàng	<ul style="list-style-type: none"> <li>- ReviewID (Khóa chính)</li> <li>- UserName</li> <li>- ReviewTime</li> <li>- Rating</li> <li>- QualityRating</li> <li>- ServiceRating</li> <li>- AtmosphereRating</li> <li>- PriceRating</li> <li>- PositionRating</li> </ul>
5	StoreCategory	Phân loại loại hình cửa hàng	<ul style="list-style-type: none"> <li>- SCategoryID (Khóa chính)</li> <li>- SCategoryName</li> </ul>
6	PriceRangeCategory	Danh mục mức giá của cửa	<ul style="list-style-type: none"> <li>- PriceRangeID (Khóa chính)</li> <li>- PriceRangeName</li> <li>- PriceRangeDetails</li> </ul>

**b. Mối quan hệ giữa các thực thể**

Tên thực thể	Mối quan hệ	Cách tham chiếu
StoreDistrict và DanangStores	1 - N (Một quận có thể chứa nhiều cửa hàng)	Khóa ngoại DistrictID trong bảng DanangStores

DanangStores và StoreRating	1 - 1 (Mỗi cửa hàng có một đánh giá tổng hợp)	Liên kết qua khóa ngoại StoreID trong bảng StoreRating
DanangStores và StoreReviews	1 - N (Một cửa hàng có nhiều đánh giá từ khách hàng)	Khóa ngoại StoreID trong bảng StoreReviews.
DanangStores và StoreCategory	N - 1 (Nhiều cửa hàng có thể thuộc một loại hình cửa hàng)	Khóa ngoại SCategoryID trong bảng DanangStores
DanangStores và PriceRangeCategory	N - 1 (Nhiều cửa hàng có thể nằm trong một khoảng giá cụ thể)	Khóa ngoại PriceRangeID trong bảng DanangStores



Hình : Mô hình quan hệ thực thể ERD

### 3.3.1.2. Thiết kế mô hình cơ sở dữ liệu

Dựa trên mô hình thực thể - quan hệ (ERD), quá trình thiết kế mô hình cơ sở dữ liệu được thực hiện nhằm chuyển đổi các thực thể và mối quan hệ thành các bảng trong SQL Server.

#### a. Thiết kế các bảng dữ liệu

Bảng Store District

Tên cột	Kiểu dữ liệu	Ràng buộc	Mô tả
DistrictID	int	Primary key	Mã quận (khóa chính)
DistrictName	nvarchar(50)	Not null	Tên quận

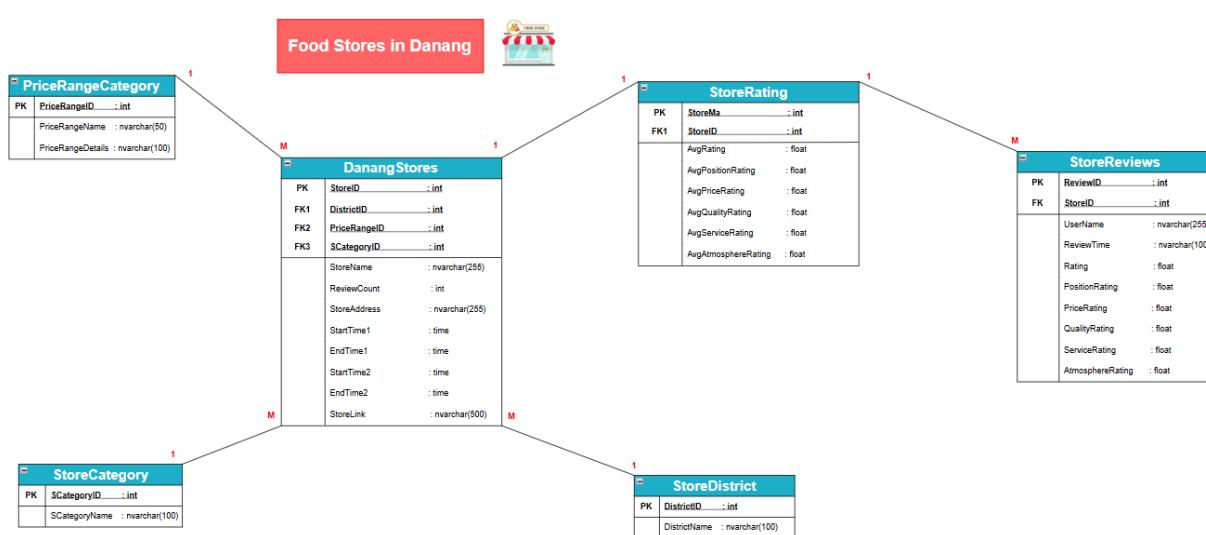
Bảng DanangStores

Tên cột	Kiểu dữ liệu	Ràng buộc	Mô tả
StoreID	int	Primary key	Mã cửa hàng (Khóa chính)
StoreName	nvarchar(100)	Not null	Tên cửa hàng
StoreAddress	nvarchar(200)	Not null	Địa chỉ
StoreLink	nvarchar(255)	Not null	Liên kết thông tin cửa hàng
StartTime1	time	Not null	Giờ mở cửa (Ca 1)
EndTime1	time	Not null	Giờ đóng cửa (ca 1)
StartTime2	time	null	Giờ mở cửa (Ca 2)
EndTime2	time	null	Giờ đóng cửa (ca 2)
ReviewCount	int	Default 0	Tổng số lượt đánh giá
DistrictID	int	Foreign key	Liên kết đến bảng StoreDistrict.
SCategoryID	int	Foreign key	Liên kết đến bảng StoreCategory
PriceRangeID	int	Foreign key	Liên kết đến bảng PriceRangeCategory

Bảng StoreRating

Tên cột	Kiểu dữ liệu	Ràng buộc	Mô tả
StoreMa	int	Primary key	Liên kết với StoreID.

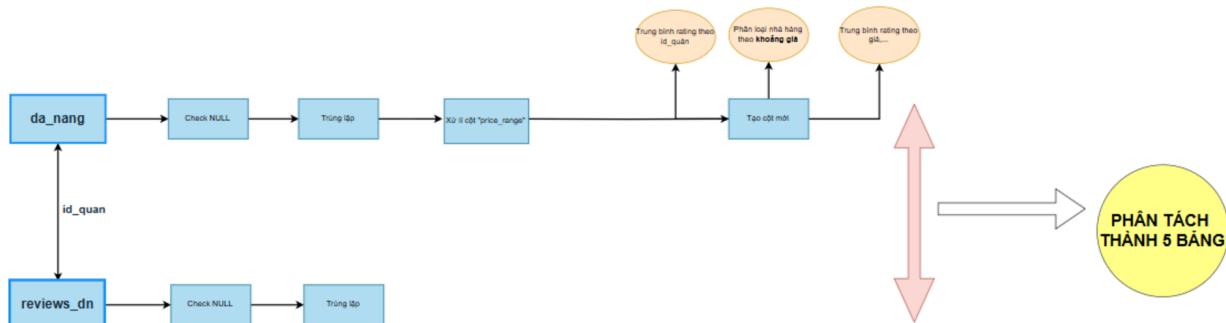
AvgQualityRating	float	null	Điểm đánh giá chất lượng
AvgServiceRating	float	null	Điểm đánh giá dịch vụ
AvgAtmosphereRating	float	null	Điểm đánh giá không gian.
AvgPriceRating	float	null	Điểm đánh giá giá cả.
AvgPositionRating	float	null	Điểm đánh giá vị trí.
AvgRating	float		Điểm đánh giá trung bình



Hình : Thiết kế cơ sở dữ liệu

### 3.3.2. Thực hiện tiền xử lý dữ liệu trên SQL Server

#### 3.3.2.1. Tổng quan các bước tiền xử lý



### 3.3.2.2. Thực hiện tiền xử lý bằng T-SQL

#### a. Xử lý các giá trị NULL

##### - Kiểm tra các giá trị NULL

Nhóm tạo một thủ tục để truy vấn các dòng dữ liệu với điều kiện tồn tại NULL trong bất kỳ 1 cột nào của dòng đó, áp dụng riêng với hai bảng da\_nang và review\_dn.

```
CREATE PROCEDURE CheckForNullValues
AS
BEGIN
    SET NOCOUNT ON;
    SELECT *
    FROM [da_nang]
    WHERE [id_quan] IS NULL
        OR [name] IS NULL
        OR [link] IS NULL
        OR [review_count] IS NULL
        OR [street_address] IS NULL
        OR [district] IS NULL
        OR [city] IS NULL
        OR [opening_hours] IS NULL
        OR [price_range] IS NULL;
END;
GO

EXEC CheckForNullValues;

CREATE PROCEDURE CheckForNullReviewValues
AS
BEGIN
    SET NOCOUNT ON;
    SELECT *
    FROM [reviews_dn]
    WHERE [id_review] IS NULL
        OR [id_quan] IS NULL
        OR [user_name] IS NULL
        OR [rating] IS NULL
        OR [position] IS NULL
        OR [price] IS NULL
        OR [service] IS NULL
        OR [atmosphere] IS NULL;
END;
GO

EXEC CheckForNullReviewValues;
```

Kết quả nhóm thấy được như sau:

- Bảng da\_nang không có bất kỳ dòng nào chứa giá trị rỗng.
- Bảng review\_dn hiển thị các dòng chứa giá trị NULL.

Đối với bảng da\_nang

id_quan	name	link	review_count	street_address	district	city	opening_hours	price_range

Đối với bảng review\_dn

	id_review	id_quan	user_name	review_time	rating	position	price	quality	service	atmosphere
1	41	3	Anh Tuan Tran	2/7/2022 19:41	5	NULL	NULL	NULL	NULL	NULL
2	1099	12	Huyền Trương	12/5/2020 19:22	4.8	NULL	NULL	NULL	NULL	NULL
3	3704	27	Ngoc Nguyen	30/6/2021 18:34	1.2	NULL	NULL	NULL	NULL	NULL
4	22864	101	Mun	10/10/2022 8:32	3	NULL	NULL	NULL	NULL	NULL

##### - Xử lý các giá trị NULL

Nếu tất cả các cột điểm khác đều null nhưng cột rating không null thì ta sẽ điền giá trị của cột rating (của dòng đó) và tất cả các cột điểm khác của đánh giá đó, vì rating vốn được tính trung bình cộng từ các cột điểm đó.

```
--2. Xu li cac cot co gia tri NULL o bang review_dn
CREATE PROCEDURE FillOrDeleteNullValues
AS
BEGIN
    --Cap nhat cac cot co gia tri NULL dua vao gia tri rating
    UPDATE reviews_dn
    SET
        position = ISNULL(position, rating),
        price = ISNULL(price, rating),
        quality = ISNULL(quality, rating),
        service = ISNULL(service, rating),
        atmosphere = ISNULL(atmosphere, rating)
    WHERE
        rating IS NOT NULL AND
        (position IS NULL AND
        price IS NULL AND
        quality IS NULL AND
        service IS NULL AND
        atmosphere IS NULL
        )
    END;
```

Nếu cột rating null nhưng các cột khác không null, ta sẽ tính toán được giá trị của ô rating từ các ô điểm khác bằng cách tính trung bình các ô điểm hiện có của đánh giá đó.

```
-- Tinh gia tri rating neu rating hien tai la NULL, nhung cac cot khac khong phai NULL
UPDATE dn_details
SET
    rating = (ISNULL(position, 0) +
               ISNULL(price, 0) +
               ISNULL(quality, 0) +
               ISNULL([service], 0) +
               ISNULL(atmosphere, 0)) /
               (CASE
                   WHEN position IS NOT NULL THEN 1 ELSE 0 END +
                   CASE
                       WHEN price IS NOT NULL THEN 1 ELSE 0 END +
                   CASE
                       WHEN quality IS NOT NULL THEN 1 ELSE 0 END +
                   CASE
                       WHEN [service] IS NOT NULL THEN 1 ELSE 0 END +
                   CASE
                       WHEN atmosphere IS NOT NULL THEN 1 ELSE 0 END
               )
WHERE
    rating IS NULL
    AND (position IS NOT NULL
    AND price IS NOT NULL
    AND quality IS NOT NULL
    AND service IS NOT NULL
    AND atmosphere IS NOT NULL
    )
```

Còn đối với trường hợp tất cả các cột điểm đều null thì nhóm quyết định xóa bỏ những cột này. Nhận thấy số lượng những trường hợp này chiếm phần rất nhỏ trong bộ dữ liệu mà nhóm cần được, đồng thời không có căn cứ để bỏ khuyết cho các giá trị thiếu giống như 2 trường hợp trước đó, đồng thời số những dòng này chiếm một phần cực kỳ nhỏ, không ảnh hưởng tới dữ liệu khi xóa.

```

-- Xóa các dòng nếu tất cả các cột liên quan đều NULL
DELETE FROM dn_details
WHERE
    rating      IS NULL
    AND position  IS NULL
    AND price     IS NULL
    AND quality    IS NULL
    AND [service]   IS NULL
    AND atmosphere IS NULL
END
-- Chạy
EXEC FillOrDeleteNullValues

```

### b. Xử lý các giá trị trùng lặp

- Kiểm tra các giá trị trùng lặp

Sau khi kiểm tra các giá trị trùng lặp ở từng bảng, nhóm thấy sự trùng lặp chỉ xảy ra ở bảng reviews\_dn với 20332 bộ giá trị không được coi là bộ giá trị duy nhất ở bảng này.

<pre> SELECT SUM(DuplicateCount - 1) AS TotalDuplicateCount FROM (     SELECT COUNT(*) AS DuplicateCount     FROM da_nang     GROUP BY [id_quan],         [name],         [link],         [review_count],         [street_address],         [district],         [city],         [opening_hours],         [price_range]     HAVING COUNT(*) &gt; 1 ) AS Duplicates_dn; </pre>	<pre> SELECT SUM(DuplicateCount - 1) AS TotalDuplicateCount FROM (     SELECT COUNT(*) AS DuplicateCount     FROM reviews_dn     GROUP BY [id_quan],         [user_name],         [review_time],         [rating],         [position],         [price],         [quality],         [service],         [atmosphere]     HAVING COUNT(*) &gt; 1 ) AS Duplicate; </pre>

- Xử lý các giá trị trùng lặp

Việc xử lý dữ liệu trùng đơn giản là xóa đi mà chỉ giữ lại dòng đầu tiên của nhóm các dữ liệu trùng, đảm bảo mỗi bộ (dòng) là duy nhất trong bộ dữ liệu. Kết quả sau khi xử lý là bảng reviews\_dn còn 3043 dòng.

```

CREATE PROCEDURE RemoveDuplicates
AS
BEGIN
    SET NOCOUNT ON;

    WITH DuplicateCTE AS (
        SELECT
            [id_review],
            [id_quan],
            [user_name],
            [review_time],
            [rating],
            [position],
            [price],
            [quality],
            [service],
            [atmosphere],
            ROW_NUMBER() OVER (PARTITION BY [id_quan], [user_name], [review_time],
                                [rating], [position], [price],
                                [quality], [service], [atmosphere]
                               ORDER BY id_review)
        AS RowNum
        FROM
            reviews_dn
    )
    DELETE FROM DuplicateCTE
    WHERE RowNum > 1;

    SET NOCOUNT OFF;
END;
-- Gọi thủ tục
EXEC RemoveDuplicates

```

SoDong

1

3043

### c. Tạo thêm các thuộc tính mới

Đầu tiên, nhóm tạo các cột rỗng ở bảng *da\_nang* để chứa điểm đánh giá trung bình của các tiêu chí và điểm toàn diện cho mỗi nhà hàng, sau đó cập nhật các giá trị cho các cột này bằng cách tính trung bình các cột điểm ở bảng *reviews\_dn* theo *id\_quan*.

```

CREATE PROCEDURE UpdateDaNang
AS
    -- Thêm cột mới vào bảng da_nang
BEGIN
    ALTER TABLE da_nang
    ADD avg_rating      DECIMAL(10, 2),
        avg_position    DECIMAL(10, 2),
        avg_price       DECIMAL(10, 2),
        avg_quality     DECIMAL(10, 2),
        avg_service     DECIMAL(10, 2),
        avg_atmosphere  DECIMAL(10, 2)
END

EXEC UpdateDaNang

```

```

---Cap nhat gia tri vao cac thuoc tinh moi cua bang da_nang
CREATE PROCEDURE CapNhatRating
AS
BEGIN
    UPDATE da_nang
    SET
        avg_rating      = (SELECT AVG(rating)           FROM reviews_dn WHERE reviews_dn.id_quan = da_nang.id_quan),
        avg_position    = (SELECT AVG(position)         FROM reviews_dn WHERE reviews_dn.id_quan = da_nang.id_quan),
        avg_price       = (SELECT AVG(price)            FROM reviews_dn WHERE reviews_dn.id_quan = da_nang.id_quan),
        avg_quality     = (SELECT AVG(quality)          FROM reviews_dn WHERE reviews_dn.id_quan = da_nang.id_quan),
        avg_service     = (SELECT AVG([service])         FROM reviews_dn WHERE reviews_dn.id_quan = da_nang.id_quan),
        avg_atmosphere  = (SELECT AVG(atmosphere)        FROM reviews_dn WHERE reviews_dn.id_quan = da_nang.id_quan);
END;

EXEC CapNhatRating

```

Vì khoảng giá ở mỗi quán rất khác nhau, nên cột price\_range có rất nhiều giá trị khác nhau, để dễ dàng cho việc phân loại, nhóm chọn khoảng giá như là một chiêu để gộp dữ liệu nên nhóm sẽ xử lý cột này.

Tạo một cột average\_price để tính giá trung bình của từng quán, điền giá trị cho cột này bằng cách: loại bỏ ký tự đặc biệt trong cột price\_range, lấy giá trị đầu cộng giá trị sau chia đôi.

Tạo cột price\_category để phân loại giá của các cửa hàng.

```

CREATE PROCEDURE AddPriceCategoryColumn
AS
BEGIN
    -- Thêm cột price_category với kiểu dữ liệu NVARCHAR
    ALTER TABLE da_nang
    ADD price_category NVARCHAR(50);
END;
EXEC AddPriceCategoryColumn;

CREATE PROCEDURE CategorizePrice
AS
BEGIN
    -- Cập nhật cột price_category dựa trên giá trị average_price
    UPDATE da_nang
    SET price_category =
    CASE
        WHEN average_price < 50000 THEN N'Thấp'
        WHEN average_price >= 50000 AND average_price < 100000 THEN N'Trung bình thấp'
        WHEN average_price >= 100000 AND average_price < 200000 THEN N'Trung bình'
        WHEN average_price >= 200000 AND average_price < 300000 THEN N'Trung bình cao'
        WHEN average_price >= 300000 AND average_price < 50000 THEN N'Cao'
        ELSE N'Rất cao'
    END
    WHERE average_price IS NOT NULL;
END;
EXEC CategorizePrice;

```

Kết quả có được như hình dưới đây:

price_range_id	price_range	average_price	price_category
1	10.000đ - 15.000đ	12500	Thấp
2	10.000đ - 20.000đ	15000	Thấp
3	10.000đ - 200.000đ	105000	Trung bình
4	10.000đ - 25.000đ	17500	Thấp
5	10.000đ - 30.000đ	20000	Thấp
6	10.000đ - 300.000đ	155000	Trung bình
7	10.000đ - 33.000đ	21500	Thấp
8	10.000đ - 41.000đ	25500	Thấp
9	10.000đ - 45.000đ	27500	Thấp
10	10.000đ - 50.000đ	30000	Thấp
11	10.000đ - 60.000đ	35000	Thấp
12	10.000đ - 70.000đ	40000	Thấp
13	12.000đ - 12.000đ	12000	Thấp
14	12.000đ - 15.000đ	13500	Thấp
15	12.000đ - 20.000đ	16000	Thấp
16	12.000đ - 30.000đ	21000	Thấp
17	12.000đ - 65.000đ	38500	Thấp
18	13.000đ - 13.000đ	13000	Thấp
19	15.000đ - 17.000đ	16000	Thấp
20	15.000đ - 199.000đ	107000	Trung bình
21	15.000đ - 20.000đ	17500	Thấp
22	15.000đ - 200.000đ	107500	Trung bình

#### d. Phân tách thành các bảng

Dựa vào hai bảng da\_nang và reviews\_dn, tiến hành phân tách thành các bảng theo như cơ sở dữ liệu đã thiết kế ban đầu.

Tạo bảng StoreDistrict

```
CREATE PROCEDURE SplitDistrictTable
```

Tạo bảng PriceRangeCategory

```
CREATE PROCEDURE SplitPriceRangeCategory
```

Tạo bảng DanangStore

```
CREATE PROCEDURE CreateDanangStores
```

Tạo bảng StoreReviews

```
CREATE PROCEDURE CreateStoreReviews
```

Tạo bảng StoreRating

```
CREATE PROCEDURE CreateStoreRating
```

- Bảng quận

Bảng này gồm các quận tại Đà Nẵng, trích ra thì cột district của bảng da\_nang.

```

---a. Tao bang DistrictTable
CREATE PROCEDURE SplitDistrictTable
AS
BEGIN
    ---B1. Tao bang moi StoreDistrict
    CREATE TABLE StoreDistrict (
        DistrictID INT PRIMARY KEY IDENTITY(1,1), --Tu dong tang
        DistrictName NVARCHAR(100) NOT NULL
    );

    ---B2. Chen du lieu vao bang StoreDistrict
    INSERT INTO StoreDistrict (DistrictName)
    SELECT DISTINCT district
    FROM da_nang;
END;
GO
EXEC SplitDistrictTable

```

	district_id	district_name
1	1	Quận Cẩm Lệ
2	2	Quận Hải Châu
3	3	Quận Liên Chiểu
4	4	Quận Ngũ Hành Sơn
5	5	Quận Sơn Trà
6	6	Quận Thanh Khê

- Bảng loại giá

Bảng này định nghĩa riêng các loại khoảng giá đã được phân loại trong cột price\_category.

```

---b. Tao bang PriceRangeCategory
CREATE PROCEDURE SplitPriceRangeCategory
AS
BEGIN
    ---B1. Tao bang moi PriceRangeCategory
    CREATE TABLE PriceRangeCategory (
        PriceRangeID INT PRIMARY KEY IDENTITY(1,1), -- Tự động tăng
        PriceRangeName NVARCHAR(50) NOT NULL,
        PriceRangeDetails NVARCHAR(100) NOT NULL
    );

    ---B2. Insert du lieu vao bang PriceRangeCategory
    INSERT INTO PriceRangeCategory (PriceRangeName, PriceRangeDetails)
    VALUES
        ('Thấp', 'Dưới 50.000 đồng'),
        ('Trung bình thấp', 'Từ 50.000 đến 100.000 đồng'),
        ('Trung bình', 'Từ 100.000 đến 200.000 đồng'),
        ('Trung bình cao', 'Từ 200.000 đến 300.000 đồng'),
        ('Cao', 'Từ 300.000 đến 500.000 đồng'),
        ('Rất cao', 'Trên 500.000 đồng');

END;
GO
EXEC SplitPriceRangeCategory

```

- Các bảng DaNangStore, StoreReviews, StoreRating

Tương tự với ba bảng còn lại:

Bảng DaNangStore gồm id kết nối với bảng quận, id kết nối với bảng loại giá, và các thông tin cơ bản của cửa hàng.

```

---c. Tao bang DanangStores
CREATE PROCEDURE CreateDanangStores
AS
BEGIN
    --B1. Tao bang moi DanangStores
    CREATE TABLE DanangStores (
        StoreID INT PRIMARY KEY IDENTITY(1,1),
        DistrictID INT,
        PriceRangeID INT,
        StoreName NVARCHAR(255) NOT NULL,
        OpeningHour NVARCHAR(100),
        StoreLink NVARCHAR(500),
        ReviewCount INT,
        StoreAddress NVARCHAR (255),
        FOREIGN KEY (DistrictID) REFERENCES StoreDistrict(DistrictID),
        FOREIGN KEY (PriceRangeID) REFERENCES PriceRangeCategory(PriceRangeID)
    );
    --B2. Chen du lieu
    INSERT INTO DanangStores (DistrictID, PriceRangeID, StoreName, OpeningHour, StoreLink, ReviewCount, StoreAddress)
    SELECT
        ds.DistrictID,
        pr.PriceRangeID,
        d.[name] AS StoreName,
        d.opening_hours AS OpeningHour,
        d.link AS StoreLink,
        d.review_count AS ReviewCount,
        d.street_address AS StoreAddress
    FROM da_nang d
    INNER JOIN StoreDistrict ds ON d.district = ds.DistrictName
    INNER JOIN PriceRangeCategory pr ON d.price_category = pr.PriceRangeName;
END;
GO

```

Bảng StoreReviews chứa thông tin của từng lượt đánh giá như người đánh giá, cửa hàng được đánh giá, điểm cho các tiêu chí của quán đó.

```

CREATE PROCEDURE CreateStoreReviews
AS
BEGIN
    CREATE TABLE StoreReviews (
        ReviewID INT PRIMARY KEY IDENTITY(1,1),
        UserName NVARCHAR(255),
        ReviewTime NVARCHAR(100),
        Rating FLOAT,
        PositionRating FLOAT,
        PriceRating FLOAT,
        QualityRating FLOAT,
        ServiceRating FLOAT,
        AtmosphereRating FLOAT,
        StoreID INT,
        FOREIGN KEY (StoreID) REFERENCES DanangStores(StoreID)
    );
    -- Bước 2: Chèn dữ liệu từ reviews_dn vào StoreReviews
    INSERT INTO StoreReviews (UserName, ReviewTime, Rating, PositionRating, PriceRating, QualityRating, ServiceRating, AtmosphereRating, StoreID)
    SELECT
        r.user_name AS UserName,
        r.review_time AS ReviewTime,
        r.rating AS Rating,
        r.position AS PositionRating,
        r.price AS PriceRating,
        r.quality AS QualityRating,
        r.service AS ServiceRating,
        r.atmosphere AS AtmosphereRating,
        s.StoreID
    FROM
        reviews_dn r
    JOIN
        DanangStores s ON r.id_quan = s.StoreID;
END;
GO

```

Bảng StoreRating chứa các điểm trung bình cho các tiêu chí của mỗi quán ăn.

```

CREATE PROCEDURE CreateStoreRating
AS
BEGIN
    -- Bước 1: Tạo bảng StoreRating
    CREATE TABLE StoreRating (
        StoreID INT PRIMARY KEY IDENTITY(1,1),
        StoreName NVARCHAR(255) NOT NULL,
        AvgRating FLOAT,
        AvgPositionRating FLOAT,
        AvgPriceRating FLOAT,
        AvgQualityRating FLOAT,
        AvgServiceRating FLOAT,
        AvgAtmosphereRating FLOAT,
        StoreID INT,
        FOREIGN KEY (StoreID) REFERENCES DanangStores(StoreID)
    );
    -- Bước 2: Chèn dữ liệu từ da_nang vào StoreRating
    INSERT INTO StoreRating (StoreName, AvgRating, AvgPositionRating, AvgPriceRating, AvgQualityRating, AvgServiceRating, AvgAtmosphereRating, StoreID)
    SELECT 
        d.[name] AS StoreName,
        d.avg_rating AS AvgRating,
        d.avg_position AS AvgPositionRating,
        d.avg_price AS AvgPriceRating,
        d.avg_quality AS AvgQualityRating,
        d.avg_service AS AvgServiceRating,
        d.avg_atmosphere AS AvgAtmosphereRating,
        s.StoreID
    FROM
        da_nang d
    JOIN
        DanangStores s ON d.id_quan = s.StoreID;  -- Liên kết id_quan từ da_nang sang StoreID trong DanangStores
END;
GO;

```

## 3.4. Xây dựng cơ chế backup cho dữ liệu (R4)

### 3.4.1. Lý do lựa chọn loại backup

#### 3.4.1.1. Lý do lựa chọn Backup trên Azure

##### a. An toàn – Bảo vệ dữ liệu một cách tối ưu

Azure Backup cung cấp các cơ chế hiện đại để phát hiện và phản ứng với các bất thường trong hệ thống. Điều này giúp bảo vệ dữ liệu khỏi các rủi ro tiềm ẩn như tấn công mạng, mất dữ liệu do lỗi phần cứng hoặc con người, và các sự cố ngoài ý muốn. Các bản sao lưu được mã hóa và lưu trữ trên nền tảng đám mây của Azure, đảm bảo tính toàn vẹn và bảo mật cao.

##### b. Khả năng chia sẻ và quản lý truy cập linh hoạt

Một trong những lợi thế nổi bật của Azure là khả năng xác thực người dùng và quản lý quyền truy cập. Hệ thống này đảm bảo rằng chỉ những người được cấp quyền mới có thể truy cập vào dữ liệu quan trọng. Đồng thời, Azure hỗ trợ tích hợp với các hệ thống xác thực hiện đại như Active Directory, cho phép phân quyền chi tiết dựa trên vai trò (Role-Based Access Control - RBAC). Giúp tăng cường bảo mật và tạo điều kiện thuận lợi cho việc làm việc nhóm và chia sẻ dữ liệu một cách an toàn giữa các thành viên.

##### c. Tự động hóa và dễ dàng quản lý

Azure Backup là một dịch vụ được thiết kế với khả năng tự động hóa cao, giúp tối ưu hóa các quy trình sao lưu và khôi phục dữ liệu. Thay vì phải thực hiện các thao tác thủ công phức

tập, có thể lên lịch backup tự động và giám sát quá trình thông qua giao diện quản lý đơn giản, thân thiện.

### **3.4.1.2. Lý do lựa chọn Backup trên Local**

#### **a. Bảo mật cao**

Sao lưu dữ liệu cục bộ (Local Backup) mang lại lợi thế lớn về bảo mật, vì toàn bộ dữ liệu được lưu trữ tại chỗ trên các thiết bị nội bộ như ổ cứng, máy chủ. Không cần kết nối với các hệ thống bên ngoài, dữ liệu tránh được các nguy cơ bị tấn công mạng, xâm nhập hoặc truy cập trái phép từ bên ngoài.

#### **b. Không phụ thuộc vào kết nối Internet**

Backup trên Local không yêu cầu kết nối Internet, điều này giúp quá trình sao lưu và khôi phục dữ liệu diễn ra ổn định và nhanh chóng, ngay cả khi mạng gặp sự cố hoặc không khả dụng.

### **3.4.1.3. Lý do lựa chọn các hình thức sao lưu (Full Backup, Differential Backup, Transaction Log Backup)**

#### **a. Full Backup (Sao lưu toàn bộ)**

Thời gian thực hiện: Hàng tháng.

Lý do lựa chọn: Full Backup đảm bảo toàn bộ dữ liệu, bao gồm tất cả các bảng, file, và cấu trúc hệ thống, được sao lưu một cách đầy đủ. Tạo ra một bản sao lưu hoàn chỉnh, giúp phục hồi dữ liệu từ đầu trong trường hợp hệ thống gặp sự cố lớn.

#### **b. Differential Backup (Sao lưu khác biệt)**

Thời gian thực hiện: Hàng tuần.

Lý do lựa chọn: sao lưu các thay đổi được thực hiện kể từ lần sao lưu Full Backup gần nhất. Nó tiết kiệm thời gian và dung lượng lưu trữ so với việc thực hiện Full Backup thường xuyên, đồng thời cung cấp dữ liệu đủ cập nhật để đáp ứng nhu cầu khôi phục ở mức chi tiết hơn.

#### **c. Transaction Log Backup (Sao lưu nhật ký giao dịch)**

Thời gian thực hiện: Hàng ngày (đối với dữ liệu có giao dịch thay đổi).

Lý do lựa chọn: Hình thức sao lưu này đặc biệt cần thiết cho các hệ thống cơ sở dữ liệu giao dịch (OLTP), nơi dữ liệu thay đổi liên tục. Transaction Log Backup giúp ghi lại các thay đổi

chi tiết trong nhật ký giao dịch, cho phép khôi phục dữ liệu chính xác đến thời điểm bất kỳ trong ngày.

### **3.4.2. Quy trình quản lý backup**

#### **3.4.2.1. Lưu trữ Đa Vị trí**

##### **a. On-site Backup**

Tất cả các bản backup được lưu trữ trên máy chủ nội bộ để phục hồi nhanh khi xảy ra sự cố. Được sử dụng làm giải pháp chính cho việc khôi phục dữ liệu ngắn hạn.

##### **b. Off-site Backup**

Lưu trữ các bản backup trên nền tảng đám mây **Microsoft Azure** nhằm đảm bảo dữ liệu được bảo vệ trước các rủi ro vật lý như cháy nổ, mất điện, hoặc lỗi thiết bị tại địa điểm chính. Đồng bộ hóa dữ liệu từ hệ thống On-site lên đám mây theo lịch định kỳ.

#### **3.4.2.2. Lịch Backup Chi Tiết**

##### **a. Full Backup (Sao lưu toàn bộ dữ liệu)**

**Mục đích:** Lưu trữ toàn bộ dữ liệu từ cơ sở dữ liệu để tạo cơ sở phục hồi hoàn chỉnh.

Thực hiện:

- Lần đầu tiên: Thực hiện ngay khi thiết lập hệ thống backup.
- Định kỳ: Thực hiện vào Chủ nhật hàng tuần, lúc 4:00 AM.

Vị trí lưu trữ: On-site và Off-site.

##### **b. Differential Backup (Sao lưu khác biệt)**

**Mục đích:** Lưu trữ những thay đổi so với bản Full Backup gần nhất, giúp tiết kiệm dung lượng lưu trữ.

Thực hiện:

- Định kỳ: Thực hiện vào thứ Hai hàng tuần, lúc 1:00 AM.
- Vị trí lưu trữ: On-site.

##### **c. Transaction Log Backup (Sao lưu nhật ký giao dịch)**

**Mục đích:** Lưu trữ các giao dịch phát sinh để phục hồi dữ liệu chi tiết tới từng thời điểm.

Thực hiện:

- Định kỳ: Thực hiện hàng ngày vào 1:00 AM.

Vị trí lưu trữ: On-site.

#### d. Kiểm tra định kỳ

Mục đích: Đảm bảo tính toàn vẹn và khả năng phục hồi của các bản backup.

Thực hiện:

- Định kỳ: Thực hiện vào thứ Hai đầu tiên mỗi tháng, lúc 1:00 AM.

Nội dung kiểm tra:

- Tính khả dụng của các bản backup.
- Đánh giá dung lượng lưu trữ.
- Thực hiện thử phục hồi dữ liệu.

#### e. Quản lý bản backup cũ

Mục đích: Giải phóng dung lượng lưu trữ và đảm bảo duy trì các bản backup gần nhất.

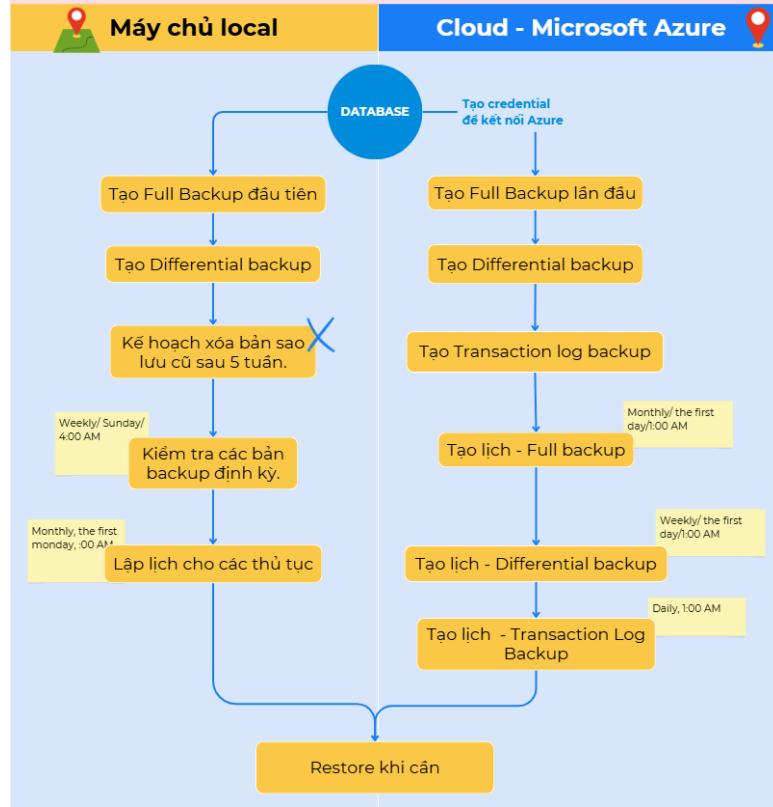
Thực hiện:

- Xóa tự động các bản backup cũ sau 5 tuần.

##### 3.4.2.3. Lập Lịch Tự Động

Toàn bộ quy trình backup được tự động hóa bằng công cụ quản lý tác vụ. Gồm các công việc:

- Full Backup: Chủ nhật hàng tuần, 4:00 AM.
- Differential Backup: Thứ Hai hàng tuần, 1:00 AM.
- Transaction Log Backup: Hàng ngày, 1:00 AM.
- Kiểm tra backup: Thứ Hai đầu tiên hàng tháng, 1:00 AM.
- Xóa backup cũ: Thực hiện tự động theo lịch sau 5 tuần.



### 3.4.3. Xây dựng cơ chế backup cho dữ liệu trên Local

#### 3.4.3.1. Sao lưu đầy đủ đầu tiên

##### a. Cách 1: Dùng T-SQL

Nhóm đặt tên cho thủ tục này là ThucHien\_FullBackup, dùng “BACKUP DATABASE” để tạo một tệp sao lưu của cơ sở dữ liệu Foody\_CrawlData tới đích đến là ổ E, thư mục Backup\_Cr, với tên file cố định là **Foody\_CrawlData\_Full\_** và phần không cố định là thời gian back up.

```

--1. Thực hiện Full Backup đầu tiên
CREATE PROCEDURE ThucHien_FullBackup
AS
BEGIN
    DECLARE @DuongDanBackup NVARCHAR(255);
    SET @DuongDanBackup = 'E:\Backup_Cr\Foody_CrawlData_Full_'
        + REPLACE(CONVERT(VARCHAR, GETDATE(), 120), ':', '')
        + '.bak';
    -- Sao lưu đầy đủ cơ sở dữ liệu
    BEGIN TRY
        BACKUP DATABASE Foody_CrawlData
        TO DISK = @DuongDanBackup
        WITH INIT -- Ghi đè lên file nếu đã tồn tại
        PRINT 'Thực hiện Full Backup thành công';
    END TRY
    BEGIN CATCH
        PRINT 'Lỗi khi thực hiện Full Backup: ' + ERROR_MESSAGE();
    END CATCH
END;
GO
--Gọi
EXEC ThucHien_FullBackup

```

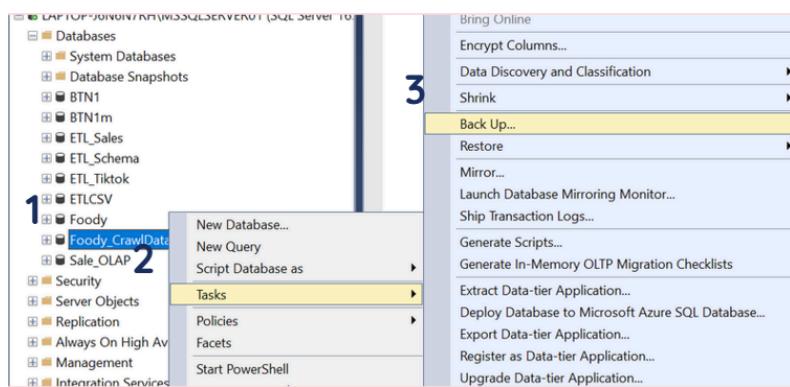
Bỏ dấu ":" trong định dạng giờ để tên tệp hợp lệ.

Lấy thời gian hiện tại làm 1 phần tên tệp (yyyy-mm-dd hh:mm:ss)

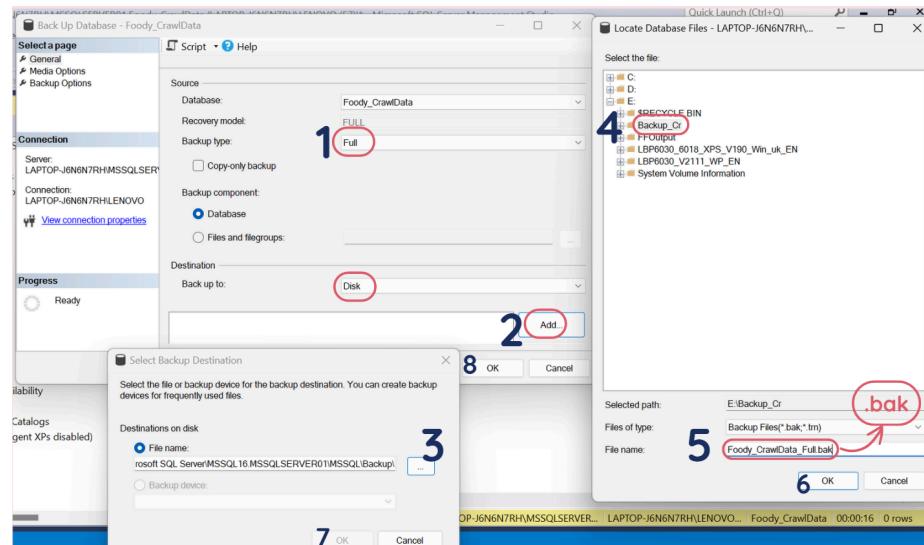
Foody\_CrawlData\_Full\_2024-11-11 194649.bak

## b. Cách 2: Dùng giao diện

Mở giao diện Back up bằng cách nhấn chuột trái vào tên cơ sở dữ liệu Foody\_CrawlData, sau đó lần lượt chọn **Tasks > Back Up...**

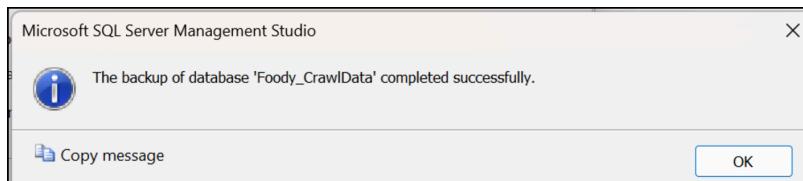


Sau thao tác trên màn hình sẽ hiển thị giao diện như hình dưới:



Ở tab General, chọn loại sao lưu ở lùi thả chọn Backup type, vì đang thực hiện sao lưu đầy đủ nên nhóm chọn “Full”, chọn Backup component là Database, thêm đường dẫn nếu chưa có.

Tên tệp sao lưu dữ liệu cần có đuôi **.bak**, sau đó nhấn ok, trên màn hình sẽ hiển thị kết quả sao lưu thành công, ta có thể vào thư mục tương tự như đường dẫn đã cung cấp để thấy tệp.



### 3.4.3.2. Sao lưu khác biệt

#### a. Cách 1: Dùng T-SQL

Dùng các lệnh tương tự với sao lưu đầy đủ nhưng ở đây nhóm đổi đuôi cố định của tệp sao lưu khác biệt **\_Diff\_**. Đồng thời thêm WITH DIFFERENTIAL để chỉ định loại sao lưu là sao lưu khác biệt.

```

--Thủ tục Differential Backup
CREATE PROCEDURE ThucHien_DifferentialBackup
AS
BEGIN
    DECLARE @DuongDanBackup NVARCHAR(255);
    SET @DuongDanBackup = 'E:\Backup_Cr\Foody_CrawlData_Diff_'
        + REPLACE(CONVERT(VARCHAR, GETDATE(), 120), ':', '')
        + '.bak';

    -- Sao lưu khác biệt
    BEGIN TRY
        BACKUP DATABASE Foody_CrawlData
        TO DTSK = @DuongDanBackup
        WITH DIFFERENTIAL -- Sao lưu chỉ các thay đổi so với bản full backup trước
        PRINT 'Thực hiện Differential Backup thành công';
    END TRY
    BEGIN CATCH
        PRINT 'Lỗi khi thực hiện Differential Backup: ' + ERROR_MESSAGE();
    END CATCH
END;
GO
-- Gọi
EXEC ThucHien_DifferentialBackup

```

Messages

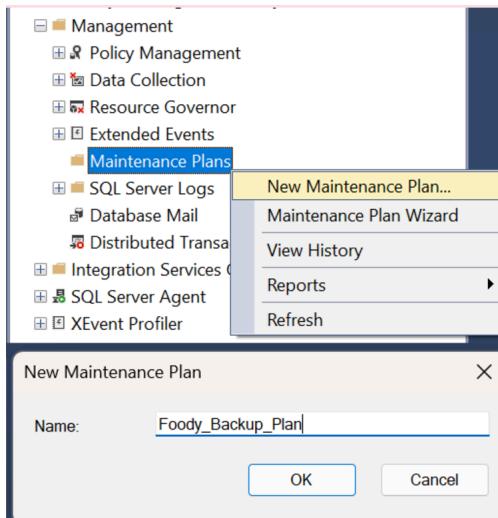
File Foody\_CrawlData\_Diff\_2024-11-11 195521.bak

Processed 56 pages for database 'Foody\_CrawlData', file 'Foody\_CrawlData' on file 1.  
 Processed 2 pages for database 'Foody\_CrawlData', file 'Foody\_CrawlData\_log' on file 1.  
 BACKUP DATABASE WITH DIFFERENTIAL successfully processed 58 pages in 0.012 seconds (37.434 MB/sec).  
 Thực hiện Differential Backup thành công

Completion time: 2024-11-11T19:55:21.7288333+07:00

## b. Cách 2: Dùng giao diện

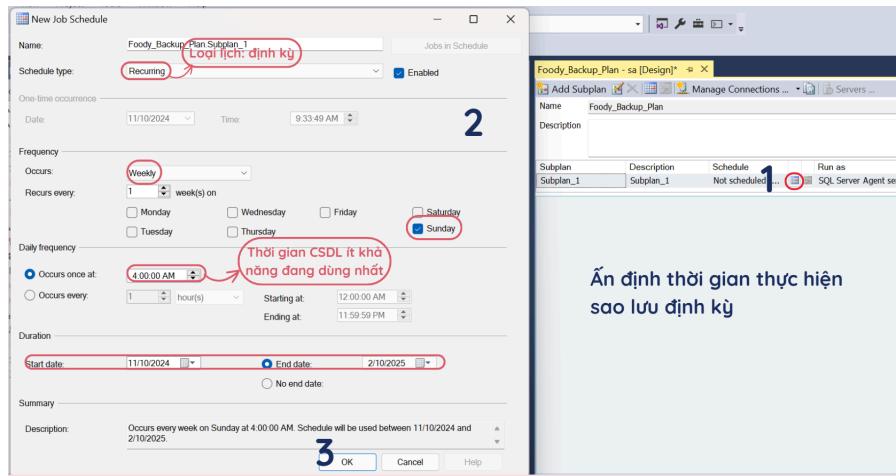
Từ thư mục Management trong SQL, chọn **Management > Maintenance Plans > New Maintenance Plan...**, sau đó đặt tên, ở đây nhóm đặt tên liên quan đến dự án để nhớ.



Nhấp chuột vào ký hiệu lịch (có màu, bên trái), giao diện New Job Schedule sẽ hiện ra.

Chọn loại lịch là Recurring nghĩa là định kỳ (tự động lập lại khi đến thời điểm预定), chọn tần suất mỗi tuần 1 lần vì ít khi có sự thay đổi đối với cơ sở dữ liệu này.

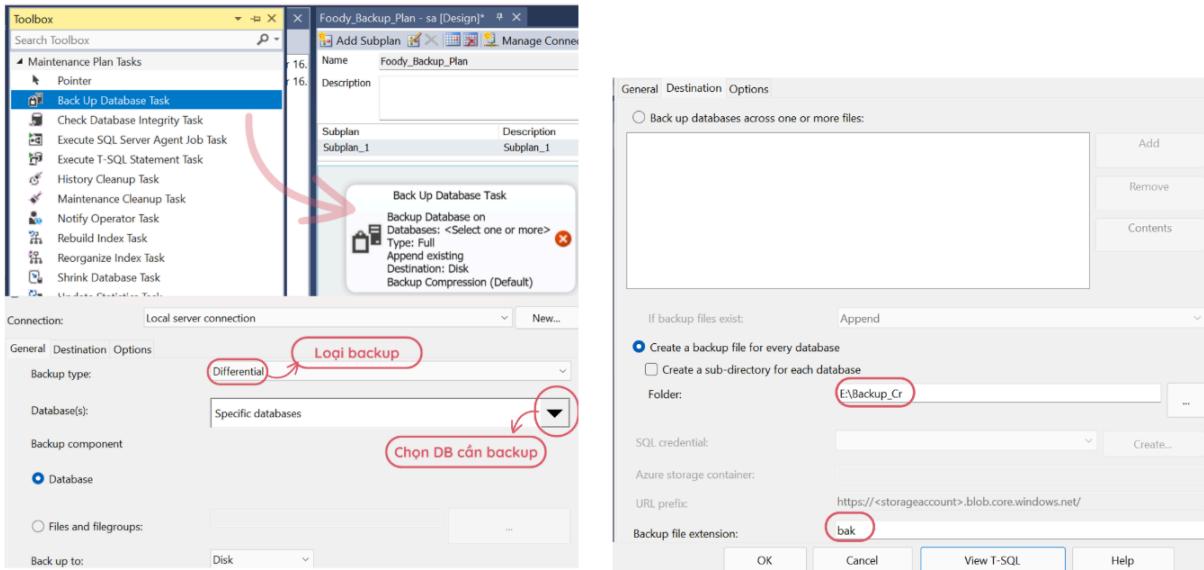
Trong Duration chọn khoảng thời gian lịch này có hiệu lực, vì dự án này chỉ diễn ra trong một kỳ học, sẽ kết thúc vào khoảng cuối tháng 12, không cần thiết tiếp tục sao lưu sau thời điểm này.



Tiếp theo, trong cửa sổ Toolbox, kéo Back Up Database Task thả vào Maintenance Plan đã tạo.

Kích đúp chuột vào khung Back Up Database Task:

- Ở tab General chọn loại sao lưu khác biệt Differential, chọn cơ sở dữ liệu cần sao lưu.
- Ở tab Destination, chọn đường dẫn tới thư mục chứa tệp sao lưu, phần mở rộng điện bak.



### 3.4.3.3. Thiết lập xóa bản sao lưu

#### a. Cách 1: Dùng T-SQL

Nhóm gọi thủ tục này là XoaOld, nhón chọn mốc thời gian xóa từ 5 tuần trước (tức các bản sao lưu cũ hơn 5 tuần kể từ lúc thủ tục này được chạy sẽ bị xóa).

Dùng xp\_cmdshell để đọc các file có tên như trong hình, bỏ vào bảng tạm BackUpFiles.

Sau đó sử dụng con trỏ Cursor để duyệt từng dòng trong bảng tạm trên, cắt lấy 10 ký tự sau \_Diff\_, chính là ngày tháng năm mà tệp đó được sao lưu, và so sánh với biến @NgayXoa đã được tạo từ trước. Sau đây dùng thủ tục có sẵn trong SQL là xp\_delete\_file để xóa tệp sao lưu đó.

```
-- Thủ tục xóa Backup cũ (sao lưu trên 5 tuần sẽ bị xóa)
CREATE PROCEDURE XoaOld
AS
BEGIN
    DECLARE @NgayXoa DATE;
    SET @NgayXoa = CAST(DATEADD(WEEK, -5, GETDATE()) AS DATE);
    DECLARE @FileName NVARCHAR(255);
    DECLARE @FileDate DATE;
    DECLARE @FullPath NVARCHAR(255);

    BEGIN TRY
        -- Tạo bảng tạm chứa danh sách các file sao lưu Differential
        CREATE TABLE #BackupFiles (BackupFile NVARCHAR(255));

        -- Dùng xp_cmdshell để lấy danh sách file trong thư mục backup
        INSERT INTO #BackupFiles
        EXEC xp_cmdshell 'dir /b "E:\Backup_Cr\Foody_CrawlData\Diff*.bak"';

        -- Xóa các file sao lưu Differential cũ (theo ngày trong tên file)
        DECLARE @BackupCursor CURSOR FOR
        -- Con trỏ để duyệt qua danh sách file
        SELECT BackupFile FROM #BackupFiles WHERE BackupFile IS NOT NULL;

        OPEN BackupCursor;
        -- Con trỏ tiến đến dòng đầu và gán tên file ở dòng đó vào @FileName.
        FETCH NEXT FROM BackupCursor INTO @FileName;
        -- Con trỏ tiến đến dòng tiếp theo và gán tên file ở dòng đó vào @FileName.
        FETCH NEXT FROM BackupCursor INTO @FileName;
    END TRY
    BEGIN CATCH
        -- Lấy phần ngày từ tên file
        SET @FileDate = CAST(SUBSTRING(@FileName, CHARINDEX('Diff ', @FileName) + 5, 10) AS DATE);
        -- Nếu có lỗi khi chuyển đổi, bỏ qua file này và tiếp tục với file tiếp theo
        PRINT 'Lỗi khi chuyển đổi ngày từ tên file: ' + @FileName;
        CONTINUE;
    END CATCH
END
```

**Xác định ngày giới hạn để xóa các file**

**Lấy phần ngày (lấy 10 ký tự sau "Diff ")**

**Xác định vị trí bắt đầu của chuỗi "Diff\_" trong FileName.**

**con trỏ tiến đến dòng tiếp theo và gán tên file ở dòng đó vào @FileName.**

**Nếu ngày trong tên file cũ hơn ngày giới hạn, xóa file**

**Điều kiện để xóa file**

**Dường dẫn file cần xóa**

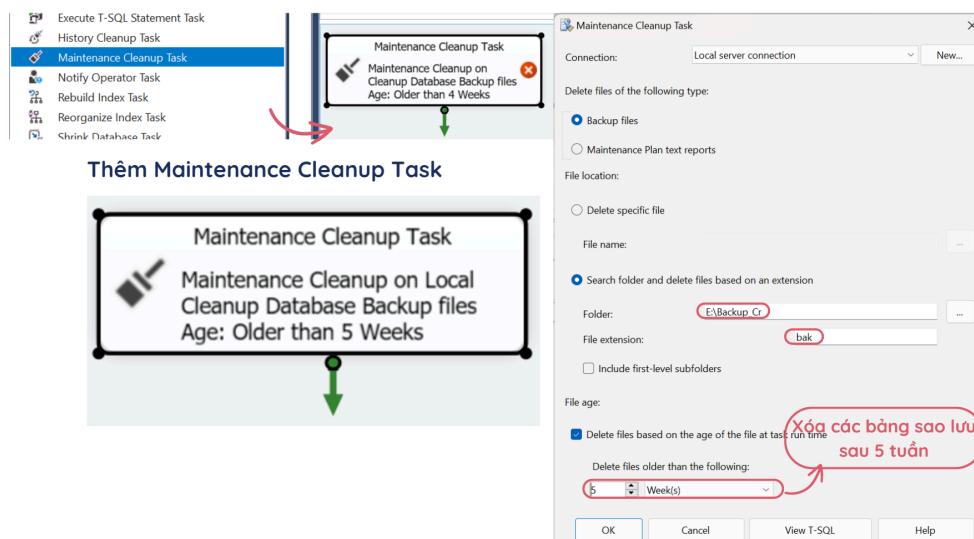
**Xóa file sao lưu Differential cũ**

**Điều kiện để xóa file**

**Xóa các bảng sao lưu sau 5 tuần**

## b. Cách 2: Dùng giao diện

Trong giao diện thì đơn giản hơn vì có sẵn block task cho việc xóa tệp cũ này.



### 3.4.3.4. Kiểm tra các bản sao lưu

### a. Cách 1: Dùng T-SQL

Thủ tục này nhằm kiểm tra tính khả dụng của tệp sao lưu mới nhất bằng cách lấy thông tin của tệp đó từ trong bảng backupset của SQL Server. Dùng lệnh RESTORE VERIFYONLY để kiểm tra tệp sao lưu đó, kết quả nếu tệp hợp lệ sẽ được hiển thị 1 dòng như trong hình ảnh dưới đây.

Nhóm sẽ thực hiện kiểm tra tệp sao lưu đầy đủ trước, sau đó kiểm tra tệp sao lưu khác biệt sau, các lệnh tương tự như với kiểm tra tệp sao lưu đầy đủ.

```
CREATE PROCEDURE KiemTra
AS
BEGIN
    DECLARE @FullBackupFile NVARCHAR(255);
    DECLARE @DifferentialBackupFile NVARCHAR(255);

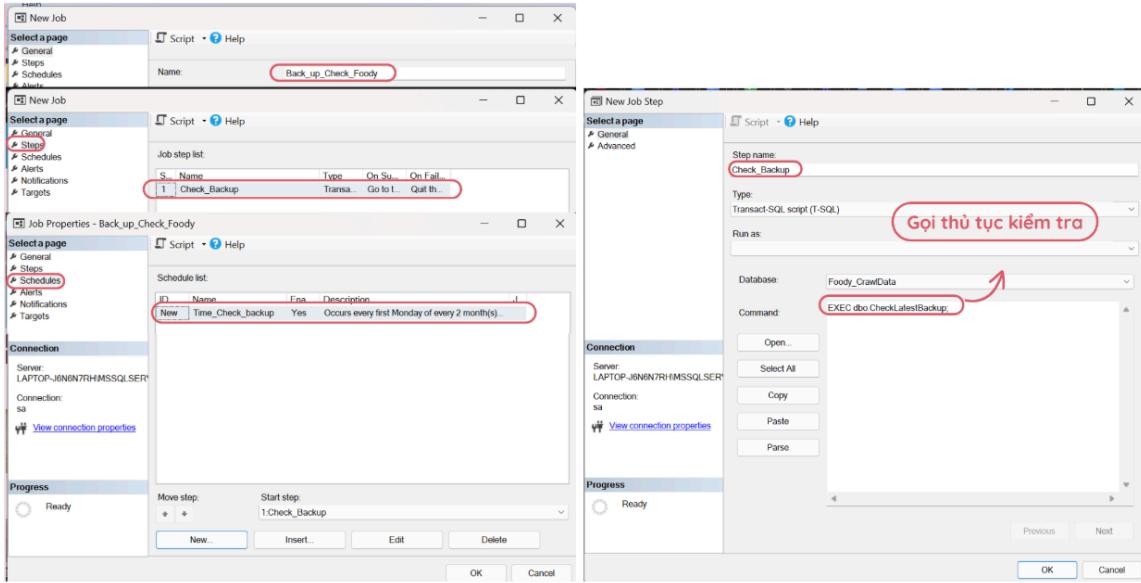
    -- Lấy đường dẫn file Full Backup mới nhất
    SET @FullBackupFile = 'E:\Backup_Cr\Foody_CrawlData_Full_'
    + REPLACE(CONVERT(VARCHAR,
        (SELECT TOP 1 backup_start_date
        FROM msdb.dbo.backupset
        WHERE database_name = 'Foody_CrawlData'
        AND type = 'D' -- 'D' là loại Full Backup
        ORDER BY backup_start_date DESC), 120), ':', '') --bỏ ':' trong time
    + '.bak';

    -- In ra đường dẫn file Full Backup
    PRINT 'Đường dẫn Full Backup: ' + @FullBackupFile;
    -- Kiểm tra file Full Backup
    BEGIN TRY
        RESTORE VERIFYONLY FROM DISK = @FullBackupFile;
        PRINT 'Kiểm tra file Full Backup thành công';
    END TRY
    BEGIN CATCH
        PRINT 'Lỗi khi kiểm tra file Full Backup: ' + ERROR_MESSAGE();
        RETURN;
    END CATCH
```

#### - Cách 2: Dùng giao diện

SQL Server Agent > Jobs > New Job....

Ở tab Steps nhóm thêm bước gọi thủ tục kiểm tra tệp sao lưu, và có thể tạo lịch luôn trong giao diện này ở tab Schedules.



### 3.4.3.5. Lập lịch cho các thủ tục

#### a. Cách 1: Dùng T-SQL

Lần lượt tạo các job, các bước trong job (step), lịch cho job (Schedule) cho các công việc như sao lưu đầy đủ, sao lưu khác biệt, xóa các tệp sao lưu cũ, kiểm tra tệp sao lưu mới nhất sau mỗi lần sao lưu.

```

CREATE PROCEDURE LapLichTT
AS
BEGIN
    -- Lập lịch Differential Backup
    EXEC msdb.dbo.sp_add_job
        @job_name = 'Differential Backup Job';
    EXEC msdb.dbo.sp_add_jobstep
        @job_name = 'Differential Backup Job',
        @step_name = 'Run Differential Backup',
        @subsystem = 'TSQL',
        @command = 'EXEC ThucHien_DifferentialBackup';
    EXEC msdb.dbo.sp_add_jobschedule
        @job_name = 'Differential Backup Job',
        @name = 'Weekly Differential Backup Schedule',
        @freq_type = 8, -- Hàng tuần
        @freq_interval = 1, -- Cứ mỗi 1 tuần
        @freq_recurrence_factor = 1, -- Lặp lại mỗi tuần
        @active_start_time = 030000; -- Thực hiện lúc 03:00 sáng
-- Lập lịch xóa Backup cũ
EXEC msdb.dbo.sp_add_job
    @job_name = 'Cleanup Old Backups Job';
EXEC msdb.dbo.sp_add_jobstep
    @job_name = 'Cleanup Old Backups Job',
    @step_name = 'Run Cleanup Old Backups',
    @subsystem = 'TSQL',
    @command = 'EXEC XoaOld';
EXEC msdb.dbo.sp_add_jobschedule
    @job_name = 'Cleanup Old Backups Job',
    @name = 'Weekly Cleanup Schedule',
    @freq_type = 8, -- Hàng tuần
    @freq_interval = 1, -- Cứ mỗi 1 tuần
    @freq_recurrence_factor = 1, -- Lặp lại mỗi tuần
    @active_start_time = 040000; -- 04:00 sáng

```

Tạo thủ tục để khởi chạy các công việc đã tạo.

```

-- Lập lịch Check Backup hàng tuần
EXEC msdb.dbo.sp_add_job
    @job_name = 'Verify Backup Job';
EXEC msdb.dbo.sp_add_jobstep
    @job_name = 'Verify Backup Job',
    @step_name = 'Run Verify Backup',
    @subsystem = 'TSQL',
    @command = 'EXEC KiemTra';
EXEC msdb.dbo.sp_add_jobschedule
    @job_name = 'Verify Backup Job',
    @name = 'Weekly Verification Schedule',
    @freq_type = 8, -- Hàng tuần
    @freq_interval = 1, -- Cứ mỗi 1 tuần
    @freq_recurrence_factor = 1, -- Lặp lại mỗi tuần
    @active_start_time = 050000; -- 05:00 sáng

PRINT 'Lịch backup tự động đã được thiết lập thành công';
END;
GO
--Gọi
EXEC LapLichTT

```

```

CREATE PROCEDURE StartAllJobs
AS
BEGIN
    -- Bắt đầu Differential Backup Job
    EXEC msdb.dbo.sp_start_job @job_name = 'Differential Backup Job'
    PRINT 'Job Differential Backup Job đã được bắt đầu thành công';

    -- Bắt đầu Cleanup Old Backups Job
    EXEC msdb.dbo.sp_start_job @job_name = 'Cleanup Old Backups Job'
    PRINT 'Job Cleanup Old Backups Job đã được bắt đầu thành công';

    -- Bắt đầu Verify Backup Job
    EXEC msdb.dbo.sp_start_job @job_name = 'Verify Backup Job';
    PRINT 'Job Verify Backup Job đã được bắt đầu thành công';
END;
GO
EXEC StartAllJobs

```

Messages

```

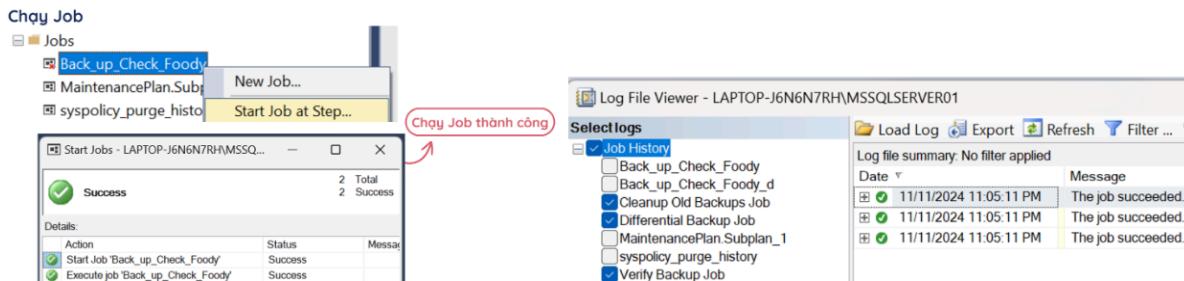
Job 'Differential Backup Job' started successfully.
Job Differential Backup Job đã được bắt đầu thành công.
Job 'Cleanup Old Backups Job' started successfully.
Job Cleanup Old Backups Job đã được bắt đầu thành công.
Job 'Verify Backup Job' started successfully.
Job Verify Backup Job đã được bắt đầu thành công.

```

## b. Cách 2: Dùng giao diện

Kích chuột phải ở tên job đã tạo, chọn **Start Job at Step....**. Kết quả chạy có thành công hay không ta sẽ thấy ở một cửa sổ được tự động bật ra như trong hình.

Đồng thời cũng có thể xem ở Job History để biết công việc đó có chạy thành công theo lịch định kỳ không, ở đây cũng mô tả lỗi nếu không thành công.



### 3.4.4. Xây dựng cơ chế sao lưu (Backup) dữ liệu trên Microsoft Azure

Đầu tiên, cần phải tạo một Storage Account và Blob Container để lưu trữ các dữ liệu cần sao lưu.

Name	Last modified	Anonymous access level	Lease state
Logs	11/9/2024, 12:49:44 AM	Private	Available
foody	11/9/2024, 12:54:11 AM	Private	Available

### Hình: Tạo Storage Account và Blob Container

Sau khi đã tạo được một Blob Container trên Microsoft Azure, ta sẽ tiếp tục tạo một Credential trong SQL Server để có thể kết nối với Azure Storage đã tạo trước đó thông qua SAS - Shared Access Signature (một cơ chế bảo mật trong Azure Storage)

```
-- 1. Tạo CREDENTIAL với SAS Token
CREATE PROCEDURE CreateBlobCredential
AS
BEGIN
    CREATE CREDENTIAL [https://foodysqlbackup.blob.core.windows.net/foody]
    WITH IDENTITY = 'SHARED ACCESS SIGNATURE',
        SECRET = 'sv=2022-11-02&ss=t&t=1667808000&st=1667799600&se=1667814400&sp=r';
END;
EXEC CreateBlobCredential;
```

### Hình: T-SQL tạo Credential

Sau khi đã có thể kết nối từ SQL Server lên Microsoft Azure thì ta sẽ tiến hành sao lưu dữ liệu. Đầu tiên, ta sẽ tạo 3 loại sao lưu lần lượt là Full Backup (saو lưu đầy đủ), Differential Backup (saو lưu khác biệt) và Transaction Log Backup (saو lưu nhật ký giao dịch). Sau khi đã tạo xong 3 loại sao lưu thì ta sẽ tạo lịch tự động cho 3 loại sao lưu trên, trong đó sẽ tạo các Job, các bước trong Job (Step) và lịch cho Job (Schedule) sau đó ta sẽ kết nối các lịch và job lại với nhau. Cuối cùng, khi đã tạo ra các loại sao lưu dữ liệu và các lịch tự động tương ứng thì ta sẽ chạy các job để có thể tự động sao lưu dữ liệu.

Đầu tiên sẽ sao lưu dữ liệu theo loại Full Backup (saو lưu đầy đủ) với chu kỳ 1 tháng 1 lần vào lúc 1 giờ sáng ngày đầu tiên của tháng đó.

```
CREATE PROCEDURE BackupFoodyCrawlData_Full
AS
BEGIN
    -- Lấy ngày giờ hiện tại làm tên tệp động - YYYY-MM-DD HH:MI:SS
    DECLARE @BackupFileName NVARCHAR(MAX) =
        'https://foodysqlbackup.blob.core.windows.net/foody/foody_crawldata_full_backup_'
        + CONVERT(NVARCHAR(20), GETDATE(), 120) + '.bak';

    -- Sao lưu cơ sở dữ liệu - Full Backup
    BACKUP DATABASE [Foody_CrawlData]
    TO URL = @BackupFileName
    WITH
        COMPRESSION; -- Nén dữ liệu sao lưu
END;
-- Gọi
EXEC BackupFoodyCrawlData_Full;
```

### Hình: Tạo liên kết và lưu trữ đối với Full Backup

```

CREATE PROCEDURE dbo.CreateFullBackupJob
AS
BEGIN
    -- Tạo Job
    EXEC msdb.dbo.sp_add_job
        @job_name = N'Full_BackupJob';

    -- Tạo Job Step
    EXEC msdb.dbo.sp_add_jobstep
        @job_name      = N'Full_BackupJob',
        @step_name     = N'Full_BackupStep',
        @subsystem     = N'TSQL',
        @command       = N'EXEC BackupFoodyCrawlData_Full;',
        @database_name = N'Foody_CrawlData';

    -- Tạo lịch
    EXEC msdb.dbo.sp_add_schedule
        @schedule_name      = N'Monthly_FullBackupSchedule',
        @enabled            = 1,           -- kích hoạt lịch
        @freq_type          = 16,          -- Hàng tháng
        @freq_interval      = 1,           -- Chạy vào ngày đầu tiên của mỗi tháng
        @freq_recurrence_factor = 1,        -- Lặp lại hàng tháng
        @active_start_time  = 010000;      -- Bắt đầu lúc 1:00:00 AM (HH:MM:SS)

    -- Thêm Job vào SQL Server
    EXEC msdb.dbo.sp_add_jobserver
        @job_name      = N'Full_BackupJob',
        @server_name   = N'(local)';

    -- Liên kết Job với lịch đã tạo
    EXEC msdb.dbo.sp_attach_schedule
        @job_name      = N'Full_BackupJob',
        @schedule_name = N'Monthly_FullBackupSchedule';

END;
-- Gọi
EXEC dbo.CreateFullBackupJob;

```

Hình: Tạo lịch cho Full Backup

Tiếp theo, sẽ tiến hành tạo một Differential Backup (sao lưu khác biệt) với chu kỳ 1 tuần 1 lần vào lúc 1 giờ sáng ngày đầu tiên của tuần đó.

```

CREATE PROCEDURE BackupFoodyCrawlData_Differential
AS
BEGIN
    -- Lấy ngày giờ hiện tại làm tên tệp động
    DECLARE @BackupFileName NVARCHAR(MAX) =
        'https://foodysqlbackup.blob.core.windows.net/foody/foody_crawldata_diff_backup_'
        + CONVERT(NVARCHAR(20), GETDATE(), 120) + '.bak';

    -- Sao lưu cơ sở dữ liệu - Full Backup
    BACKUP DATABASE [Foody_CrawlData]
    TO URL = @BackupFileName
    WITH
        DIFFERENTIAL, -- Sao lưu các thay đổi kể từ lần sao lưu toàn bộ gần nhất
        COMPRESSION; -- Nén dữ liệu sao lưu
END;
-- Gọi
EXEC BackupFoodyCrawlData_Differential;

```

Hình: Tạo liên kết và lưu trữ đối với Differential Backup

```

CREATE PROCEDURE dbo.CreateDiffBackupJob
AS
BEGIN
    -- Tạo Job
    EXEC msdb.dbo.sp_add_job
        @job_name = N'Diff_BackupJob';

    -- Tạo Job Step
    EXEC msdb.dbo.sp_add_jobstep
        @job_name      = N'Diff_BackupJob',
        @step_name     = N'Diff_BackupStep',
        @subsystem     = N'TSQL',
        @command       = N'EXEC BackupFoodyCrawlData_Differential;',
        @database_name = N'Foody_CrawlData';

    -- Tạo lịch
    EXEC msdb.dbo.sp_add_schedule
        @schedule_name      = N'Weekly_DiffBackupSchedule',
        @enabled            = 1,          -- kích hoạt lịch
        @freq_type          = 8,          -- Hàng tuần
        @freq_interval      = 1,          -- Chạy vào ngày đầu tiên của mỗi tuần
        @freq_recurrence_factor = 1,      -- Lặp lại hàng tuần
        @active_start_time  = 010000;    -- Bắt đầu lúc 1:00:00 AM (HH:MM:SS)

    -- Thêm Job vào SQL Server
    EXEC msdb.dbo.sp_add_jobserver
        @job_name      = N'Diff_BackupJob',
        @server_name   = N'(local)';

    -- Liên kết Job với lịch đã tạo
    EXEC msdb.dbo.sp_attach_schedule
        @job_name      = N'Diff_BackupJob',
        @schedule_name = N'Weekly_DiffBackupSchedule';

END;
-- Gọi
EXEC dbo.CreateDiffBackupJob;

```

*Hình: Tạo lịch cho Differential Backup*

Cuối cùng, ta sẽ tiến hành tạo một Transaction Log Backup (sao lưu nhật ký giao dịch) vào lúc 1 giờ sáng mỗi ngày.

```

CREATE PROCEDURE BackupFoodyCrawlData_TransactionLog
AS
BEGIN
    -- Lấy ngày giờ hiện tại làm tên tệp động
    DECLARE @BackupFileName NVARCHAR(MAX) =
        'https://foodysqlbackup.blob.core.windows.net/foody/foody_crawldata_log_backup_'
        + CONVERT(NVARCHAR(20), GETDATE(), 120) + '.trn';

    -- Sao lưu Transaction Log
    BACKUP LOG [Foody_CrawlData]
        TO URL = @BackupFileName
        WITH
            COMPRESSION      -- Nén dữ liệu sao lưu
END;
-- Gọi thủ tục
EXEC BackupFoodyCrawlData_TransactionLog;

```

*Hình: Tạo liên kết và lưu trữ đối với Transaction Log Backup*

```

CREATE PROCEDURE dbo.CreateLogBackupJob
AS
BEGIN
    -- Tạo Job cho Transaction Log Backup
    EXEC msdb.dbo.sp_add_job
        @job_name = N'Log_BackupJob';

    -- Tạo Job Step cho Transaction Log Backup
    EXEC msdb.dbo.sp_add_jobstep
        @job_name      = N'Log_BackupJob',
        @step_name     = N'Log_BackupStep',
        @subsystem     = N'TSQL',
        @command       = N'EXEC BackupFoodyCrawlData_TransactionLog',
        @database_name = N'Foody_CrawlData';

    -- Tạo lịch sao lưu hàng ngày (hoặc có thể thay đổi tần suất theo yêu cầu)
    EXEC msdb.dbo.sp_add_schedule
        @schedule_name      = N'Daily_LogBackupSchedule',
        @enabled            = 1,           -- kích hoạt lịch
        @freq_type          = 4,           -- Hàng ngày (Daily)
        @freq_interval      = 1,           -- Chạy mỗi ngày
        @freq_recurrence_factor = 1,      -- Lặp lại hàng ngày
        @active_start_time  = 010000;     -- Bắt đầu lúc 1:00:00 AM (HH:MM:SS)

    -- Thêm Job vào SQL Server
    EXEC msdb.dbo.sp_add_jobserver
        @job_name      = N'Log_BackupJob',
        @server_name   = N'(local)';

    -- Liên kết Job với lịch đã tạo
    EXEC msdb.dbo.sp_attach_schedule
        @job_name      = N'Log_BackupJob',
        @schedule_name = N'Daily_LogBackupSchedule';
END;
-- Gọi thủ tục
EXEC dbo.CreateLogBackupJob;

```

Hình: Tạo lịch cho Transaction Log Backup

Kết quả cuối cùng sau khi đã sao lưu dữ liệu lên Microsoft Azure và chạy thành công các lịch tự động sao lưu

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
foody_crawlData_diff_backup_2024_11_09.bak	11/10/2024, 11:36:38 AM	Hot (Inferred)		Block blob	512 KiB	Available
foody_crawlData_full_backup_2024_11_09_020000.bak	11/10/2024, 2:49:31 AM	Hot (Inferred)		Block blob	4.81 MiB	Available
foody_crawlData_log_backup_2024_11_09.trn	11/10/2024, 11:38:42 AM	Hot (Inferred)		Block blob	448 KiB	Available

Hình: Backup dữ liệu lên Microsoft Azure

Date	Step ID	Server	Job Name	Step Name	Notifications	Message
11/12/2024 2:29:15 AM		DESKTOP-OQ9PQ9\SQLSERVER_MTHU	Full_BackupJob			The job succeeded.
11/12/2024 2:29:15 AM		DESKTOP-OQ9PQ9\SQLSERVER_MTHU	Log_BackupJob			The job succeeded.
11/12/2024 2:29:15 AM		DESKTOP-OQ9PQ9\SQLSERVER_MTHU	Diff_BackupJob			The job succeeded.

Hình: Chạy thành công các lịch tự động sao lưu

### 3.5. Phân quyền người dùng trên cơ sở dữ liệu (R5)

#### 3.5.1. Tổng quan về vai trò/quyền được sử dụng

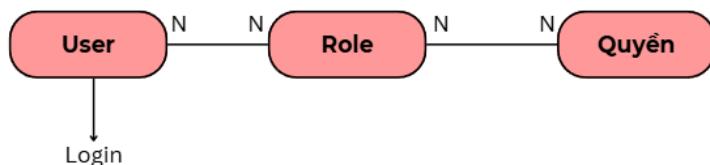
##### 3.5.1.1. Tổng quan

Nhóm thực hiện phân quyền cho 3 vị trí admin (quản trị viên), developer (lập trình viên) và data analyst (chuyên viên phân tích dữ liệu) trong SQL thông qua việc gắn các vai trò và quyền tương ứng cho từng vị trí. Cụ thể:

- **Admin** sẽ được trao vai trò db\_owner, toàn quyền quản trị trên CSDL (Foody\_CrawlData).
- **Developer** là cấp được phép thêm, sửa, xóa, truy vấn dữ liệu trên bảng, thêm, sửa, xóa các thủ tục hoặc hàm, những vai trò và quyền được cấp cho vị trí này là: db\_datareader (đọc dữ liệu), db\_datawriter (ghi dữ liệu: thêm, sửa, xóa), db\_ddladmin (quản lý cấu trúc cơ sở dữ liệu) và quyền thực thi hàm/thủ tục.
- **Data Analyst** chỉ được truy vấn dữ liệu nên chỉ nhận một vai trò trong SQL đó là db\_datareader.

Mối quan hệ giữa người dùng và quyền

- User (Người dùng): Đại diện cho các tài khoản người dùng trong hệ thống.
- Role (Vai trò): Một nhóm quyền được gán cho các người dùng. Một Role có thể được cấp cho nhiều người dùng và sở hữu các quyền cụ thể.
- Quyền (Permission): Các hành động cụ thể mà một Role được phép thực hiện, như truy cập, chỉnh sửa, xóa, hoặc thêm dữ liệu trong hệ thống.



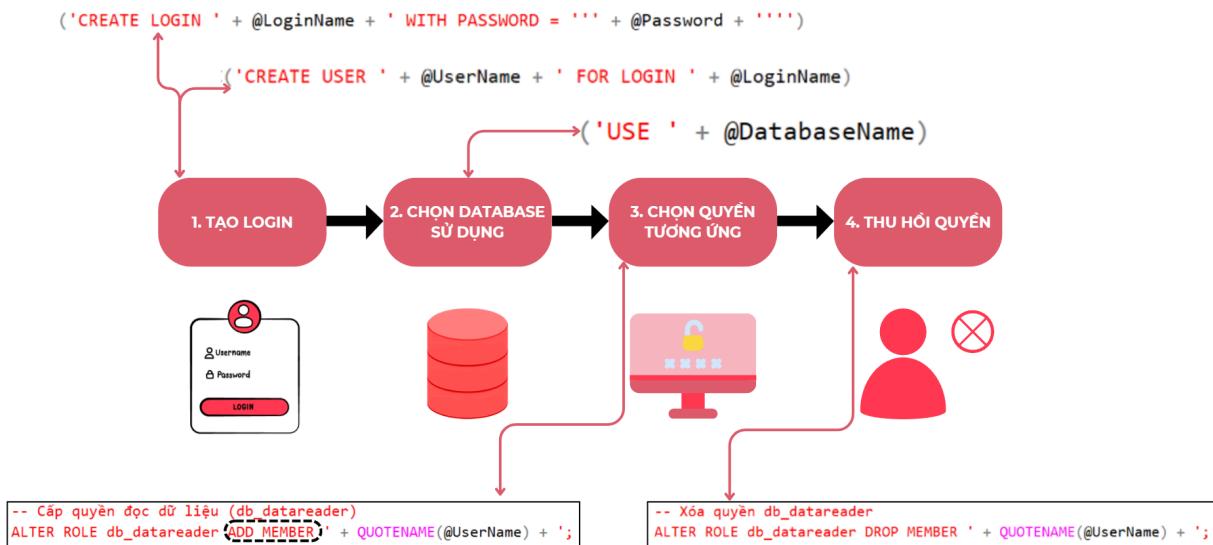
Quan hệ giữa người dùng và vai trò (N:N) tức là một người dùng có thể được gán nhiều vai trò và một vai trò có thể được gán cho nhiều người dùng.

Quan hệ giữa vai trò và quyền (N:N) nghĩa là một vai trò có thể bao gồm nhiều quyền khác nhau và một quyền có thể được gán cho nhiều vai trò.

Login: Kết nối giữa User và hệ thống. Khi người dùng đăng nhập, hệ thống sẽ kiểm tra vai trò (Role) và quyền (Permission) để quyết định những hành động mà người dùng được phép thực hiện.

### 3.5.1.2. Quy trình phân quyền

Hình dưới đây minh họa các bước phân quyền và các lệnh quan trọng được sử dụng trong quá trình phân quyền:



Hình: Quy trình phân quyền

Cụ thể nhóm sẽ tạo những tài khoản đăng nhập khác nhau cho từng vị trí, sau đó để tên những người dùng đó vào bảng vai trò tương ứng, thêm quyền tương ứng cho các tài khoản. Dưới đây là bảng tổng hợp các quyền cũng như vai trò mà mỗi tài khoản/vị trí được phép và không được phép thực hiện:

Vai trò/ Quyền	Quyền	Admin (admin_1)	Developer (dev_1)	Data Analyst (da_1)
db_owner	db_datareader	SELECT (Đọc dữ liệu)	✓	✓
	db_datawriter	INSERT, UPDATE, DELETE	✓	✓
	db_ddladmin	CREATE, ALTER, DROP TABLE	✓	✓
		CREATE, ALTER, DROP VIEW/PROCEDURE/FUNCTION/TRIGGER	✓	✓
		VIEW DEFINITION (Xem định nghĩa đối tượng)	✓	✓
	EXEC	EXECUTE Stored Procedures	✓	✓
		Cấp/Thu hồi quyền người dùng	✓	✗
	BACKUP & RESTORE	Sao lưu và Khôi phục CSDL	✓	✗
		Thay đổi thuộc tính của CSDL	✓	✗

### 3.5.2. Cách phân quyền người dùng trên cơ sở dữ liệu bằng T-SQL

Đầu tiên, nhóm sẽ tạo một tài khoản Login và UserName tương ứng với mỗi loại người dùng bao gồm người dùng là Admin, Developer và Data Analyst. Câu lệnh T-SQL để tạo:

```
-- Tạo Tên Login và UserName
CREATE PROCEDURE CreateLoginAndUser
    @LoginName NVARCHAR(100),
    @Password NVARCHAR(100),
    @DatabaseName NVARCHAR(100),
    @UserName NVARCHAR(100)
AS
BEGIN
    -- Kiểm tra xem tài khoản đăng nhập đã tồn tại chưa
    IF NOT EXISTS (SELECT * FROM sys.server_principals WHERE name = @LoginName)
    BEGIN
        -- Tạo tài khoản đăng nhập mới
        EXEC('CREATE LOGIN ' + @LoginName + ' WITH PASSWORD = ''' + @Password + ''''');
        PRINT N'Tài khoản đăng nhập đã được tạo thành công!';
    END
    ELSE
    BEGIN
        PRINT N'Tài khoản đăng nhập đã tồn tại!';
    END

    -- Sử dụng cơ sở dữ liệu cụ thể
    EXEC('USE ' + @DatabaseName);

    -- Kiểm tra xem người dùng đã tồn tại trong cơ sở dữ liệu chưa
    IF NOT EXISTS (SELECT * FROM sys.database_principals WHERE name = @UserName)
    BEGIN
        -- Tạo người dùng trong cơ sở dữ liệu
        EXEC('CREATE USER ' + @UserName + ' FOR LOGIN ' + @LoginName);
        PRINT N'Người dùng đã được tạo thành công trong cơ sở dữ liệu!';
    END
    ELSE
    BEGIN
        PRINT N'Người dùng đã tồn tại trong cơ sở dữ liệu!';
    END
END;
```

Sau đó sẽ gọi thủ tục cho 3 vị trí, đầu tiên là của người dùng có vị trí là Admin

```
EXEC CreateLoginAndUser
    @LoginName = 'minhthu',
    @Password = 'qtcSDL2024',
    @DatabaseName = 'Foody_CrawlData',
    @UserName = 'admin_1';
```

Tiếp theo là của người dùng có vị trí là Developer

```
EXEC CreateLoginAndUser
    @LoginName = 'developer',
    @Password = 'qtcSDL2024',
    @DatabaseName = 'Foody_CrawlData',
    @UserName = 'dev_1';
```

Và cuối cùng là của người dùng có vị trí là Data Analyst

```
EXEC CreateLoginAndUser
    @LoginName = 'da',
    @Password = 'qtcSDL2024',
    @DatabaseName = 'Foody_CrawlData',
    @UserName = 'da_1';
```

Sau khi đã tạo được tài khoản đăng nhập và UserName thì nhóm sẽ tiến hành phân các vai trò và quyền cho từng vị trí. Đối với người dùng có vị trí Admin thì nhóm đã phân cho người dùng này vai trò db\_owner trong 1 Database là Foody\_CrawlData.

```

-- Sử dụng cơ sở dữ liệu
DECLARE @sql NVARCHAR(MAX);
SET @sql = N'
USE ' + QUOTENAME(@DatabaseName) + ';

-- Cấp quyền db_owner
ALTER ROLE db_owner ADD MEMBER ' + QUOTENAME(@UserName) + ';

PRINT N'Da cap quyen thanh cong cho Admin ' + @UserName + '!';

-- Thực thi lệnh SQL
EXEC sp_executesql @sql;

-- Gọi
EXEC Capquyen_Admin
    @DatabaseName = 'Foody_CrawlData',
    @UserName = 'admin_1';

```

Tiếp tới người dùng có vị trí Developer thì nhóm đã phân cho người dùng này 3 vai trò và 1 quyền như sau:

```

-- Sử dụng cơ sở dữ liệu
DECLARE @sql NVARCHAR(MAX);
SET @sql = N'
USE ' + QUOTENAME(@DatabaseName) + ';

-- Cấp quyền đọc (db_datareader)
ALTER ROLE db_datareader ADD MEMBER ' + QUOTENAME(@UserName) + ';

-- Cấp quyền ghi (db_datawriter)
ALTER ROLE db_datawriter ADD MEMBER ' + QUOTENAME(@UserName) + ';

-- Cấp quyền quản lý schema và các lệnh DDL (db_ddladmin)
ALTER ROLE db_ddladmin ADD MEMBER ' + QUOTENAME(@UserName) + ';

-- Cấp quyền EXECUTE cho người dùng
GRANT EXECUTE TO ' + QUOTENAME(@UserName) + ';

PRINT N'Da cap quyen thanh cong cho Developer ' + @UserName + '!';

-- Thực thi lệnh SQL
EXEC sp_executesql @sql;

-- Gọi
EXEC Capquyen_Dev
    @DatabaseName = 'Foody_CrawlData',
    @UserName = 'dev_1';

```

Cuối cùng, nhóm sẽ cấp vai trò đọc dữ liệu cho người dùng có vị trí là Data Analyst

```

-- Sử dụng cơ sở dữ liệu
DECLARE @sql NVARCHAR(MAX);
SET @sql = N'
    USE ' + QUOTENAME(@DatabaseName) + ';

    -- Cấp quyền đọc dữ liệu (db_datareader)
    ALTER ROLE db_datareader ADD MEMBER ' + QUOTENAME(@UserName) + ';

    PRINT N''Da cap quyen thanh cong cho DA ' + @UserName + '!'';

    -- Thực thi lệnh SQL
    EXEC sp_executesql @sql;

-- Gọi
EXEC Capquyen_DA
    @DatabaseName = 'Foody_CrawlData',
    @UserName = 'da_1';

```

Sau khi đã cấp quyền cho những người dùng ở các vị trí khác nhau, nếu muốn thu hồi các quyền đã cấp khi người dùng đó không còn ở vị trí đó hoặc vì lí do nào khác, ta có thể dùng lệnh như sau:

```

-- Sử dụng cơ sở dữ liệu
DECLARE @sql NVARCHAR(MAX);
SET @sql = N'
    USE ' + QUOTENAME(@DatabaseName) + ';

    -- Xóa quyền db_datareader
    ALTER ROLE db_datareader DROP MEMBER ' + QUOTENAME(@UserName) + ';

    -- Xóa quyền db_datawriter
    ALTER ROLE db_datawriter DROP MEMBER ' + QUOTENAME(@UserName) + ';

    -- Xóa quyền db_ddladmin
    ALTER ROLE db_ddladmin DROP MEMBER ' + QUOTENAME(@UserName) + ';

    -- Xóa quyền EXECUTE
    REVOKE EXECUTE TO ' + QUOTENAME(@UserName) + ';

    PRINT N''Da xoa quyen cho Developer ' + @UserName + '!'';

    -- Thực thi lệnh SQL
    EXEC sp_executesql @sql;

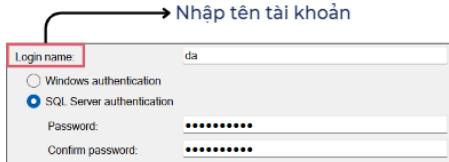
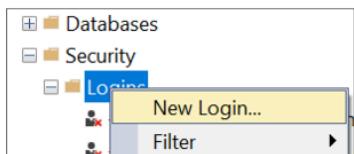
```

### 3.5.3. Cách phân quyền người dùng trên cơ sở dữ liệu bằng giao diện

#### 3.5.3.1. Cấp quyền cho vị trí Data Analyst

## Bước 1: Tạo tài khoản cho Data Analyst (DA)

Mở Object Explorer trong SSMS.



## Bước 2: Chọn Database cho DA



## Bước 3: Cấp quyền db\_datareader cho DA



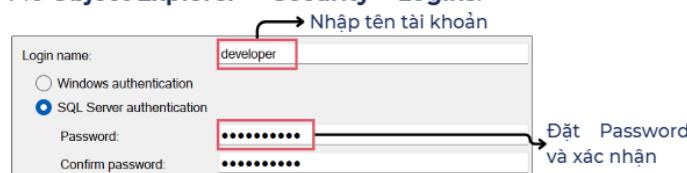
## Bước 4: Kết quả



### 3.5.3.2. Cấp quyền cho vị trí Developer

## Bước 1: Tạo tài khoản cho Developer

Mở Object Explorer -> Security > Logins.

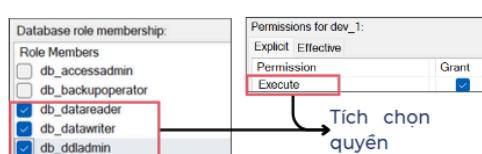


Đặt Password và xác nhận

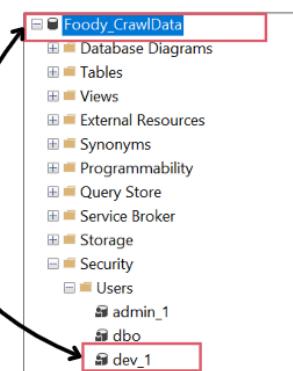
## Bước 2: Chọn Database cho Developer



## Bước 3: Cấp quyền cho Developer



## Bước 4: Kết quả



=> Tài khoản dev đã được cấp quyền trên database.

### 3.5.3.3. Cấp quyền cho vị trí Admin

Hình: Cấp quyền cho vị trí Developer bằng giao diện

## Bước 1: Tạo tài khoản cho Admin

Mở Object Explorer -> Security > Logins.



Nhập tên tài khoản

## Bước 2: Chọn Database cho Admin

Users mapped to this login:		
Map	Database	User
<input checked="" type="checkbox"/>	Foody_CrawlData	admin_1

Database muốn cấp quyền

## Bước 3: Cấp quyền cho Admin

Database role membership:	
Role Members	
<input type="checkbox"/> db_accessadmin	
<input type="checkbox"/> db_backupoperator	
<input type="checkbox"/> db_datareader	
<input type="checkbox"/> db_datawriter	
<input type="checkbox"/> db_ddladmin	
<input type="checkbox"/> db_denydatareader	
<input type="checkbox"/> db_denydatawriter	
<input checked="" type="checkbox"/> db_owner	

Nhấn chuột phải vào **db\_owner** - toàn quyền trên một database

## Bước 4: Kết quả

<input checked="" type="checkbox"/>	Foody_CrawlData
	Database Diagrams
	Tables
	Views
	External Resources
	Synonyms
	Programmability
	Query Store
	Service Broker
	Storage
	Security
	Users
	admin_1

=> Tài khoản admin đã được cấp quyền trên database.



Hình: Cấp quyền cho vị trí Admin bằng giao diện

### 3.5.3.4. Thu hồi quyền bằng giao diện

**Giả sử:** Thu hồi tất cả quyền của tài khoản dev\_1

Bước 1: Mở phân quyền của tài khoản dev\_1

Database User - dev_1	
Select a page	Script ▾ Help
<input type="checkbox"/> General	
<input type="checkbox"/> Owned Schemas	
<input checked="" type="checkbox"/> Membership	
<input type="checkbox"/> Securables	
<input type="checkbox"/> Extended Properties	
Database role membership:	
Role Members	
<input type="checkbox"/> db_accessadmin	
<input type="checkbox"/> db_backupoperator	
<input checked="" type="checkbox"/> db_datareader	
<input checked="" type="checkbox"/> db_datawriter	
<input checked="" type="checkbox"/> db_ddladmin	
<input type="checkbox"/> db_denydatareader	
<input type="checkbox"/> db_denydatawriter	
<input type="checkbox"/> db_owner	
<input type="checkbox"/> db_securityadmin	



Bỏ tick các quyền

Bước 2: Thu hồi quyền ở cấp Database Role

Database User - dev_1	
Select a page	Script ▾ Help
<input type="checkbox"/> General	
<input type="checkbox"/> Owned Schemas	
<input type="checkbox"/> Membership	
<input type="checkbox"/> Securables	
<input type="checkbox"/> Extended Properties	
Database role membership:	
Role Members	
<input type="checkbox"/> db_accessadmin	
<input type="checkbox"/> db_backupoperator	
<input type="checkbox"/> db_datareader	
<input type="checkbox"/> db_datawriter	
<input type="checkbox"/> db_ddladmin	
<input type="checkbox"/> db_denydatareader	
<input type="checkbox"/> db_denydatawriter	
<input type="checkbox"/> db_owner	
<input type="checkbox"/> db_securityadmin	

Hình: Thu hồi quyền bằng giao diện

## 3.6. Trực quan dữ liệu bằng Microsoft Power BI và Python (R6)

Nhóm sử dụng Microsoft Power BI và Python để thực hiện trực quan hóa dữ liệu, giúp làm nổi bật các thông tin quan trọng từ dataset. Các biểu đồ, bảng và số liệu được trình bày nhằm hỗ trợ việc phân tích sâu hơn, phục vụ xây dựng mô hình Machine Learning và đề xuất chiến lược phù hợp.

### **3.6.1. Trực quan dữ liệu bằng Microsoft Power BI**

#### **3.6.1.1. Tổng quan dữ liệu**

##### **a. Tổng quan chính:**

- Số loại hình cửa hàng: 7 loại hình (Quán ăn, Ăn vặt/vỉa hè, Cafe, Nhà hàng, Tiệm bánh, Ăn chay, Quán nhậu)
- Tổng số cửa hàng: 111 cửa hàng.
- Tổng số lượt đánh giá: 3043 lượt.
- Số lượng tài khoản tham gia đánh giá: 1581 tài khoản.
- Thang đánh giá: Từ 1.00 đến 10.00 điểm.

##### **b. Phân bố cửa hàng:**

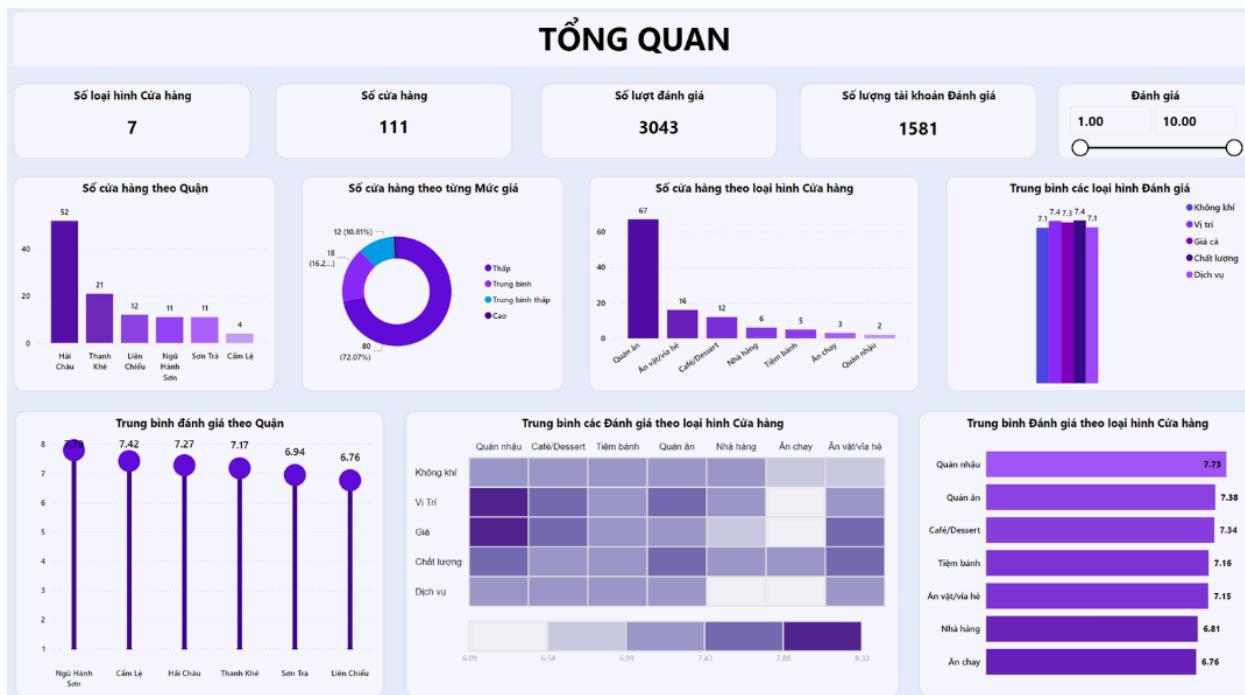
- Theo quận: Hải Châu nhiều nhất với 52 cửa hàng, tiếp theo là Thanh Khê (21 cửa hàng). Liên Chiểu, Ngũ Hành Sơn, Sơn Trà có khoảng 11-12 cửa hàng, còn Cẩm Lệ ít nhất với 4 cửa hàng.
- Theo mức giá: Mức giá thấp chiếm đa số (80 cửa hàng, 72.07%), tiếp theo là trung bình (18 cửa hàng) và trung bình thấp (12 cửa hàng). Có 1 cửa hàng nào ở mức giá cao.
- Theo loại hình: Quán ăn chiếm nhiều nhất (67 cửa hàng), tiếp đến là Ăn vặt/ Vỉa hè (16 cửa hàng) và Cafe/Dessert (12 cửa hàng). Các loại hình khác như tiệm bánh, nhà hàng, ăn chay, ăn vặt/vỉa hè có số lượng khá ít.

##### **c. Trung bình đánh giá:**

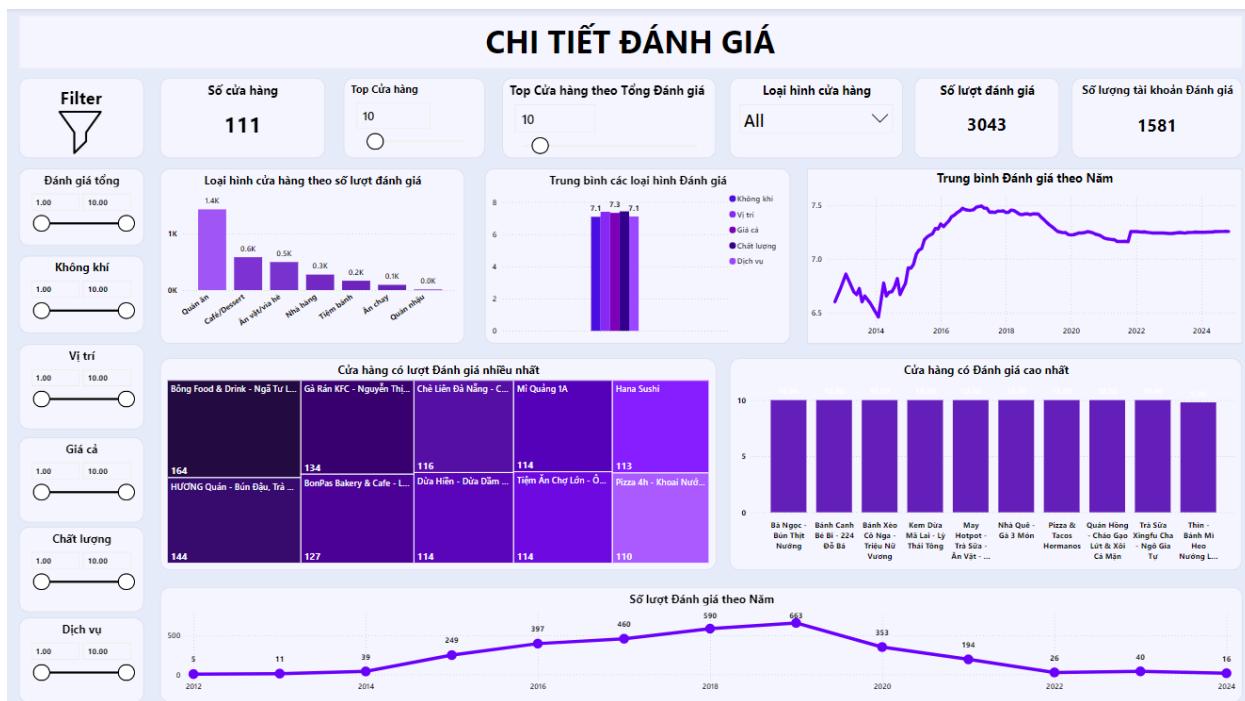
Trung bình đánh giá theo quận cho thấy Ngũ Hành Sơn có điểm cao nhất với 7.79, trong khi Liên Chiểu đạt điểm thấp nhất là 6.76.

Về loại hình cửa hàng, Quán nhậu dẫn đầu với điểm trung bình 7.73, còn Ăn chay có đánh giá thấp nhất là 6.76.

Xét theo các tiêu chí như không khí, vị trí, giá cả, chất lượng và dịch vụ, điểm đánh giá dao động trong khoảng 7.1 đến 7.4, cho thấy mức độ hài lòng chung tương đối ổn định.



Hình: Dashboard Tổng quan

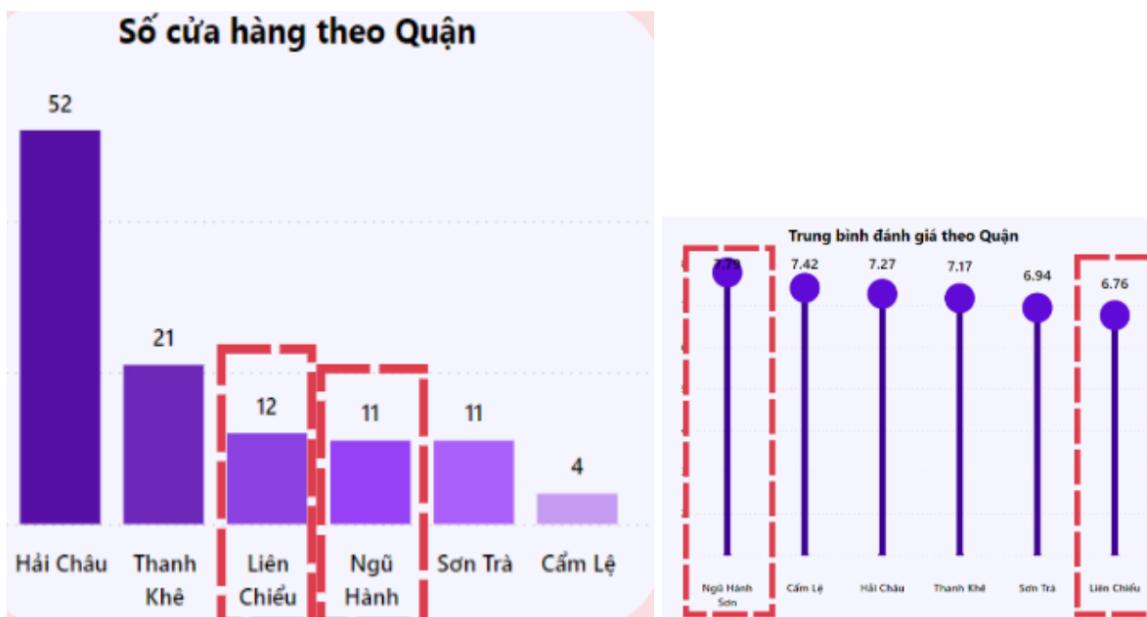


Hình: Dashboard Chi tiết đánh giá

#### 3.6.1.2. Đánh giá về các quận trong thành phố Đà Nẵng

Hai khu vực Ngũ Hành Sơn và Liên Chiểu được đánh giá chi tiết dựa trên số lượng cửa hàng, điểm trung bình đánh giá và các tiêu chí đánh giá cao/thấp.

Tiêu chí/ Loại hình	Ngũ Hành Sơn	Liên Chiểu
Số cửa hàng	Top 4 ( $11/111 = 10\%$ )	Top 3 ( $12/111 = 11\%$ )
Điểm trung bình đánh giá	Cao nhất ( $> 7/10$ )	Thấp nhất ( $< 7/10$ )
Tiêu chí đánh giá cao/thấp	Điểm cao ( $> 8/10$ ) (Vị trí + Chất lượng)	Điểm thấp ( $< 6/10$ ) (Dịch vụ)



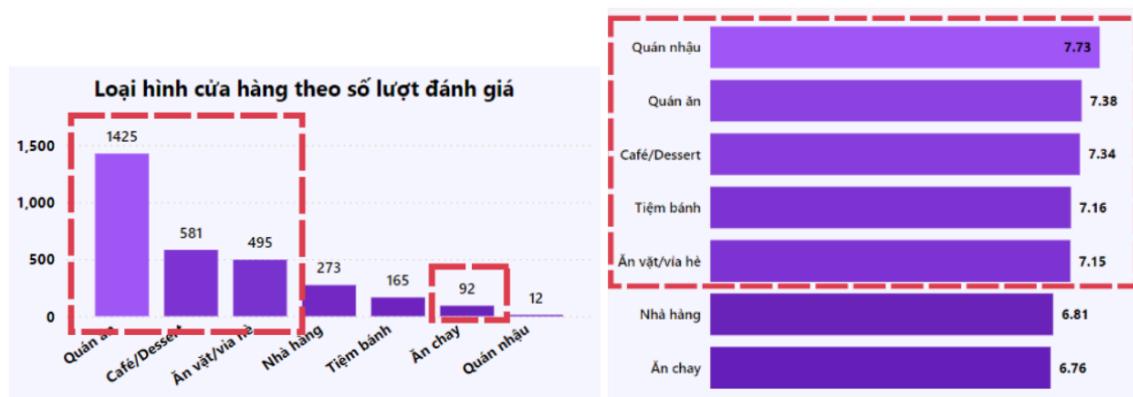
### 3.6.1.3 Đánh giá về 7 loại hình cửa hàng

Hai nhóm cửa hàng được phân loại và phân tích cụ thể gồm:

- Nhóm 1: Quán nhậu, quán ăn, cafe/dessert, tiệm bánh, ăn vặt.
- Nhóm 2: Nhà hàng và các quán ăn chay.

Tiêu chí/ Loại hình	Quán nhậu, quán ăn, Cafe/Dessert, Tiệm bánh, Ăn vặt	Nhà hàng, Ăn chay
------------------------	--	-------------------

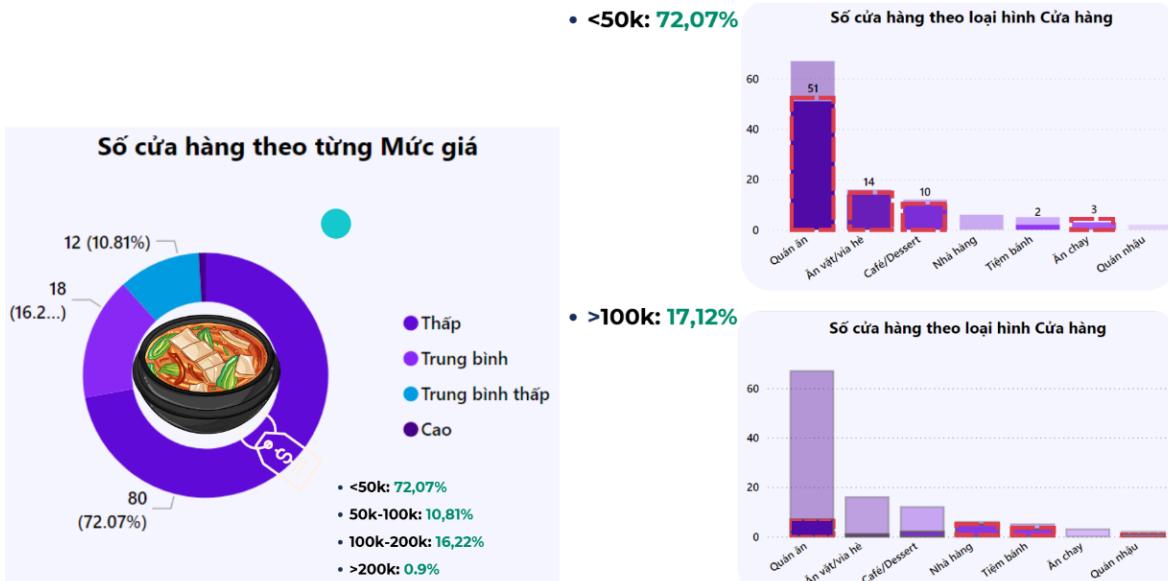
Số lượt đánh giá	> 2.0 K	< 0.5 K
Điểm trung bình	> 7/10 --> cao	< 7/10 --> thấp
Tiêu chí	Điểm cao (>8/10) - Vị trí + Giá	Điểm thấp (<6/10) - Dịch vụ (Vị trí và Giá cả - quán Ăn chay)



#### 3.6.1.4. Mức giá của các cửa hàng

Từ biểu đồ ta thấy hơn 70% các cửa hàng ở Đà Nẵng có mức giá thấp (<50k/1 người). Điều này là quá dễ hiểu bởi nơi đây vốn nổi tiếng là thành phố lớn với chi phí sinh hoạt thấp trong nước.

Mặc dù giá rẻ chiếm phần lớn, vẫn có sự xuất hiện của các phân khúc trung và cao cấp, đáp ứng nhu cầu đa dạng của du khách và người dân địa phương.



Hầu hết cả quán ăn, ăn vặt/vỉa hè, tiệm chay đều nằm trong mức giá thấp.

Gần như là tất cả nhà hàng đều nằm trong mức giá trung-cao. (100% mức giá cao >200k thuộc loại hình nhà hàng), đa số các tiệm bánh và 1/2 các quán nhậu thuộc phân khúc này.



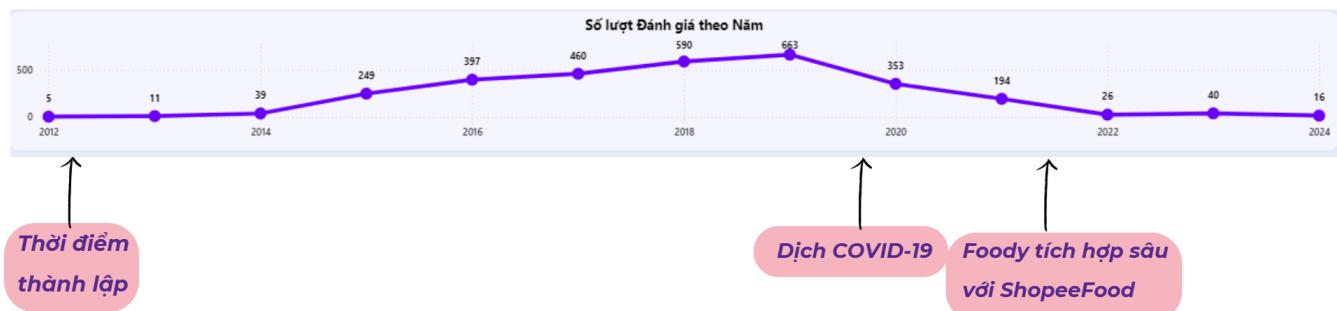
Từ biểu đồ tương quan giữa mức giá và điểm của từng tiêu chí đánh giá của các cửa hàng nhóm thấy được điều đáng lưu ý ở phân khúc giá cao, điểm thấp toàn diện. Liệu có phải vì bỏ ra chi phí cao khách hàng kỳ vọng cao và khắt khe hơn ở những phân khúc khác? Một phần nguyên nhân có thể đến từ “hiệu ứng tiêu cực”, khi chi tiêu ở mức cao, khách hàng thường phóng đại những trải nghiệm không hài lòng.

### 3.6.1.5. Biểu diễn về đánh giá theo thời gian

Vì năm 2012 là thời điểm thành lập Foody, trang web này chưa phổ biến với mọi người, lượng người dùng thấp nên số lượng đánh giá cũng rất thấp, sau đó tăng trưởng đều theo thời gian và đạt cực đại ở năm 2021.

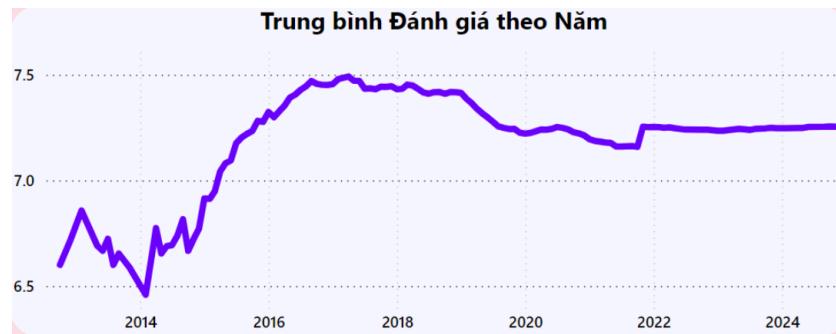
Cuối năm 2019, đại dịch COVID-19 diễn ra khiến số lượng người đến các quán ăn giảm đi, số lượng đánh giá giảm đi đáng kể (lúc bấy giờ Foody vẫn là trang chia sẻ địa điểm với trải nghiệm tại chỗ), cũng trong thời gian này, xu hướng đặt giao đồ ăn nổi lên.

Tháng 8/2021 sau sự kiện chính thức đổi tên, Foody tích hợp sâu với ShopeeFood, các đánh giá được thực hiện trực tiếp trên nền tảng ShopeeFood, website riêng của Foody không còn được tương tác mạnh mẽ như trước nữa, số lượng đánh giá giảm mạnh.



Từ biểu đồ trung bình điểm đánh giá tích lũy qua các năm cho thấy, ở những năm đầu các đánh giá tại Foody có điểm trung bình khá thấp, khoảng 6,5-6,8, biên độ dao động lớn (do số lượng đánh giá ít), sau đây cải thiện rõ rệt từ 2015 đến 2017 cho thấy khả năng nhờ những đánh giá trước đó đã tạo động lực cũng như chỉ ra vấn đề để các cửa hàng cải thiện được những điểm khách hàng cho là chưa tốt.

Từ năm 2022 trở đi điểm đánh giá bão hòa, một phần nhỏ nhờ số lượng đánh giá sau 12 năm tích lũy đủ nhiều để biến động điểm đánh giá trở nên khó phân biệt.

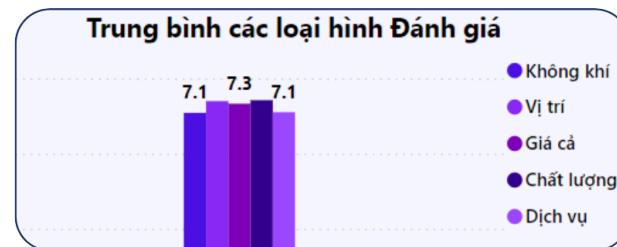


### 3.6.1.6. Đánh giá theo cửa hàng.



*Biểu đồ các cửa hàng có lượt đánh giá cao nhất.*

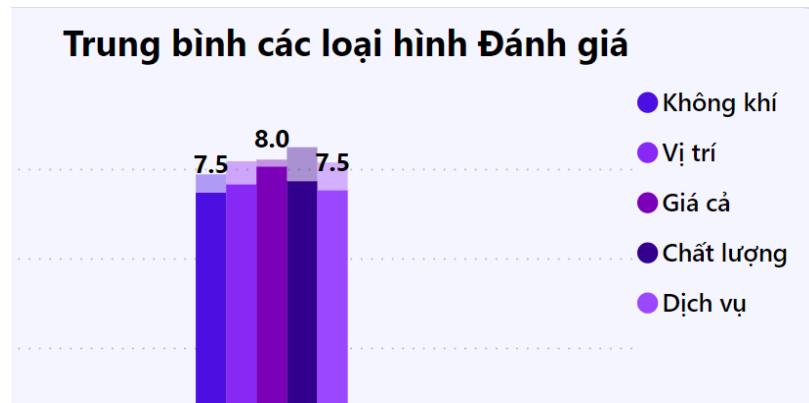
Các cửa hàng nằm trong top những cửa hàng có lượt đánh giá cao thường là Cafe và Quán Ăn, cho thấy việc đây là những loại hình cửa hàng thường được khách hàng nhăm đến khi truy cập Foody.



*Trung bình các loại đánh giá của top 5 cửa hàng có lượt đánh giá cao nhất.*

Bông Food & Drink (chi nhánh Ngã Tư Lê Lợi, Lý Thường Kiệt) là cửa hàng có lượt đánh giá nhiều nhất với 164 lượt đánh giá. Đây là con số nổi bật trong top 5 cửa hàng được đánh

giá nhiều nhất, gồm các cửa hàng như Hương Quán - Bún đậu (144 lượt đánh giá), Gà Rán KFC - Nguyễn Tri Phương (134 lượt), Chè Liên Đà Nẵng (116 lượt). Tổng số lượt đánh giá của 5 cửa hàng này đạt 685 lượt, trong đó có tới 502 lượt đánh giá từ 7 trở lên, cho thấy các cửa hàng này nhận được tín hiệu tích cực từ khách hàng.

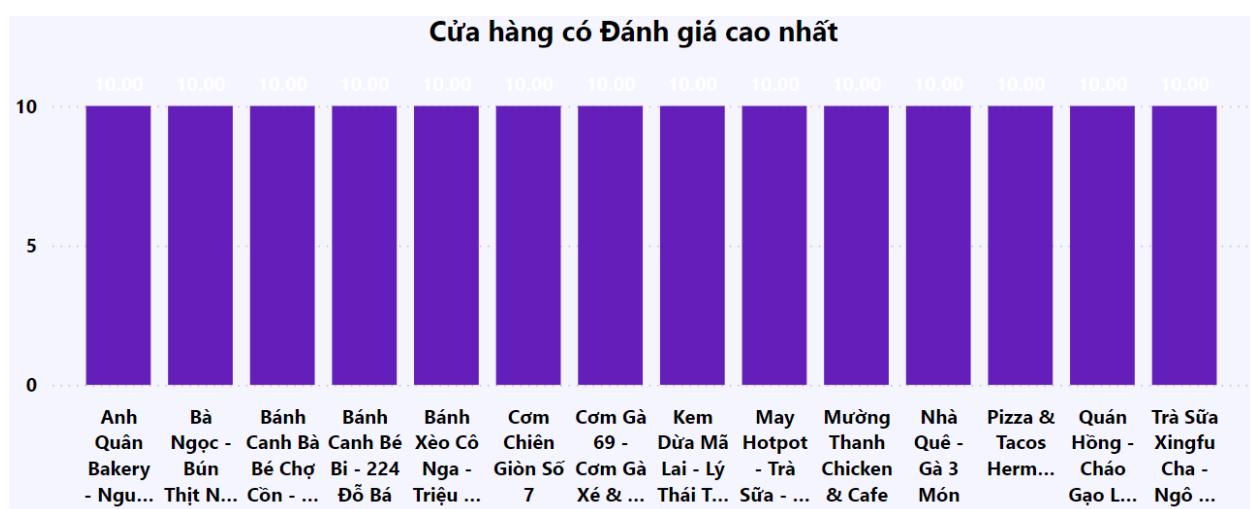


*Trung bình đánh giá cửa hàng Bông Food & Drink.*

Về đánh giá chi tiết dành cho Bông Food & Drink. Cửa hàng này ghi nhận gần 100 đánh giá từ 50 tài khoản có điểm số từ 7 trở lên, minh chứng cho sự ủng hộ của khách hàng.

Các chỉ số đáng chú ý bao gồm Giá cả đạt 8.0 điểm và Chất lượng với 7.7 điểm, cho thấy khách hàng đặc biệt hài lòng với sự hợp lý về giá cả và chất lượng của cửa hàng. Những con số này khẳng định Bông Food & Drink không chỉ thu hút nhiều đánh giá mà còn duy trì chất lượng dịch vụ ổn định, tạo nên sự tin tưởng và yêu thích từ khách hàng.

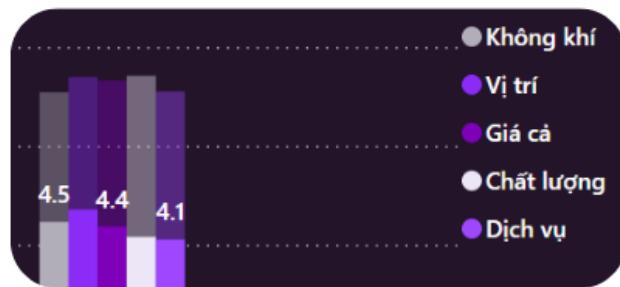
#### 3.6.1.7. Đánh giá cửa hàng theo chỉ số đánh giá.



10 cửa hàng có điểm đánh giá cao nhất đều thuộc loại hình Quán ăn, Cafe, và Nhà hàng.

Các cửa hàng này nhận được đánh giá tích cực ở các tiêu chí Vị trí, Giá cả, Chất lượng và Dịch vụ, với điểm đánh giá trung bình là 10.0. Điều này phản ánh chất lượng phục vụ tốt, vị trí thuận tiện và giá cả hợp lý của các cửa hàng trong nhóm dẫn đầu.

Cửa hàng có đánh giá thấp nhất:



Ngược lại, 10 cửa hàng có điểm đánh giá thấp nhất đều thuộc loại hình Quán ăn. Các tiêu chí bị đánh giá thấp nhất là Dịch vụ, Chất lượng và Giá cả, với điểm trung bình dưới 4.5. Điều này cho thấy các cửa hàng này gặp hạn chế trong việc đáp ứng kỳ vọng của khách hàng về chất lượng dịch vụ và trải nghiệm tổng thể.

Và khu vực Liên Chiểu là nơi có điểm đánh giá trung bình thấp nhất. Điều này có thể do các cửa hàng tại khu vực này chưa chú trọng cải thiện chất lượng dịch vụ, không gian và vị trí, dẫn đến trải nghiệm của khách hàng chưa được tốt như mong đợi.

### 3.6.2. Trực quan bằng Python

#### 3.6.2.1. Thống kê tổng

Đầu tiên, truy vấn những dữ liệu cần thiết hiện bằng T-SQL, sau đó thể hiện những giá trị đó qua html, có thể tùy chỉnh cách trình bày.

Ở đây nhóm trình bày tổng số cửa hàng, số loại hình, tổng lượt đánh giá, số tài khoản đánh giá, cho biết quy mô dữ liệu.

## DashBoard Foody Đà Nẵng

Tổng số cửa hàng: 111

Số loại hình cửa hàng: 7

Tổng số lượt đánh giá: 3043

Số tài khoản đánh giá: 1580

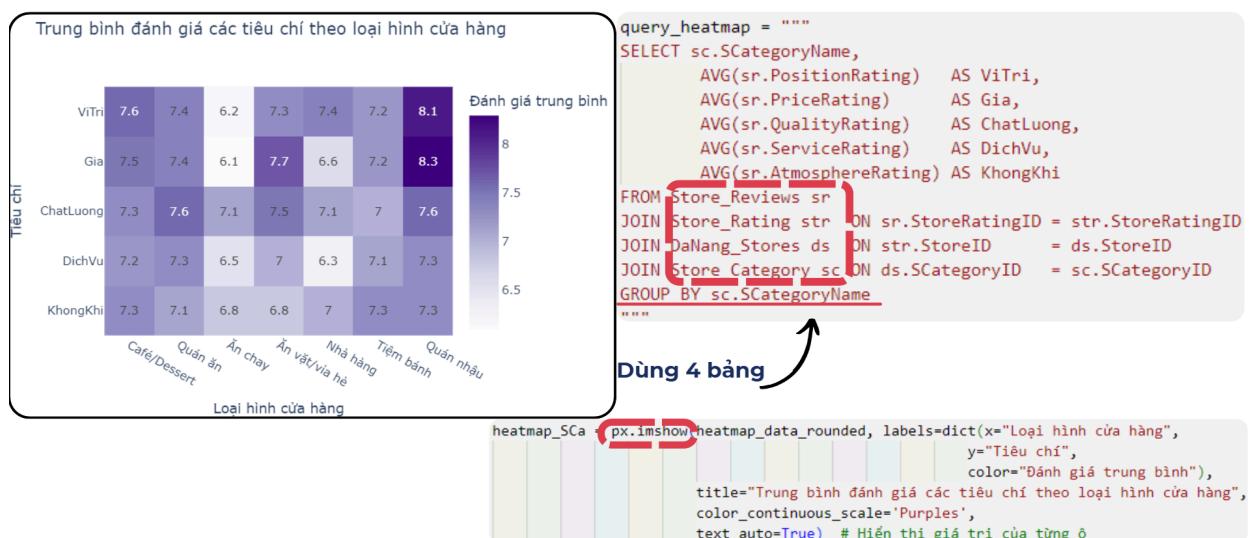
```
query_TQ = """
SELECT
    (SELECT COUNT(DISTINCT StoreID) FROM DaNang_Stores) AS SoCuaHang,
    (SELECT COUNT(DISTINCT SCategoryID) FROM Store_Category) AS SoLoaiCH,
    (SELECT COUNT(DISTINCT ReviewID) FROM Store_Reviews) AS TSDanhGia,
    (SELECT COUNT(DISTINCT UserName) FROM Store_Reviews) AS TongTK
"""

html.H3(f"Tổng số cửa hàng: {summary_data['SoCuaHang']}")
```

### 3.6.2.2. Bản đồ nhiệt trung bình đánh giá các tiêu chí theo loại hình cửa hàng

Để có dữ liệu cho bản đồ nhiệt này, cần truy vấn trung bình các điểm nối với thuộc tính loại hình cửa hàng qua 4 bảng.

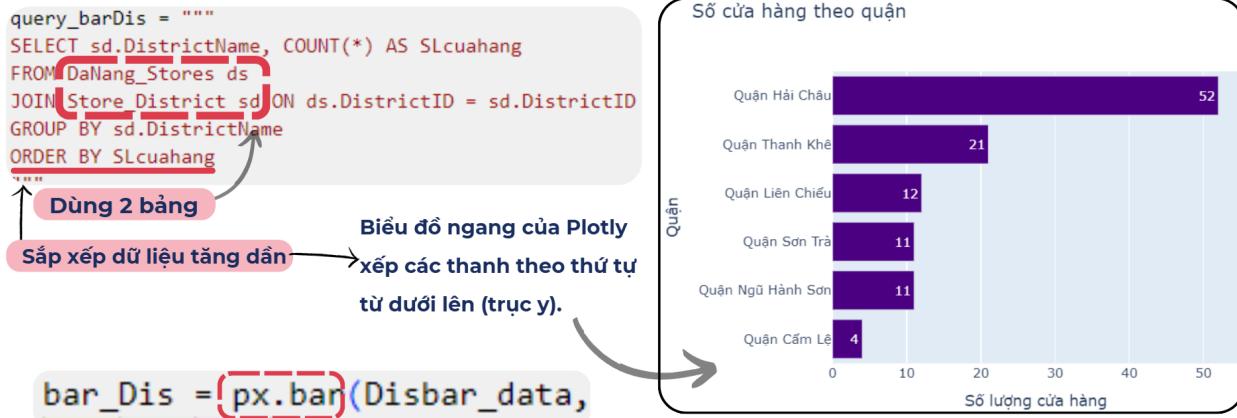
Trong thư viện Plotly Express, px.imshow là hàm dùng để biểu diễn dữ liệu ma trận dưới dạng heatmap (bản đồ nhiệt).



### 3.6.2.3. Số cửa hàng theo quận

Dữ liệu cho biểu đồ này có được từ truy vấn đếm toàn bộ của bảng DaNang\_Stores, nối với bảng Store\_District bằng DistrictID để lấy và gộp nhóm theo cột DistrictName.

Để biểu đồ trông đẹp hơn, nhóm chọn sắp xếp theo thứ tự tăng dần (trong truy vấn), dùng px.bar trong thư viện Plotly để vẽ biểu đồ cột ngang, nó sẽ xếp lần lượt các thanh theo thứ tự từ dưới lên, vì vậy biểu đồ sẽ trông giống như đang sắp xếp giảm dần.



### 3.6.2.4. Số lượt đánh giá theo từng năm

Đối với biểu đồ này thì dữ liệu chỉ cần truy vấn trong 1 bảng Store\_Reviews, dùng CONVERT để đổi kiểu dữ liệu của cột ReviewTime thành DATETIME (định dạng 103: DD/MM/YYYY) trong kết quả truy vấn (dữ liệu trong cơ sở dữ liệu không thay đổi), sau đó đếm gộp nhóm các ReviewID theo năm, sắp xếp dữ liệu theo chiều thời gian.

Dùng px.line trong thư viện Plotly để vẽ biểu đồ đường từ dữ liệu đã truy vấn, ta được biểu đồ như hình.



### 3.6.2.5. Trung bình Đánh giá theo loại hình Cửa hàng

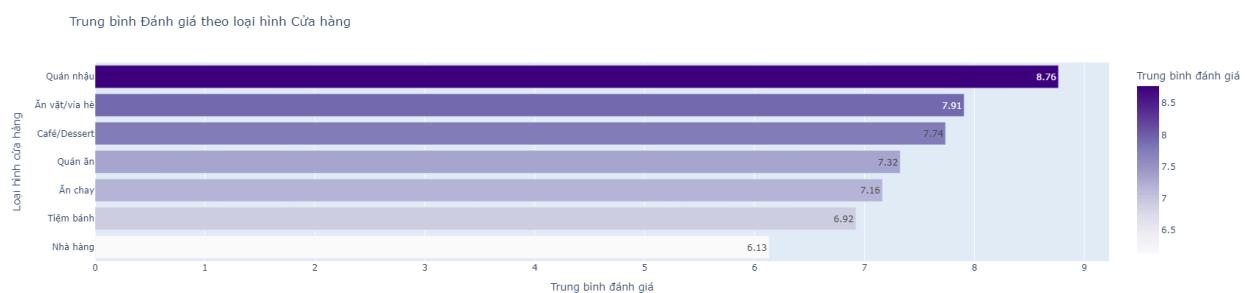
Truy vấn dữ liệu của 3 bảng Store\_Rating, Danang\_Stores, Store\_Category.

```
# Truy vấn SQL
query_avg_rating_by_category = """
SELECT
    SC.SCategoryName AS StoreCategory,
    AVG(SR.AvgRating) AS AvgRating
FROM Store_Rating SR
JOIN DaNang_Stores DS ON SR.StoreID = DS.StoreID
JOIN Store_Category SC ON DS.SCategoryID = SC.SCategoryID
GROUP BY SC.SCategoryName
ORDER BY AvgRating DESC;
"""
```

Dùng px.bar để vẽ biểu đồ:

```
# Tạo biểu đồ thanh ngang
bar_category = px.bar(data_avg_rating,
                       x='AvgRating', # Trung bình đánh giá trên trục x
                       y='StoreCategory', # Loại hình cửa hàng trên trục y
                       title='Trung bình Đánh giá theo loại hình Cửa hàng',
                       labels={'StoreCategory': 'Loại hình cửa hàng',
                               'AvgRating': 'Trung bình đánh giá'},
                       text=data_avg_rating['AvgRating'].round(2), # Hiển thị giá trị trung bình đánh giá
                       color='AvgRating', # Màu sắc thay đổi theo giá trị
                       color_continuous_scale='purples') # Sử dụng thang màu tím
```

Biểu đồ:



### 3.6.2.6. Số cửa hàng theo loại hình Cửa hàng

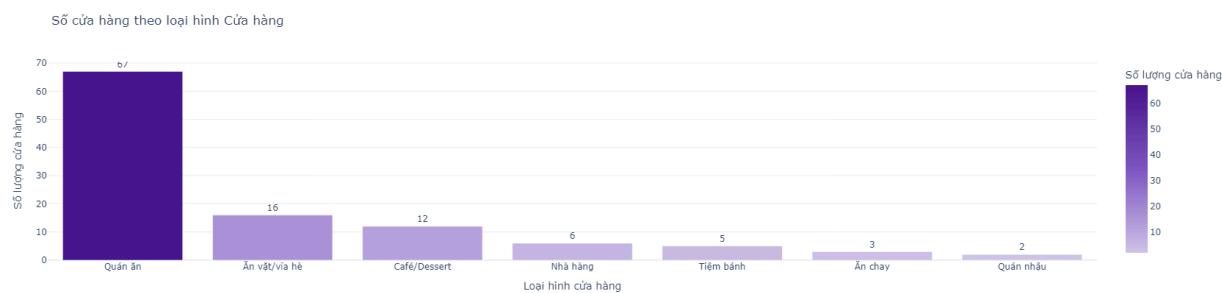
Truy vấn dữ liệu từ Danang\_Stores và Store\_Category.

```
# Truy vấn SQL: Đếm số cửa hàng theo loại hình cửa hàng
query_store_count_by_category = """
SELECT
    SC.SCategoryName AS StoreCategory,
    COUNT(DS.StoreID) AS StoreCount
FROM DaNang_Stores DS
JOIN Store_Category SC ON DS.SCategoryID = SC.SCategoryID
GROUP BY SC.SCategoryName
ORDER BY StoreCount DESC;
"""
```

Dùng px.bar để vẽ biểu đồ:

```
# Tạo biểu đồ cột dọc cho số cửa hàng theo loại hình cửa hàng
bar_store_count = px.bar(data_store_count,
                           x='StoreCategory',
                           y='StoreCount',
                           title='Số cửa hàng theo loại hình Cửa hàng',
                           labels={'StoreCategory': 'Loại hình cửa hàng', 'StoreCount': 'Số lượng cửa hàng'},
                           text=data_store_count['StoreCount'], # Hiển thị số lượng trên cột
                           color='StoreCount', # Màu cột theo giá trị
                           color_continuous_scale=[(0, "#D1C4E9"), (0.5, "#7E57C2"), (1, "#4A148C")])
```

Biểu đồ:



### 3.6.2.7. Số cửa hàng theo từng Mức giá

Truy vấn dữ liệu từ Bảng Danang\_Stores và PriceRange\_Category.

```
# Truy vấn SQL: Đếm số cửa hàng theo từng mức giá
query_store_count_by_price = """
SELECT
    PRC.PriceRangeName AS PriceRange,
    COUNT(DS.StoreID) AS StoreCount
FROM DaNang_Stores DS
JOIN PriceRange_Category PRC ON DS.PriceRangeID = PRC.PriceRangeID
GROUP BY PRC.PriceRangeName
ORDER BY StoreCount DESC;
"""
```

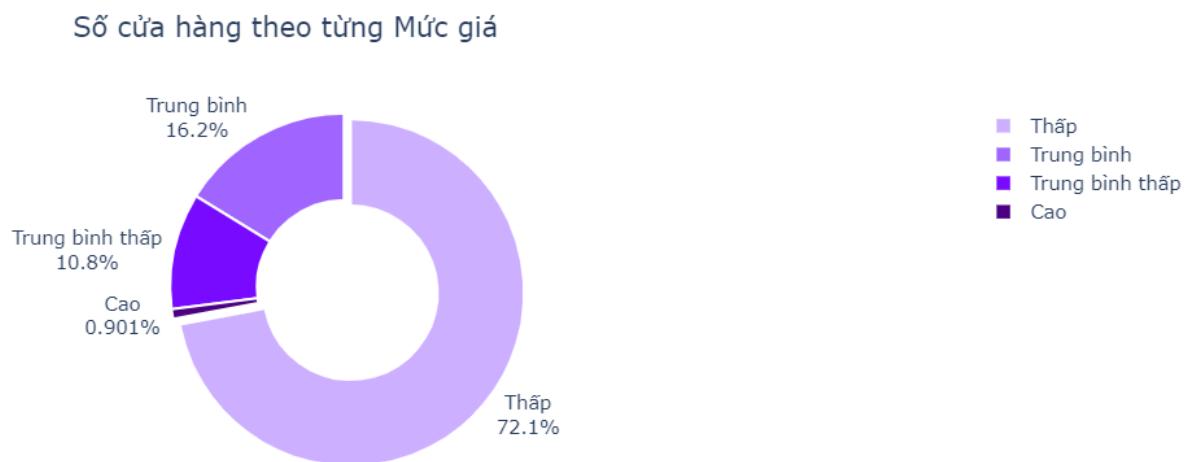
Sử dụng px.pie để vẽ biểu đồ:

```

# Tạo biểu đồ Donut
donut_chart = px.pie(
    data_store_by_price,
    names='PriceRange',
    values='StoreCount',
    title='Số cửa hàng theo từng Mức giá',
    hole=0.5, # Tạo phần rỗng ở giữa
    color_discrete_sequence=purple_shades # Sử dụng dải màu tím
)

```

Biểu đồ:



### 3.6.2.8. Top N cửa hàng có số lượng đánh giá nhiều nhất

Đầu tiên, sẽ truy vấn dữ liệu topN cửa hàng có số lượng đánh giá nhiều nhất với câu lệnh SQL từ dữ liệu đã có

```

# Truy vấn dữ liệu top N cửa hàng có lượt đánh giá nhiều nhất
query = f"""
SELECT TOP {N} StoreName, ReviewCount
FROM DaNang_Stores
ORDER BY ReviewCount DESC
"""

```

Sau đó, sẽ thực hiện vẽ biểu đồ TreeMap để hiển thị topN cửa hàng có số lượng đánh giá nhiều nhất với thư viện squarify

```

# Vẽ biểu đồ treemap
plt.figure(figsize=(12, 8))
squarify.plot(
    sizes=df["ReviewCount"],
    label=wrapped_labels, # Áp dụng văn bản đã được chia dòng
    alpha=1.0,
    color=colors, # Áp dụng màu sắc từ bảng màu tím
    pad=True # Điều chỉnh padding giữa văn bản và các cạnh của ô
)
plt.title(f"Biểu đồ Treemap: Top {N} cửa hàng có số lượng đánh giá nhiều nhất")
plt.axis("off")
plt.show()

```

Tiếp theo, sẽ tạo một Slicer để lọc biểu đồ theo topN mong muốn với mặc định là top 10

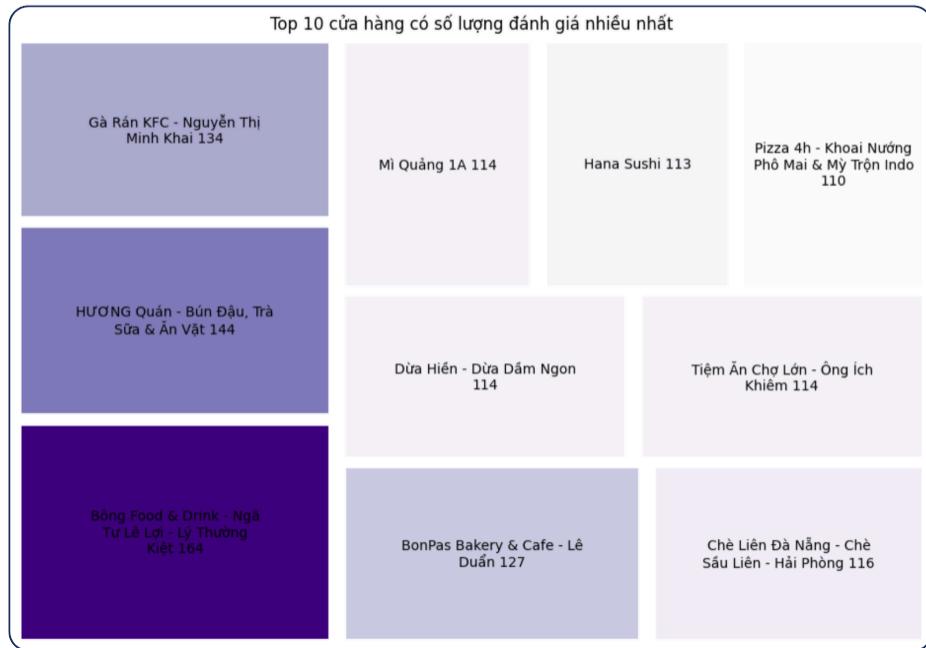
```

# Tạo slicer (widget) để chọn số lượng cửa hàng top N
top_n_slider = widgets.IntSlider(
    value=10,
    min=1,
    max=111,
    step=1,
    description="Top N:",
    continuous_update=False)
# Kết nối slicer với hàm vẽ biểu đồ
widgets.interactive(plot_top_stores, N=top_n_slider)

```

Kết quả biểu đồ sau khi chạy:

Top N:  10



### 3.6.2.9. Top N cửa hàng có điểm đánh giá cao nhất

Đầu tiên, sẽ truy vấn dữ liệu topN cửa hàng có điểm đánh giá trung bình cao nhất với câu lệnh SQL từ dữ liệu đã có

```
# Truy vấn Top N cửa hàng có điểm đánh giá cao nhất
query = f"""
SELECT TOP {n} S.StoreName, R.AvgRating
FROM Store_Rating R
JOIN DaNang_Stores S ON R.StoreID = S.StoreID
ORDER BY R.AvgRating DESC
"""
```

Sau đó, sẽ thực hiện vẽ biểu đồ cột để hiển thị topN cửa hàng có điểm đánh giá trung bình cao nhất

```

# Vẽ biểu đồ
plt.title(f"Top {n} cửa hàng có điểm đánh giá cao nhất", fontsize=16)
plt.xlabel("Tên cửa hàng", fontsize=14)
plt.ylabel("Điểm đánh giá trung bình", fontsize=14)
plt.xticks(rotation=45, ha="right") # Xoay nhãn trục X để dễ đọc hơn
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.tight_layout() # Tự động điều chỉnh để không bị tràn chữ
plt.show()

```

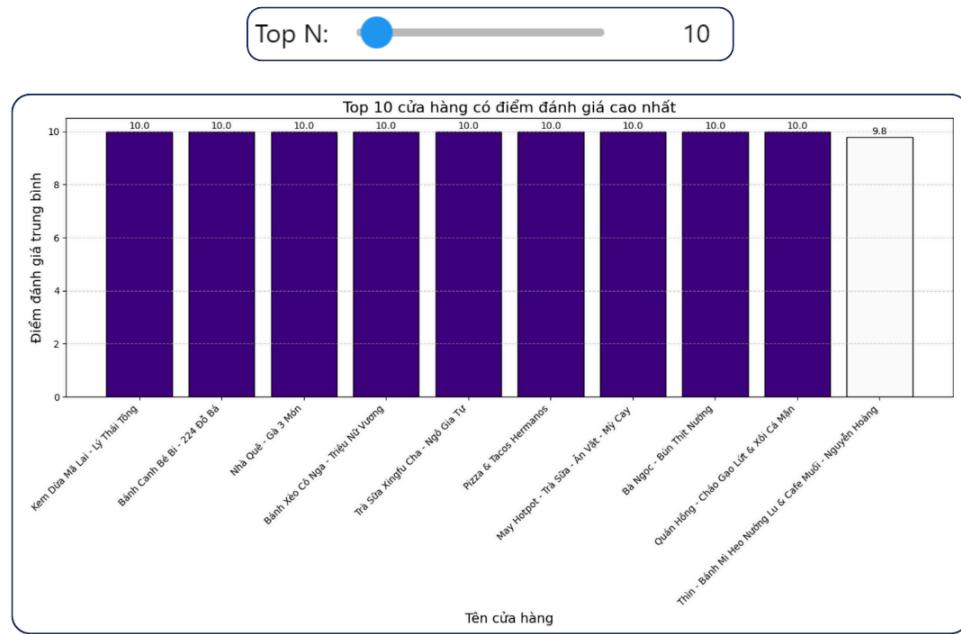
Tiếp theo, sẽ tạo một Slicer để lọc biểu đồ theo topN mong muốn với mặc định là top 10

```

# Tạo slider để chọn giá trị N
n_slider = widgets.IntSlider(
    value=10, # Giá trị mặc định
    min=1, # Giá trị tối thiểu
    max=111, # Giá trị tối đa
    step=1, # Bước nhảy
    description="Top N:",
    continuous_update=False
)
# Sử dụng widgets.interactive để kết nối slider với hàm
interactive_plot = widgets.interactive(plot_top_n_stores, n=n_slider)

```

Kết quả biểu đồ sau khi chạy:



### 3.6.2.10. Trung bình các loại hình đánh giá

Đầu tiên, sẽ truy vấn dữ liệu trung bình của 5 loại hình đánh giá dựa vào bảng Store\_Reviews từ với câu lệnh SQL từ dữ liệu đã có

```

# Truy vấn dữ liệu trung bình của 5 loại hình đánh giá từ bảng Store_Reviews
query = """
SELECT
    AVG(PositionRating) AS AvgPositionRating,
    AVG(PriceRating) AS AvgPriceRating,
    AVG(QualityRating) AS AvgQualityRating,
    AVG(ServiceRating) AS AvgServiceRating,
    AVG(AtmosphereRating) AS AvgAtmosphereRating
FROM Store_Reviews
"""

```

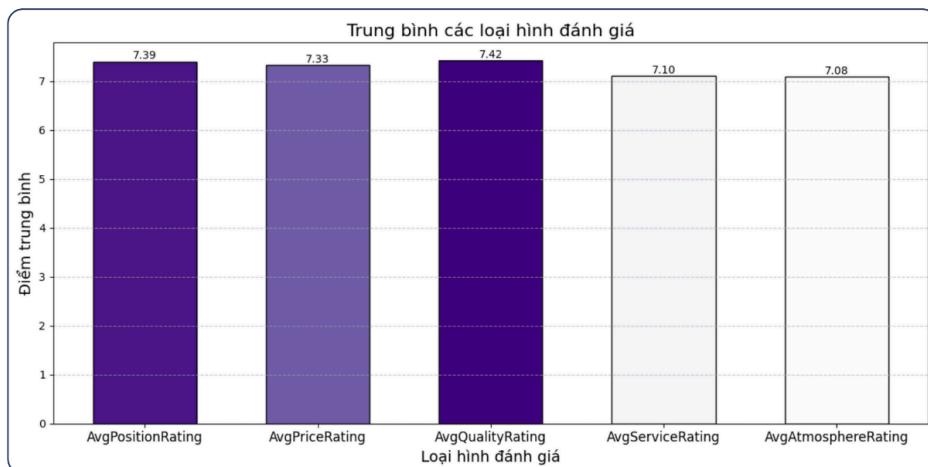
Tiếp theo, vẽ biểu đồ dạng Clustered Column để hiển thị cả 5 loại hình đánh giá với thư viện plt.bar

```

# Vẽ biểu đồ dạng Clustered Column Chart
plt.figure(figsize=(12, 6))
bars = plt.bar(
    df_melted["RatingType"],      # Trục X: Tên loại đánh giá
    df_melted["AverageScore"],    # Trục Y: Điểm trung bình
    color=colors,                # Ánh xạ màu sắc
    edgecolor="black",            # Viền cột màu đen để làm rõ
    width=0.6                    # Độ rộng cột
)

```

Kết quả biểu đồ sau khi chạy



## Chương 4. XÂY DỰNG MÔ HÌNH HỌC MÁY VÀ ỨNG DỤNG (R7)

### 4.1. Tổng quan quy trình xây dựng hệ thống đề xuất

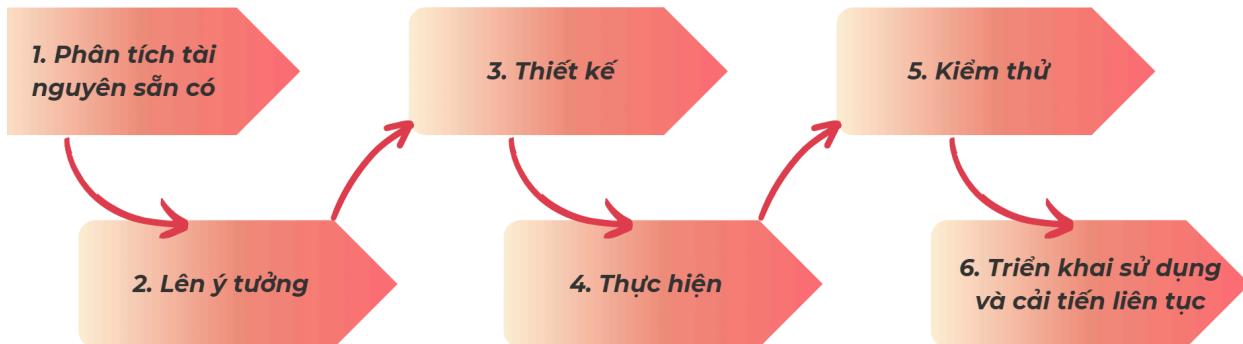
Nhóm sẽ dùng bộ dữ liệu đánh giá các cửa hàng trong thành phố Đà Nẵng để xây dựng hệ thống đề xuất cửa hàng tại Đà Nẵng ứng dụng mô hình học máy KNN.

Giới thiệu về hệ thống/Ý tưởng:

Một trong những việc mà chúng ta băn khoăn nhiều nhất trong đời có lẽ là “*bữa nay ăn gì?*”, và để hỗ trợ cho vấn đề nan giải này, nhóm đưa ra một hệ thống để xuất cửa hàng ăn uống với hy vọng sẽ cung cấp được câu trả lời dù cho câu hỏi này đột phát ra khi người dùng đang ở bất kỳ đâu trong Đà Nẵng.

Có khi nào lúc đang tụ tập với bạn bè ở quán cà phê, có dự định sau đó sẽ đi ăn cùng nhau và câu hỏi “*bữa nay ăn gì?*” sẽ xuất hiện, kèm theo đó là những yêu cầu như “nay cuối tháng rồi, ăn gì rẻ thôi!”, “đói rồi, tìm chỗ gần gần đi!”, “ bụng yếu, lựa quán có chất lượng nhé!”,... Không sao, “nó” sẽ trả lời cho bạn trong 3 giây. Cùng xem “nó” được tạo ra như thế nào nhé!

Quy trình xây dựng hệ thống:



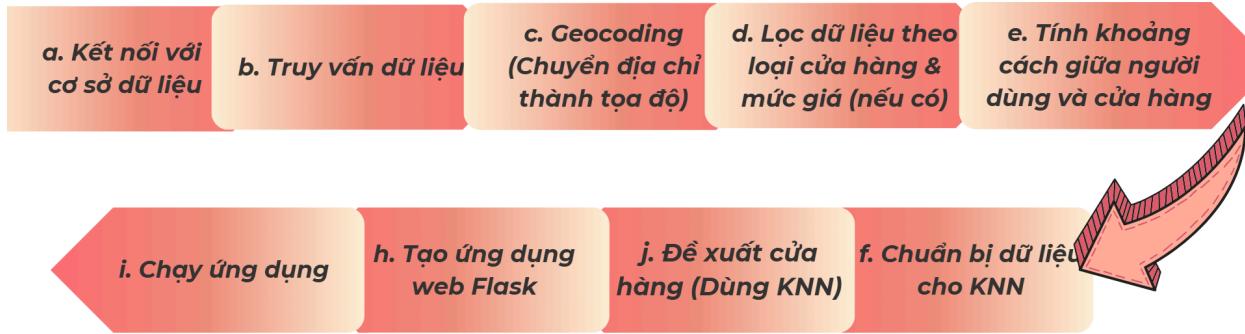
Những công đoạn trước nhóm đã tìm hiểu và phân tích dữ liệu, ngoài tài nguyên dữ liệu đã có trong sql server, nhóm dùng ngôn ngữ lập trình python đã được học để xây dựng hệ thống này.

Từ những dữ liệu đó đã cho nhóm ý tưởng về hệ thống, sau khi ý tưởng hình thành, thiết kế khung logic và thực thi theo khung đã thiết kế để có được hệ thống ban sơ, qua nhiều lần kiểm thử và cải thiện, hệ thống đã có thể sử dụng, trong quá trình sử dụng sẽ tiếp tục được cập nhật và chỉnh sửa.

## 4.2. Xây dựng hệ thống đề xuất

### 4.2.1. Back-end

Tổng quát quá trình xây dựng back-end được thể hiện qua hình dưới đây:



### - **Kết nối với cơ sở dữ liệu (SQL Server)**

Tương tự như ở việc lấy dữ liệu từ SQL để trực quan hóa bằng python, sử dụng thư viện pyodbc để kết nối với cơ sở dữ liệu SQL Server và lấy dữ liệu về các cửa hàng từ bảng DaNang\_Stores, kết hợp với các bảng liên quan để có được thông tin như tên cửa hàng, địa chỉ, đánh giá trung bình, phạm vi giá, loại cửa hàng, v.v.

### - **Lấy dữ liệu cửa hàng bằng truy vấn:**

Hàm get\_store\_data() thực hiện truy vấn SQL để lấy dữ liệu về các cửa hàng từ cơ sở dữ liệu. Dữ liệu bao gồm tên cửa hàng, địa chỉ, phạm vi giá, loại cửa hàng, và các đánh giá trung bình.

```
def get_store_data():
    query_ML = """
        SELECT ds.StoreID, ds.StoreName, ds.StoreAddress, ds.StoreLink,
               pr.PriceRangeName, sc.SCategoryName,
               sr.AvgRating, sr.AvgAtmosphereRating, sr.AvgPositionRating,
               sr.AvgPriceRating, sr.AvgQualityRating, sr.AvgServiceRating
        FROM DaNang_Stores ds
        JOIN PriceRange_Category pr ON ds.PriceRangeID = pr.PriceRangeID
        JOIN Store_Category sc ON ds.SCategoryID = sc.SCategoryID
        JOIN Store_Rating sr ON ds.StoreID = sr.StoreID
    """
    df = pd.read_sql(query_ML, conn)
    df['StoreAddress'] = df['StoreAddress'] + ", Đà Nẵng, Việt Nam"
    return df
df = get_store_data()
```

### - **Geocoding (Chuyển địa chỉ thành tọa độ):**

Sử dụng thư viện geopy để chuyển đổi địa chỉ của cửa hàng thành tọa độ địa lý (vĩ độ, kinh độ) thông qua API Nominatim. Hàm geocode\_address() xử lý việc tìm kiếm tọa độ và lưu trữ vào bộ nhớ cache để giảm thiểu số lần gọi API.

```

# Khởi tạo Geolocator
geolocator = Nominatim(user_agent="Foody_dexuat")

# Hàm geocoding để lấy vĩ độ và kinh độ từ địa chỉ
cache_coordinates = {}

def geocode_address(address):
    if address in cache_coordinates:
        return cache_coordinates[address]
    try:
        location = geolocator.geocode(address)
        if location:
            coordinates = (location.latitude, location.longitude)
            cache_coordinates[address] = coordinates
            return coordinates

```

Áp dụng hàm này lên từng giá trị trong cột StoreAddress.

```
: df['StoreAddress'].apply(geocode_address)
```

- **Lọc dữ liệu theo loại cửa hàng:**

Hàm filter\_data() lọc các cửa hàng dựa trên yêu cầu của người dùng như phạm vi giá (price\_range), loại cửa hàng (category).

```

# Hàm lọc dữ liệu theo yêu cầu
def filter_data(df, price_range=None, category=None):
    df = df[df['SCategoryName'] == category]
    if price_range:
        price_mapping = {
            "<50k": "Thấp",
            "50k-100k": "Trung bình thấp",
            "100k-200k": "Trung bình",
            ">200k": "Cao"
        }
        df = df[df['PriceRangeName'] == price_mapping.get(price_range)]
    return df

```

- **Tính khoảng cách giữa người dùng và cửa hàng:**

Hàm calculate\_distance() tính toán khoảng cách giữa vị trí của người dùng và các cửa hàng bằng công thức Haversine (sử dụng geopy.distance.geodesic).

```

# Hàm tính khoảng cách giữa hai điểm
def calculate_distance(user_lat, user_lon, store_lat, store_lon):
    if pd.isna(store_lat) or pd.isna(store_lon):
        return float('inf')
    user_location = (user_lat, user_lon)
    store_location = (store_lat, store_lon)
    return geodesic(user_location, store_location).km

```

- Chuẩn bị dữ liệu cho KNN:

Hàm prepare\_knn\_data() chuẩn bị dữ liệu cho thuật toán KNN bằng cách tính toán khoảng cách giữa người dùng và cửa hàng, cũng như điểm ưu tiên của cửa hàng (theo các yếu tố như đánh giá không khí, vị trí, giá cả, chất lượng, dịch vụ).

Các giá trị này được chuẩn hóa để tạo thành các đặc trưng đầu vào cho KNN (bảng knn\_features).

```

# Hàm chuẩn bị dữ liệu đầu vào cho KNN
def prepare_knn_data(df, user_lat, user_lon, priority):
    df['Distance'] = df.apply(lambda row: calculate_distance(user_lat, user_lon, row['Latitude'], row['Longitude']), axis=1)

    if priority:
        priority_mapping = {
            "Không khí": "AvgAtmosphereRating",
            "Vị trí": "AvgPositionRating",
            "Giá": "AvgPriceRating",
            "Chất lượng": "AvgQualityRating",
            "Dịch vụ": "AvgServiceRating"
        }
        priority_col = priority_mapping.get(priority, "AvgRating")
        df['PriorityScore'] = df[priority_col]
    else:
        df['PriorityScore'] = df['AvgRating']

    # Chuẩn hóa dữ liệu để KNN hoạt động tốt hơn
    df['NormalizedDistance'] = df['Distance'] / df['Distance'].max()
    df['NormalizedPriorityScore'] = df['PriorityScore'] / 10 # Rating từ 0 đến 10

    knn_features = df[['NormalizedDistance', 'NormalizedPriorityScore']].values
    return df, knn_features

```

- Đề xuất cửa hàng (Sử dụng KNN):

Hàm recommend\_stores() sử dụng thuật toán KNN để tìm ra các cửa hàng gần người dùng nhất dựa trên khoảng cách và điểm ưu tiên. KNN sử dụng NearestNeighbors từ thư viện sklearn để tìm ra các cửa hàng tương tự nhất dựa trên các đặc trưng đã chuẩn hóa.

```

# Hàm để xuất cửa hàng
def recommend_stores(df, user_lat, user_lon, price_range=None, category=None, priority=None):
    # Lọc dữ liệu
    df = filter_data(df, price_range, category)

    # Kiểm tra dữ liệu rỗng
    if df.empty:
        return "Không có cửa hàng thích hợp, vui lòng nhập yêu cầu khác nhé!"

    # Chuẩn bị dữ liệu cho KNN
    df, knn_features = prepare_knn_data(df, user_lat, user_lon, priority)

    # Kiểm tra số lượng cửa hàng sau lọc
    num_samples = len(knn_features)
    if num_samples == 0:
        return "Không có cửa hàng thích hợp, vui lòng nhập yêu cầu khác nhé!"
    elif num_samples < 3:
        # Nếu ít hơn 3 cửa hàng, trả về toàn bộ cửa hàng hiện có
        return df.iloc[:num_samples]
    else:
        try:
            # Áp dụng KNN nếu đủ dữ liệu
            knn = NearestNeighbors(n_neighbors=3, metric='euclidean')
            knn.fit(knn_features)
            distances, indices = knn.kneighbors([[0, 1]]) # Điểm lý tưởng: rating cao và khoảng cách gần
            top_indices = indices[0]
            return df.iloc[top_indices]
        except ValueError as e:
            return f"Không có cửa hàng thích hợp, vui lòng nhập yêu cầu khác nhé!"

```

### - Tạo ứng dụng web Flask:

Ứng dụng Flask phục vụ một trang web cho phép người dùng nhập địa chỉ và các yêu cầu như phạm vi giá, loại cửa hàng, và ưu tiên của họ.

Khi người dùng nhập dữ liệu và gửi yêu cầu (POST), ứng dụng Flask sẽ:

- Chuyển đổi địa chỉ người dùng thành tọa độ.
- Gọi hàm recommend\_stores() để nhận các cửa hàng phù hợp.
- Hiển thị kết quả hoặc thông báo nếu không có cửa hàng phù hợp.

```

# Khởi tạo Flask app
app = Flask(__name__)

@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        user_address = request.form['user_address']
        price_range = request.form.get('price_range')
        category = request.form['category']
        priority = request.form.get('priority')

        user_lat, user_lon = geocode_address(user_address)
        top3 = recommend_stores(df, user_lat, user_lon, price_range, category, priority)
        # Kiểm tra nếu top3 là chuỗi (thông báo) hoặc DataFrame
        if isinstance(top3, str):
            # Trả về thông báo nếu không có cửa hàng phù hợp
            return render_template('index.html', message=top3)
        else:
            # Trả về kết quả cửa hàng
            return render_template('index.html', stores=top3.to_dict(orient='records'))

    return render_template('index.html')

```

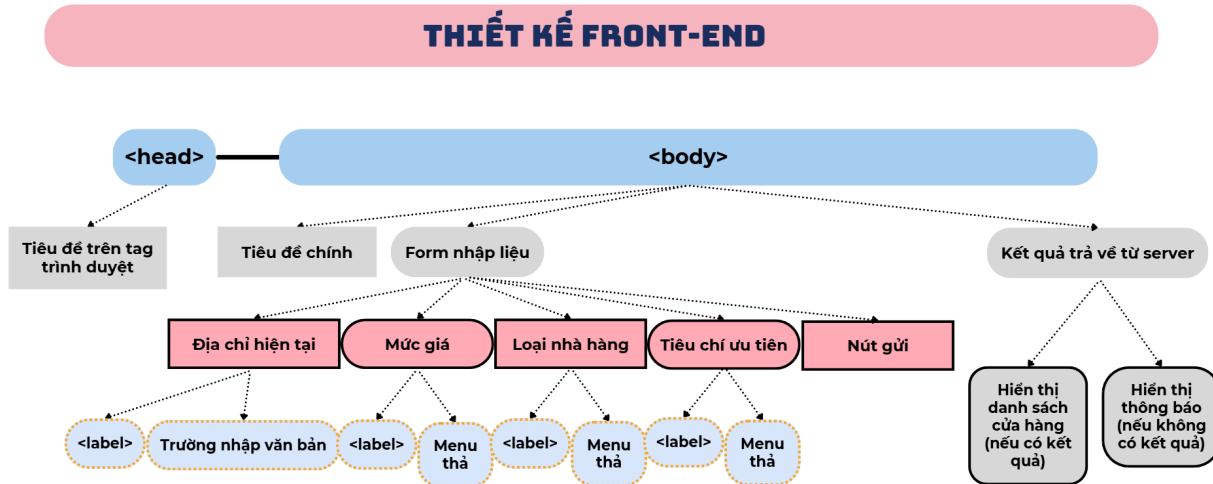
### - Chạy ứng dụng:

Ứng dụng Flask chạy trên môi trường phát triển (debug mode) và xử lý các yêu cầu HTTP, trả về kết quả dưới dạng trang HTML thông qua render\_template().

```
app.run(debug=True)
```

#### 4.2.2. Front-end

Tổng quát front-end được thể hiện qua hình dưới đây:



### - Phần đầu: Thông tin cơ bản

<!DOCTYPE html>: Xác định phiên bản HTML5.

lang="vi": Ngôn ngữ hiển thị là tiếng Việt.

meta charset="UTF-8": Hỗ trợ hiển thị ký tự tiếng Việt.

meta name="viewport": Đảm bảo giao diện responsive, tương thích với các thiết bị như điện thoại hoặc máy tính bảng.

title: Tiêu đề trang là "Đè xuất cửa hàng Đà Nẵng", sẽ hiện ở tag.

```

<!DOCTYPE html>
<html lang="vi">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Đè xuất cửa hàng Đà Nẵng</title>
</head>

```

### - Tiêu đề trang và biểu mẫu nhập liệu

Tiêu đề trang: "Đè xuất cửa hàng Đà Nẵng" hiển thị ở đầu trang.

Form nhập liệu (method="POST"): Người dùng nhập thông tin để gửi yêu cầu, các tiêu chí này sẽ được xử lý ở server-side (hậu trường).

```

<body>
    <h1>Đè xuất cửa hàng Đà Nẵng</h1>
    <form method="POST">

```

- + Trường nhập thông tin địa chỉ:

user\_address: Nhập địa chỉ hiện tại của người dùng.  
required: Trường này bắt buộc phải điền.

```

        <label for="user_address">Địa chỉ hiện tại:</label>
        <input type="text" name="user_address" required>

```

- + Lựa chọn mức giá: có 4 khoảng giá, hoặc chọn "Tất cả" để không lọc theo mức giá.

```

        <label for="price_range">Mức giá:</label>
        <select name="price_range">
            <option value="">Tất cả</option>
            <option value="<50k">Thấp</option>
            <option value="50k-100k">Trung bình thấp</option>
            <option value="100k-200k">Trung bình</option>
            <option value=">200k">Cao</option>
        </select>

```

- + Lựa chọn loại nhà hàng: Người dùng bắt buộc chọn một trong các loại đã được định nghĩa.

```

        <label for="category">Loại nhà hàng:</label>
        <select name="category">
            <option value="Ăn vặt/via hè">Ăn vặt/via hè</option>
            <option value="Ăn chay">Ăn chay</option>
            <option value="Café/Dessert">Café/Dessert</option>
            <option value="Nhà hàng">Nhà hàng</option>
            <option value="Quán ăn">Quán ăn</option>
            <option value="Quán nhậu">Quán nhậu</option>
            <option value="Tiệm bánh">Tiệm bánh</option>
        </select>

```

- + Lựa chọn tiêu chí ưu tiên: Chọn tiêu chí ưu tiên để đè xuất cửa hàng, chẳng hạn: Chất lượng, Vị trí, hoặc Không khí. Nếu không có tiêu chí nào, hệ thống sẽ dùng giá trị mặc định.

```

        <label for="priority">Tiêu chí ưu tiên:</label>
        <select name="priority">
            <option value="">Không chọn</option>
            <option value="Không khí">Không khí</option>
            <option value="Vị trí">Vị trí</option>
            <option value="Giá">Giá</option>
            <option value="Chất lượng">Chất lượng</option>
            <option value="Dịch vụ">Dịch vụ</option>
        </select>

```

- + Nút gửi yêu cầu: Khi nhấn nút này, toàn bộ thông tin từ biểu mẫu được gửi lên server để xử lý.

```
<button type="submit">Tìm cửa hàng</button>
```

#### - Kết quả đề xuất

- + Trường hợp có cửa hàng được đề xuất: nếu có danh sách cửa hàng (stores) được trả về từ server.  
Duyệt danh sách (vòng lặp for): Hiển thị thông tin từng cửa hàng  
Khoảng cách đến người dùng được làm tròn 2 chữ số thập phân (Distance).

```
{% if stores %}
    <h2>Kết quả đề xuất:</h2>
    <ul>
        {% for store in stores %}
            <li>
                <strong>{{ store['StoreName'] }}</strong><br>
                Địa chỉ: {{ store['StoreAddress'] }}<br>
                <a href="{{ store['StoreLink'] }}>Link cửa hàng</a><br>
                Loại: {{ store['SCategoryName'] }}<br>
                Điểm đánh giá: {{ store['AvgRating'] }}<br>
                Khoảng cách: {{ store['Distance']|round(2) }} km
            </li>
        {% endfor %}
    </ul>
```

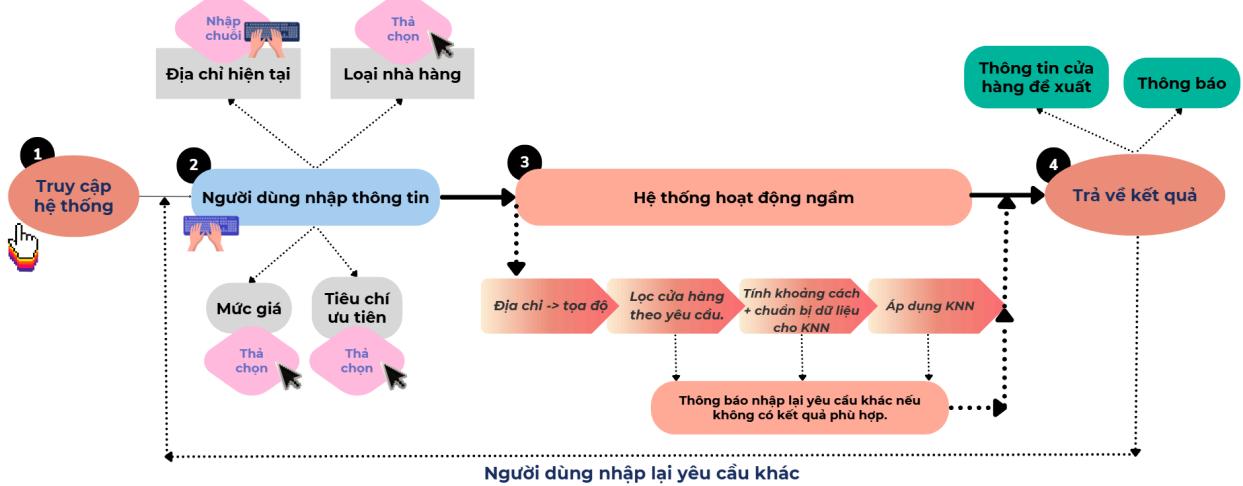
- + Trường hợp không có cửa hàng đề xuất: Nếu không có cửa hàng nào phù hợp, hiển thị thông báo từ server thông qua biến message.

```
{% elif message %}
    <p>{{ message }}</p>
{% endif %}
```

### 4.3. Nguyên lý vận hành và diễn thử

#### 4.3.1. Sơ đồ luồng hoạt động

Sau khi truy cập hệ thống, người dùng nhập thông tin và yêu cầu của mình, hệ thống sẽ hoạt động và đưa ra những cửa hàng đề cử hoặc đưa ra thông báo nhập lại yêu cầu khác vì không tìm thấy cửa hàng phù hợp.



#### 4.3.2. Demo

Nhập địa chỉ và chọn mức giá, loại nhà hàng, tiêu chí ưu tiên, hệ thống sẽ trả về kết quả như hình:

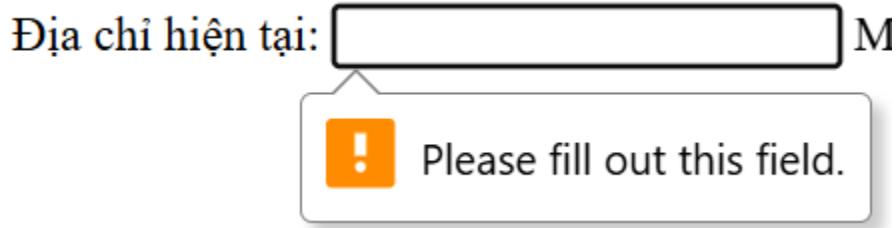
### Đè xuất cửa hàng Đà Nẵng

Địa chỉ của bạn: [30 Thành Duyên, Thanh Bình] Mức giá: [Trung bình thấp] Loại nhà hàng: [Quán ăn] Tiêu chí ưu tiên: [Chất lượng] Tìm cửa hàng

#### Kết quả đề xuất:

- **Tiệm Ăn Chợ Lớn - Ông Ích Khiêm**  
Địa chỉ: 267 Ông Ích Khiêm, Đà Nẵng, Việt Nam  
[Link cửa hàng](#)  
Loại: Quán ăn  
Điểm đánh giá: 7.09  
Khoảng cách: 1.39 km
- **MappiDo - Cháo Éch & Cháo Gan Rim**  
Địa chỉ: 105 Thái Phiên , P. Phước Ninh, Đà Nẵng, Việt Nam  
[Link cửa hàng](#)  
Loại: Quán ăn  
Điểm đánh giá: 7.88  
Khoảng cách: 2.14 km
- **Quán Ngon - Beefsteak**  
Địa chỉ: 368 Đồng Đa, P. Thanh Bình, Đà Nẵng, Việt Nam  
[Link cửa hàng](#)  
Loại: Quán ăn  
Điểm đánh giá: 8.78  
Khoảng cách: 2.59 km

Khi bỏ trống những ô bắt buộc nhập, sẽ hiện thông báo:



Khi không có cửa hàng phù hợp với yêu cầu, hệ thống sẽ đưa ra thông báo:

# Đề xuất cửa hàng Đà Nẵng

Địa chỉ hiện tại:  Mức giá:  Loại nhà hàng:  Tiêu chí ưu tiên:

Không có cửa hàng thích hợp, vui lòng nhập yêu cầu khác nhé!

## Chương 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 5.1. Tổng kết

Sau một hành trình dài đồng hành cùng dự án này, nhóm đã học được rất nhiều bài học và kỹ năng mới, đồng thời có cơ hội rèn luyện những kỹ năng đã có trước đây. Nhóm vô cùng biết ơn giảng viên tiến sĩ Nguyên Vũ và tổ giảng viên phụ trách học phần này đã tạo điều kiện cho nhóm có cơ hội tiếp cận với phương thức thực hành này, một dự án có thời gian đầu tư, chiều sâu, chiều rộng rất gần với thực tiễn, yêu cầu nhiều kỹ năng và đoàn đội phối hợp để thực hiện.

Nhờ thực hiện dự án, nhóm đã biết cách:

- Cào dữ liệu.
- Nhập và lấy dữ liệu giữa ngôn ngữ lập trình python và SQL Server.
- Tiền xử lý bằng T-SQL.
- Sao lưu dữ liệu cả thủ công và tự động.
- Phân quyền cho cơ sở dữ liệu.
- Trực quan dữ liệu bằng Power BI và Python.
- Dụng web đơn giản.

Dưới sự hướng dẫn của giảng viên, nhóm được tiếp cận với những công cụ làm việc với dữ liệu mới, phát triển thêm những kỹ năng cứng mới, đồng thời tôi luyện những kỹ năng mềm như kỹ năng làm việc nhóm, kỹ năng lập kế hoạch, kỹ năng thuyết trình,...và các cách nắm bắt bản chất của vấn đề, cách trình bày khoa học hơn và nhiều bài học giá trị giúp ích cho nhóm trong hành trình tương lai, trên con đường học tập cũng như công việc sau này.

### 5.2. Hướng phát triển trong tương lai

- Crawl thêm dữ liệu đánh giá và thông tin đa chiều:
  - + Tiếp tục thu thập dữ liệu từ nền tảng đánh giá Foody như nội dung đánh giá, ... để đảm bảo nguồn dữ liệu phong phú và đa dạng
- Tự động cập nhật dữ liệu và cải thiện mô hình:
  - + Xây dựng pipeline ETL tự động cập nhật dữ liệu thường xuyên từ các nguồn online đảm bảo rằng hệ thống luôn có dữ liệu mới nhất
- Cải thiện trải nghiệm người dùng (UX/UI):
  - + Phát triển giao diện người dùng trực quan, dễ sử dụng, với các công cụ lọc linh hoạt theo tiêu chí như loại hình dịch vụ, đánh giá, giá cả, và vị trí.
  - + Sử dụng biểu đồ, bản đồ và các loại hình visualization khác để hiển thị thông tin một cách rõ ràng, giúp người dùng dễ dàng tìm hiểu và so sánh các cửa hàng