TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO BÀI TẬP LỚN
TÍCH HỢP DỮ LIỆU

ĐỀ TÀI: Xây dựng hệ thống tích hợp dữ liệu cho ô tô từ nhiều nguồn dữ liệu khác nhau

Giáo viên hướng dẫn: TS. Đỗ Bá Lâm

Nhóm sinh viên thực hiện: Nhóm 11

Phạm Minh Khôi	20183566
Đỗ Lương Kiên	20183568
Nguyễn Hữu Kiệt	20183571
Nguyễn Đình Lâm	20183574

Hà Nội, 2022

Mục lục

Chương 1: Giới thiệu bài toán

Trong chương này, sẽ mô tả động lực và sơ lược về bài toán mà nhóm giải quyết. Cùng với phương pháp tiếp cận bài toán mà nhóm đưa ra.

1.1 Giới thiệu

Tích hợp dữ liệu là việc kết hợp dữ liệu đồng nhất hoặc không đồng nhất từ các nguồn khác nhau để đưa vào một lược đồ chung nhằm phục vụ mục đích đọc ghi, truy xuất và xử lý dữ liệu một cách tổng quan và thống nhất hơn. Việc tích hợp ngày càng trở nên phổ biến khi mà khối lượng dữ liệu ngày càng lớn và tần suất cũng như nhu cầu sử dụng tăng lên theo, ví dụ như trong thương mại (trao đổi mua bán hàng hóa trực tuyến), giáo dục (dạy và học online, nghiên cứu tài liệu từ các nguồn toàn cầu) để tạo ra các bản báo cáo tối ưu lợi nhuận … Vì vậy việc tích hợp cần phải có một hệ thống rõ ràng để lấy thông tin từ nhiều nguồn mà vẫn giữ được ràng buộc; để có thể thống nhất và đưa ra câu trả lời đúng nhất đối với các yêu cầu từ người dùng.

Trên cơ sở đó, nhóm đã đề xuất đề tài "Xây dựng hệ thống tích hợp dữ liệu cho ô tô từ nhiều nguồn" để có một cái nhìn thực tế hơn đối với công việc tích hợp dữ liệu, đồng thời tích góp được những trải nghiệm khi đã xây dựng được một hệ thống có đầy đủ chức năng và giao diện phục vụ cho việc gợi ý và đề xuất sản phẩm, cụ thể ở đây là ô tô.

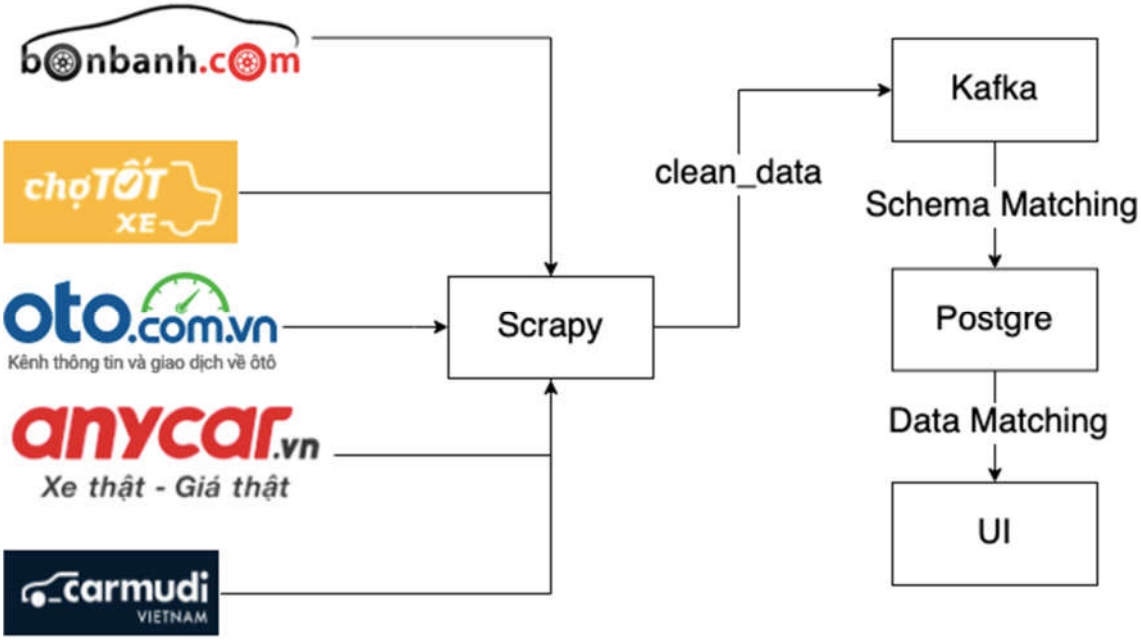
1.2 Mô tả bài toán

Sau một thời gian khủng hoảng bởi đại dịch Covid, nhu cầu mua sắm và đi lại của mọi người đang được khuyến khích và đều được tất cả mọi người ủng hộ. Trong đó, ô tô lại là một sản phẩm đắt tiền đáp ứng được 2 mục đích trên nên đang nhận được sự chú ý từ cả hai bên là người bán và người mua. Để có thể phục vụ cho việc gợi ý giá cả cũng như giảm bớt thời gian tham khảo các mẫu xe cho những người đang có ý định mua xe, nhóm đề xuất xây dựng một hệ thống tích hợp dữ liệu ô tô từ nhiều nguồn.

Hệ thống sẽ giải quyết những mục tiêu như:

- Giao diện để người dùng nhập các trường thông tin cần thiết.
- Đưa ra các sản phẩm liên quan đến nhau như cùng một tầm giá, cùng hãng, …

Công việc sẽ bao gồm: thu thập dữ liệu. xử lý dữ liệu để đưa vào hàng đợi kafka, tích hợp vào một nguồn đích và lưu trữ ở database, người dùng truy xuất database thông qua một giao diện đơn giản.



Hình ảnh 1: Pipeline xây dựng hệ thống tích hợp

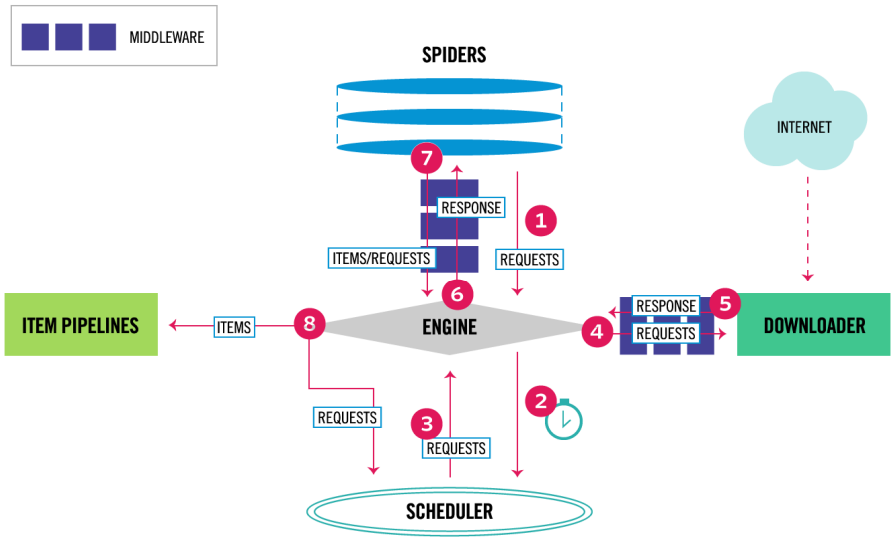
Hình trên là toàn bộ pipeline xây dựng của hệ thống, về mặt triển khai hệ thống để cho tiện chạy lại và không phụ thuộc vào máy từng cá nhân. Nhóm sử dụng phần mềm “DOCKER" để cài đặt các framework như Kafka, Postgre (“postgre database” và “pgadmin" để monitor). Về phần phía backend xử lý data matching, nhóm sử dụng flask (hoặc fastapi) để tạo ra các route api cho phía client javascript truy xuất, gọi đến để thực hiện lấy dữ liệu load ra phía người dùng.

Chương 2: Khái niệm và phương pháp tiếp cận

Trong phần này, nhóm sẽ trình bày về một vài khái niệm: Scrapy, Kafka,... những công cụ mà được sử dụng để xử lý bài toán. Trình bày chi tiết những bước giải quyết bài toán.

2.1 Công nghệ sử dụng để thu thập dữ liệu

- Công nghệ chính mà nhóm sử dụng là Scrapy, bởi nó có rất nhiều những tiện ích và cấu hình. Cấu trúc thư mục module hoá và trực quan, dễ hiểu. Đáp ứng được các nhu cầu thu thập dữ liệu cơ bản và nhanh chóng. Hơn thế nữa, Scrapy có thể thực hiện điều hướng sử dụng ngôn ngữ “lua" để có thể thực hiện các thao tác người dùng trên các mã javascript như “selenium" mà chúng ta thường dùng.
- Đối với trang web oto.com.vn, do kiến thức ban đầu còn hạn chế và chưa thành thạo scrapy. Nhóm có sử dụng selenium để crawl dữ liệu từ trang web này. Nhưng số lượng các item lên đến gần 40k, trong 3h chỉ có thể crawl được 800 item. Vì tính không hiệu quả, nhóm đã nghĩ cách crawl lại bằng scrapy và thấy được tốc độ và độ chính xác của scrapy mang lại.



Hình 2: Kiến trúc framework scrapy gồm các module

Các thành phần của Scrapy bao gồm:

- Scrapy Engine: có trách nhiệm kiểm soát luồng dữ liệu giữa tất cả các thành phần của hệ thống và kích hoạt các sự kiện khi một số hành động xảy ra.

- Scheduler: hoạt động như một hàng đợi, sắp xếp thứ tự các URL cần tải.

- Downloader: thực hiện download từ web và cung cấp cho engine.

- Spiders: là class được viết bởi người dùng, có trách nhiệm bóc tách dữ liệu cần thiết và tạo các url mới để nạp lại scheduler qua engine.

- Item pipeline: dữ liệu được bóc tách từ spiders sẽ được đưa về đây để xử lý và lưu trữ.

- Middlewares: là các thành phần khác, chúng đều có mục đích là giúp người dùng có thể tùy biến, mở rộng khả năng xử lý cho các thành phần, có các loại middlewares bao gồm:

  - Spider middlewares: là thành phần nằm giữa Engine và Spiders, chúng xử lý các response đầu vào của Spiders và đầu ra.
  - Downloader middlewares: nằm giữa Engine và Downloader, chúng xử lý các request được đẩy vào từ Engine và các response được tạo ra từ Downloader.
  - Scheduler middlewares: nằm giữa Engine và Scheduler để xử lý những request giữa hai thành phần.

2.2. Thu thập dữ liệu dữ liệu

2.2.1. Các nguồn dữ liệu thu thập

- Phân tích cấu trúc trang web:

  Đầu tiên phải xem trang web định crawl dữ liệu có cung cấp service để dùng api lấy dữ liệu trực tiếp luôn để không phải mất công crawl; hay trang web có cấu trúc css như thế nào, sử dụng script để fetch dữ liệu không; có thiết kế để chặn các bot crawl hay không?

  Sau khi phân tích cấu trúc trang web thì tiếp theo công cụ chính mà nhóm chọn sử dụng để crawl dữ liệu về là Scrapy, bởi vì công cụ sử dụng đơn giản, có api hỗ trợ để vượt qua việc web chặn bot crawl.
- Thu thập dữ liệu:

Nhóm đã tham khảo các dữ liệu từ nhiều nguồn và chọn ra được các nguồn có trường thông tin khá tương đồng đó là:

- Carmudi: https://www.carmudi.vn/
- Bonbanh: https://bonbanh.com/
- Chotot: https://xe.chotot.com/
- Anycar: https://anycar.bonbanh.com/
- Oto.com : https://oto.com.vn/

Dữ liệu đều lấy từ các nguồn trên về bằng scrapy: mỗi nguồn lấy khoảng 20pages đầu thì được khoảng 400 bản ghi mỗi trang web. Dữ liệu được lưu thành các file csv với các trường thuộc tính như: url, tên, giá, hãng xe, màu xe, …

```
item = {"id": str(uuid.uuid4()),
        "domain": self.allowed_domains[0],
        "url": response.url,
        "crawled_date": datetime.now(),
        "ten": response.xpath('//h1[@class="title-detail"]/text()').extract_first().lower().strip(),
        "gia_ban": ' '.join(response.xpath('//span[contains(@class, "price-big")]/text()').extract()).strip().lower(),
        "gia_lan_banh": response.xpath('//span[@id="chiphilanbanh"]//text()').extract()[-1].lower()
        }
tmp = response.xpath('//div[@class="box-info-detail"]//ul[@class="list-info"]//li//text()').extract()
thong_so_ky_thuat = []
for _ in tmp:
    _ = _.strip()
    if len(_) > 1:
        thong_so_ky_thuat.append(_)
key = None
for _, tmp in enumerate(thong_so_ky_thuat):
    if _ % 2 == 0:
        key = unidecode(tmp.replace(" ", "_").lower())
    else:
        item[key] = tmp.lower()
```

Hình 3. trích xuất thông tin bằng xpath selector

2.2.2. Xử lý sau khi crawl

- Loại bỏ các dữ liệu không phù hợp như bị lặp lại hoàn toàn hay bị trống quá nhiều trường thông tin.
- Xử lý mềm như loại bỏ khoảng trắng, các ký tự thực sự không cần thiết.

- Xử lý riêng trường thuộc tính của từng file dữ liệu như :

Hình 4. Thông số cơ bản của ô tô

Đối với bonbanh.com, trường "giá" được tách ra từ trường "tên", còn trường "dung tích xi lanh" thì cũng nằm trong "nhiên liệu".

Một số trường sẽ được thống nhất về chỉ có 2 thuộc tính: cũ/mới đối với "tình trạng", trong nước/ nhập khẩu đối với "xuất xứ".

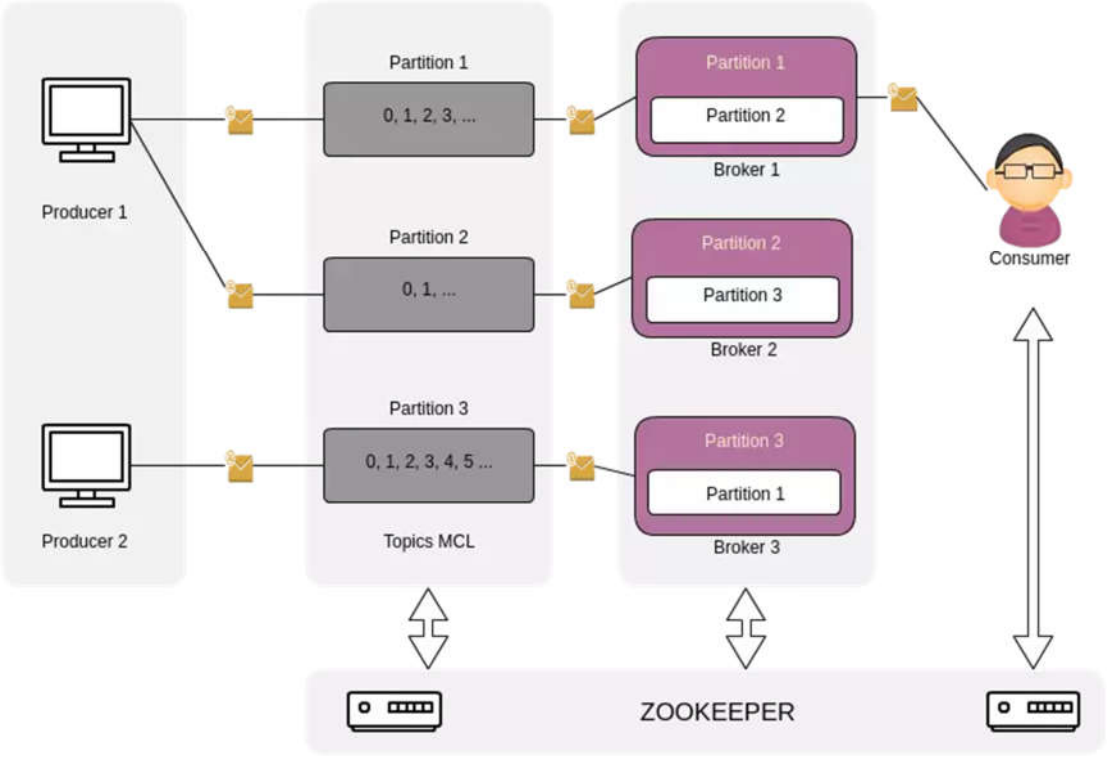
2.3. Kafka

Apache Kafka là hệ thống truyền thông điệp phân tán, độ tin cậy cao, dễ dàng mở rộng và có thông lượng cao. Kafka cung cấp cơ chế offset để lấy thông điệp một cách linh hoạt, cho phép các ứng dụng xử lý lại dữ liệu nếu việc xử lý trước đó bị lỗi. Apache Kafka có những đặc điểm như :

- Tốc độ nhanh: một máy đơn Kafka có thể xử lý số lượng dữ liệu đến hàng trăm megabyte trong một giây.
- Khả năng mở rộng: khi Kafka chạy trên một cụm, luồng dữ liệu sẽ được phân chia và được vận chuyển tới các nút trong cụm, do đó cho phép trung chuyển các dữ liệu mà có khối lượng lớn hơn nhiều so với sức chứa của một máy đơn.
- Độ tin cậy: Dữ liệu vào hàng đợi sẽ được lưu trữ trên ổ đĩa và được sao chép tới các nút khác trong cụm để ngăn ngừa việc mất dữ liệu.

Kafka sử dụng mô hình truyền thông public-subscribe, bên public dữ liệu được gọi là producer còn subscribe nhận dữ liệu theo topic được gọi là consumer. Các thành phần chính của Kafka như sau:

- Topic: Topic tương tự như một table trong CSDL SQL và là nơi chứa các message. Dữ liệu trong kafka theo topic, khi muốn truyền dữ liệu khác nhau hay truyền cho các ứng dụng khác nhau ta sẽ tạo ra các topic
- Partition: một topic được phân chia thành nhiều partition (là nơi chứa dữ liệu cho các topic). Các dữ liệu được lưu theo một thứ tự bất biến (offset) . Một partition sẽ có tối thiểu 1 replica và số lượng replica luôn nhỏ hơn số lượng broker.
- Broker: kafka chạy trên một cụm nhiều máy (node) thì các máy đó được gọi là broker. Broker là nơi lưu trữ các partition, một broker có thể lưu trữ nhiều partition
- Producer: là nơi đẩy dữ liệu từ người dùng vào các partition của topic. Tùy vào việc có chỉ định rõ ghi vào phân vùng nào không thì producer sẽ gửi phân vùng của topic đó hoặc phân bổ đều vào các partition
- Consumer: sẽ đọc dữ liệu từ broker. Kafka là hệ thống sử dụng mô hình truyền thông public-subscribe nên mỗi một topic có thể đc xử lý bởi nhiều consumer khác nhau, miễn là consumer đấy subcribe topic đấy.



Hình 5. Cấu trúc Kafka

=> Kafka được sử dụng làm một hàng đợi để khi hệ thống crawl dữ liệu sẽ đẩy vào kafka rồi mới đưa vào cơ sở dữ liệu, tránh quá tải hệ thống.

2.4. Postgre

Giống như tên của nó, PostgreSQL thực chất là một phiên bản mở rộng của “Mysql". Nó thêm một vài các chức năng mở rộng để hỗ trợ cho các ứng dụng yêu cầu xử lý lượng dữ liệu cao, tăng hiệu năng. Khá thích hợp làm database để truy xuất làm báo cáo của datawarehouse.

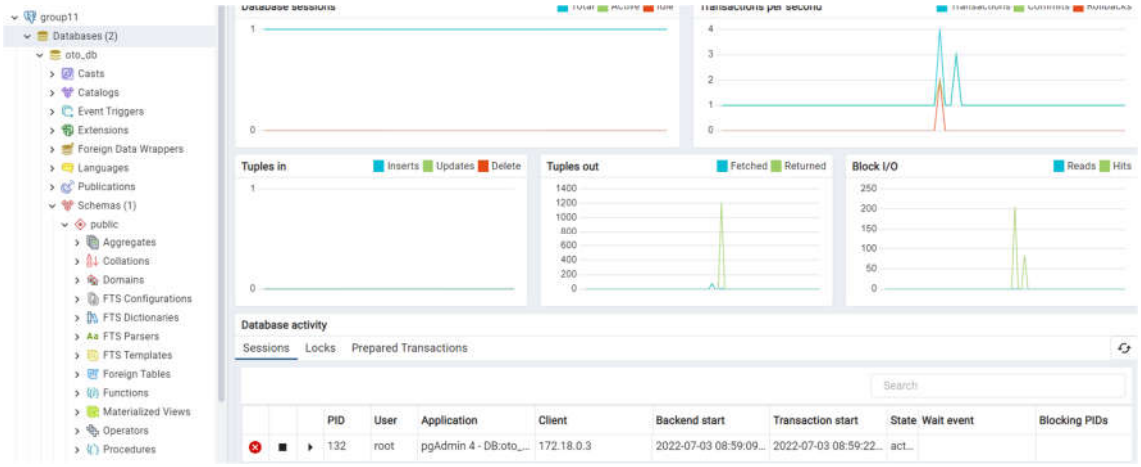
PostgreSQL là một hệ thống quản trị cơ sở dữ liệu quan hệ và đối tượng (object-relational database management system) miễn phí và nguồn mở (RDBMS) tiên tiến nhất hiện nay. khả năng mở rộng cao và tuân thủ các tiêu chuẩn kỹ thuật. Nó được thiết kế để xử lý một loạt các khối lượng công việc lớn, từ các máy tính cá nhân đến kho dữ liệu hoặc dịch vụ Web có nhiều người dùng đồng thời.

Các ưu điểm của PostgreSQL có thể kể đến như:

- Câu truy vấn phức hợp
- Truy vấn xử lý song song
- Sao chép dữ liệu dạng luồng

Là hệ thống quản lý cơ sở dữ liệu đối tượng, PostgreSQl cho phép thêm các tính năng tùy chỉnh được phát triển bằng các ngôn ngữ như C++, Java… Bên cạnh đó luôn có một cộng đồng lớn sẵn sàng cung cấp các giải pháp đối với vấn đề gặp phải khi làm việc với PostgreSQL.

Hình 6. Giao diện PostgreSQL



Hình 7. Giao diện trang chủ PostgreSQL

2.5. Schema matching

Thuật toán sử dụng: COMA từ thư viện Valentine

- COMA sử dụng cả tên trường và tên thuộc tính để đối sánh lược đồ
- Ánh xạ lần lượt từng cặp thuộc tính giữa hai lược đồ, từ đó tính ra độ tương đồng và chọn ra các cặp có độ tương đồng lớn.

=> Xây dựng một lược đồ chung, rồi lần lượt ánh xạ các lược đồ nguồn đối với lược đồ chung để tổng hợp.

Thuộc tính	Mô tả
id	Id sản phẩm khi đưa vào lược đồ chung
domain	trang web nguồn sản phẩm
url	nguồn sản phẩm
crawled_date	thời gian crawl
anh_xe	url của ảnh xe chính của bài đăng bán
ten	tên sản phẩm
gia_ban	giá sản phẩm
nam_san_xuat	năm sản xuất
xuat_xu	nguồn gốc sản phẩm (trong nước/ nhập khẩu)
tinh_trang	tình trạng sản phẩm (cũ/ mới)

kieu_dang	kiểu dáng
so_km_da_di	quãng đường đã đi
mau_ngoai_that	màu xe bên ngoài
mau_noi_that	màu xe bên trong
so_cua	số cửa
so_cho_ngoi	số chỗ ngồi
nhien_lieu	nhiên liệu
hop_so	hộp số
dan_dong	dẫn động
dung_tich_xi_lanh	dung tích xi lanh
thong_tin_mo_ta	thông tin mô tả

```
project_path = os.getcwd()
data_path = project_path + '/WebScrapy/Crawl_Data'
mediated_schema, anycar_bonbanh, bonbanh, carmudi, chotot, oto = read_data(data_path=data_path)
matcher = Coma(strategy="COMA_OPT_INST")
config_mapping = project_path + '/config_mapping'
schema_matching(matcher=matcher, config_folder=config_mapping)
```

Hình 7. Schema matching

Kết quả độ tương quan nhận được khi ánh xạ giữa lược đồ chung và lược đồ file nguồn oto.com.vn:

```
{(('mediated_schema', 'crawled_date'), ('oto', 'crawled_date')): 0.6956766,
 (('mediated_schema', 'domain'), ('oto', 'domain')): 0.6794861,
 (('mediated_schema', 'gia_ban'), ('oto', 'gia_ban')): 0.6818517,
 (('mediated_schema', 'hop_so'), ('oto', 'hop_so')): 0.8639822,
 (('mediated_schema', 'id'), ('oto', 'id')): 0.66820407,
 (('mediated_schema', 'kieu_dang'), ('oto', 'kieu_dang')): 0.85735285,
 (('mediated_schema', 'nam_san_xuat'), ('oto', 'nam_san_xuat')): 0.8115007,
 (('mediated_schema', 'nhien_lieu'), ('oto', 'nhien_lieu')): 0.8834623,
 (('mediated_schema', 'ten'), ('oto', 'ten')): 0.73533446,
 (('mediated_schema', 'thong_tin_mo_ta'), ('oto', 'mo_ta')): 0.44183627,
 (('mediated_schema', 'tinh_trang'), ('oto', 'tinh_trang')): 0.81426734,
 (('mediated_schema', 'url'), ('oto', 'url')): 0.688542,
 (('mediated_schema', 'xuat_xu'), ('oto', 'xuat_xu')): 0.8818517}
```

Hình 8. Kết quả khi so sánh lược đồ chung với trang oto.

Một điểm đặc thù của bài toán tích hợp của nhóm, tên đặc tính của hầu hết các trang web đều theo một chuẩn chung và khá tương đồng nhau. Một số trang web chỉ khác nhau về một số thuộc tính như ‘kieu_dang' với ‘dong_xe' hay ‘so_ghe_ngoi' vs ‘so_cho_ngoi', ‘mo_ta' vs ‘thong_tin_mo_ta' …. Lúc này, thuật toán COMMA sử dụng cả so sánh dựa trên độ tương đồng về các giá trị của thuộc tính. Qua kiểm tra, thuật toán cho kết quả tối ưu và phù hợp với miền bài toán.

2.6. Data matching

2.6.1. Khái niệm

Data matching là quá trình so khớp 2 hay nhiều tập dữ liệu với nhau. Mục đích của quá trình này là để tìm ra các bản ghi thuộc các tập dữ liệu khác nhau đang cùng đề cập đến một đối tượng. Các dữ liệu có thể đến từ nhiều nguồn khác nhau mà không có một lược đồ chung nào. Việc cần làm của data matching là xác định xem các bản ghi nào của các tập dữ liệu là tương đồng để có thể giảm bớt sự trùng lặp khi đưa các nguồn về chung một data warehouse.

2.6.2. Thực hiện

Phương pháp: phân cụm dựa trên độ đo khoảng cách
Ý tưởng:

- Các xe sẽ thuộc về một hãng xe thì tên của hãng xe sẽ nằm trong tên của xe. Tên của hãng xe sẽ không trùng nhau và cũng không nằm trong tên của xe hãng khác (nếu có điều này thì các hãng đã kiện nhau rồi). Vì vậy, dựa vào tên của xe sẽ phân được xe nào thuộc hãng nào.
- Trong mỗi hãng, ta sẽ thực hiện phân cụm dựa trên khoảng cách giữa trường ‘ten’ của các bản ghi.

Các vấn đề cần giải quyết:

- Lựa chọn độ đo khoảng cách
- Cần chọn bao nhiêu cụm
- Giải thuật dừng khi nào (sau một lần lặp, …)
- Hiệu năng
- Nhiễu, …
- …

Lựa chọn độ đo khoảng cách:

- Dựa trên thực nghiệm
- Dựa trên mục đích của việc tích hợp đó là đem các xe của cùng một dòng xe gom vào một nhóm, đồng thời loại hạn chế query các bản ghi trùng lặp
- Dựa trên đặc trưng của dữ liệu, vì tên của các xe giữa các trang sẽ có sự tương đồng về cách viết
  Ví dụ: Xe Toyota Vios 2.4G 2019, Toyota Vios 2018, …

Vậy nên chỉ cần chỉnh sửa một chút tên của xe này là có thể chuyển thành tên của xe kia, khi đó có thể dùng khoảng cách Levenshtein (khoảng cách chỉnh sửa, edit distance) làm độ đo khoảng cách cho giải thuật phân cụm.

Số lượng cụm cần chọn để bắt đầu phân cụm:

- Có thể dựa trên số loại xe của mỗi hãng (cách này rất tốn thời gian và công sức, hơn nữa còn có thể bị out of date)
- Có thể dựa trên dữ liệu hiện tại (ta sẽ dùng cách này): ta sẽ duyệt lần lượt qua các xe, đến xe nào ta sẽ lấy xe đó làm sample, rồi dùng sample để tính khoảng cách đến các xe còn lại, nếu khoảng cách nhỏ hơn một ngưỡng nào đó thì ta cho tạm các xe đó thành một cụm, đồng thời loại các xe trong cụm đó ra khỏi tập các xe, thực hiện đến khi nào tập các xe rỗng.
- Với mỗi cụm, ta sẽ tìm lại điểm trung bình của cụm (điểm mà có trung bình khoảng cách đến các điểm khác trong cụm là nhỏ nhất), rồi lấy các điểm trung bình đó như các điểm khởi tạo ban đầu của giải thuật phân cụm.

Vấn đề về hiệu năng:

- Một trong những vấn đề lớn đó là giải thuật phân cụm đòi hỏi chạy trên toàn bộ tập dữ liệu, vì vậy khi dữ liệu phình to ra việc update regular data warehouse sẽ tương đối tốn kém.
- Có thể tăng hiệu năng bằng cách đánh chỉ mục, hoặc dùng lại các cụm đã có

Kết quả:

- Chất lượng phân cụm khá tốt với các hãng xe phổ thông vì tên của các xe phổ thông thường dài nên khoảng cách Levenshtein sẽ hoạt động tốt, còn với các hãng xe cao cấp thì kết quả tương đối kém vì tên của các dòng xe cao cấp thường được đánh chỉ số chứ không được gọi bằng tên như các dòng xe phổ thông (ví dụ Audi A8, Q8, …; Mercedes Benz S400, S450, S500, ...)

- Hiệu năng chưa thực sự tốt

Khi thực hiện query có dùng thêm thư viện fuzzyfinder để lọc các nhóm liên quan.

```
Xe Toyota Vios 1.5E CVT 2018
Xe Toyota Vios 1.5G 2020
Xe Toyota Vios 1.5G 2020
Xe Toyota Vios 1.5G 2019
Xe Toyota Vios 1.5G 2020
Xe Toyota Vios 1.5G 2016
Xe Toyota Vios E CVT 2021
Xe Toyota Vios 1.5G 2020
Xe Toyota Vios G 1.5 CVT 2022
Xe Toyota Vios 1.5G 2020

########################################################################################################

Xe Toyota Corolla altis 2.0V Sport 2018

########################################################################################################

Xe Toyota Sienna Limited 3.5 2015

########################################################################################################

Xe Toyota Yaris 1.33 AT 2014
Xe Toyota Yaris 1.3 AT 2011

########################################################################################################

Xe Toyota Camry 2.5Q 2019
Xe Toyota Camry 2.5Q 2021
Xe Toyota Camry 2.0E 2017
Xe Toyota Camry LE 2.5 2009
Xe Toyota Camry 2.4G 2010
Xe Toyota Camry 2.5Q 2020
Xe Toyota Camry 2.5Q 2019
Xe Toyota Camry 2.0G 2019
Xe Toyota Camry 2.5Q 2019
Xe Toyota Camry 2.5Q 2019
Xe Toyota Camry 2.0G 2020

########################################################################################################

Xe Toyota Land Cruiser 4.6 V8 2021
Xe Toyota Land Cruiser VX 4.6 V8 2016
Xe Toyota Land Cruiser 4.6 V8 2020
Xe Toyota Land Cruiser 3.5 V6 2022

########################################################################################################
```

Hình 9: Kết quả của phân cụm trên một hãng xe

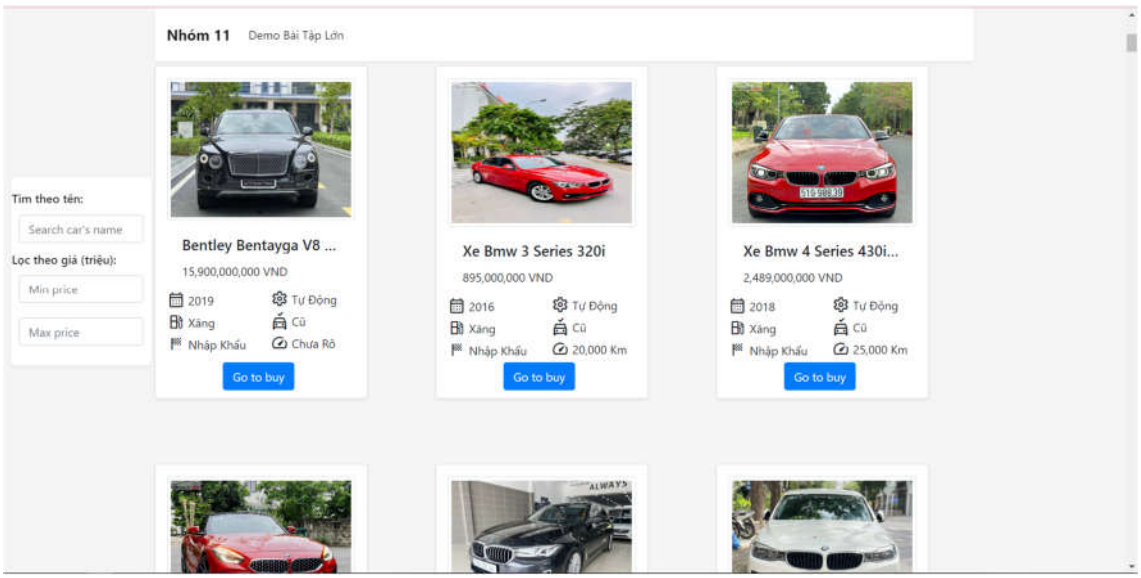
```
##########################################################################################################
Toyota Vios G 2020 - Bảo hành chính hãng tại Toyota
##########################################################################################################
Bán Toyota Land Cruiser Prado VX 2018 cũ Hà Nội
##########################################################################################################
Bán Toyota Yaris 1.33 xe nhập Pháp
##########################################################################################################
"Toyota Prado 2.7TX.L 2014, trang bị nhiều đồ chơi xin - Liên hệ : 0915.39.39.39 Thành"
##########################################################################################################
Toyota Fortuner AT Dầu 4x2 2019
##########################################################################################################
"Toyota Vios G 2019, Xe Cực Đẹp, Odo 67.000km"
##########################################################################################################
TOYOTA VELOZ 1.5 TOP MÀU BẠC TÍM GIAO NGAY
##########################################################################################################
```

Hình 10: Nhiễu khi thực hiện phân cụm

Chương 3: Giao diện hệ thống

Sau bước data matching, số record còn lại là 207. Có khá nhiều ô tô mà được bán từ hơn 1 nguồn.

Dưới đây là hình ảnh giao diện của hệ thống mà nhóm xây dựng



Hình 11: Giao diện hệ thống

Hệ thống hỗ trợ tìm theo tên xe, nhãn hiệu của xe, kèm theo đó là hỗ trợ lọc theo khoảng giá do người dùng nhập vào.

Danh sách các xe biểu diễn đầy đủ thông tin giúp người dùng dễ dàng đánh giá rằng chiếc xe đó có thực sự phù hợp với tiêu chuẩn của mình đưa ra hay không, ví dụ như: năm sản xuất, chạy bằng nhiên liệu gì? Nhập khẩu hay trong nước ? Số tự động hay số sàn hay bán tự động? Là xe cũ hay mới? Nếu cũ thì đã chạy được bao nhiêu km rồi?...

Dựa vào đó, người dùng dễ dàng tìm kiếm thông tin các loại xe đang được bày bán. Khi người dùng chốt 1 chiếc xe nào đó, thì bấm nút "**Go to buy**" để điều hướng trực tiếp người dùng đến trang bán gốc của sản phẩm, ở đây người dùng có thể xem chi tiết thông tin chiếc xe mình muốn mua.

Chương 4: Tổng kết

Nhóm đã xây dựng được một hệ thống tích hợp dữ liệu ô tô từ nhiều nguồn khác nhau với đầy đủ các chức năng cần thiết. Tuy nhiên do kiến thức cũng như thời gian còn hạn chế, hệ thống chỉ giải quyết vừa đủ các vấn đề được đặt ra ngay từ đầu, chưa thực sự hiệu quả cũng như tối ưu để có thể sử dụng thực tế. Vì vậy hướng phát triển tiếp theo trong tương lai, hệ thống có thể mở rộng thêm các nguồn dữ liệu; tự động thu thập dữ liệu (airflow) và thu thập nhiều ngày để theo dõi biến động giá; thử sử dụng các phương pháp data matching để tăng độ chính xác giữa các cụm dữ liệu, …

PHỤ LỤC

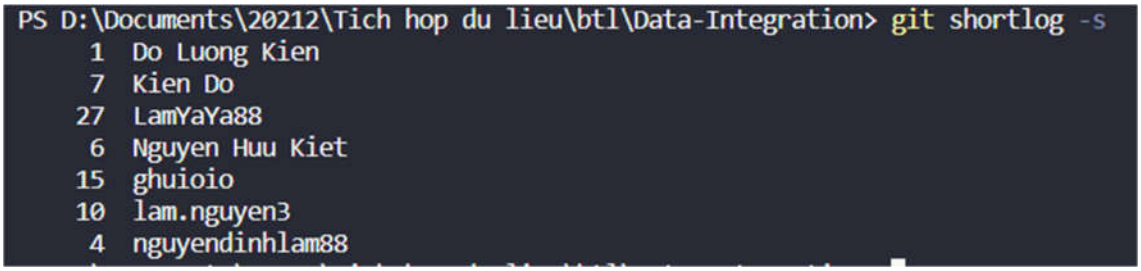
Phân chia công việc

Tên	Công việc	Nhóm trưởng
Phạm Minh Khôi	Crawl data, Kafka, Báo cáo	
Đỗ Lương Kiên	Crawl data, Frontend, Slide, Thuyết trình	
Nguyễn Hữu Kiệt	Crawl data, Data matching, backend	
Nguyễn Đình Lâm	Crawl data, schema matching, hệ thống	x

Thông tin Github

Link github: https://github.com/nguyendinhlam88/Data-Integration
Lúc trước nhóm dùng nhánh main là nhánh chính nhưng sau đó đổi sang nhánh dev. Thêm nữa, bạn Lâm là trưởng nhóm nên commit có thêm việc merge nhánh và các commit resolve conflict.

Thông tin đóng góp trên toàn bộ github



Phạm Minh Khôi	ghuioio	15
Đỗ Lương Kiên	doflamingo0	8
Nguyễn Hữu Kiệt	KietNguyen10112000	6
Nguyễn Đình Lâm	nguyendinhlam88	51