**Câu hỏi tìm hiểu:**

1. **Khi DE làm việc với SE, có thể gặp những khó khăn gì?(anh Cường)**

a. Nhiệm vụ của Data Engineer, Software Engineer và công việc chung trong một dự án:

- Nhiệm vụ chính của Data Engineer: Xây dựng và duy trì hạ tầng dữ liệu, thiết kế pipeline dữ liệu, đảm bảo chất lượng và bảo mật dữ liệu, tối ưu hóa hiệu suất cho việc lưu trữ, truy xuất và xử lý dữ liệu.

- Nhiệm vụ chính của Software Engineer: Phát triển và triển khai ứng dụng, đảm bảo tính ổn định và hiệu năng cho ứng dụng, thu thập phản hồi người dùng và cải tiến ứng dụng.

- Nhiệm vụ chung của Data Engineer và Software Engineer: Tích hợp dữ liệu và ứng dụng, phối hợp trong thiết kế kiến trúc ứng dụng (lưu trữ và xử lý dữ liệu, đáp ứng yêu cầu người dùng), Xử lý vấn đề và tối ưu hóa hệ thống.

b. Những khó khăn trong quá trình làm việc giữa Data Engineer và Software Engineer:

- Nắm bắt các thay đổi trong dữ liệu: Data Engineer cần xác định chính xác các thay đổi trong nguồn dữ liệu gốc, trong khi đó Software Engineer cần đảm bảo ứng dụng thể hiện được chính xác đến từ những thay đổi trong dữ liệu gốc.

- Quản lý lỗi hệ thống: Khi xảy ra lỗi thì Data Engineer và Software Engineer cần xem xét rằng lỗi đến từ dữ liệu gốc hay sai sót trong mã nguồn hệ thống.

- Đồng bộ hoá và tích hợp nguồn dữ liệu: Cần duy trì tính nhất quán giữa các nguồn dữ liệu khác nhau, các hành động để xử lý độ trễ dữ liệu.

- Vấn đề liên quan đến bảo mật: Cần kiểm duyệt chặt chẽ vào nguồn dữ liệu vào, mã hoá thông tin và quyền truy cập cập vào các dữ liệu đó.

- Tối Ưu Hiệu Năng Và Quản Lý Tài Nguyên: Giảm thiểu độ trễ truy vấn, tối ưu hóa cấu trúc lưu trữ và quản lý hiệu quả tài nguyên hệ thống.

1. **Công nghệ trích xuất dữ liệu, ngoài việc viết query( giao tiếp giữa DBMS này sang DBMS khác cần làm gì)(Hậu)**
2. Tổng quan:

Trích xuất dữ liệu (Data Extraction) là quá trình thu thập dữ liệu từ nhiều nguồn khác nhau, chuyển đổi chúng thành định dạng có thể sử dụng cho phân tích, lưu trữ hoặc tích hợp vào các hệ thống khác.

1. Các nguồn dữ liệu phổ biến

- Paper documents (Tài liệu giấy)

- Web pages (Trang web)

- Analog audio/video (Âm thanh/video analog)

- Survey, statistic, economic (Khảo sát, thống kê, kinh tế)

- Transaction data (Dữ liệu giao dịch)

- Social media (Mạng xã hội)

- Weather Station Networks (Mạng lưới trạm thời tiết)

- IoT (Internet of Things - thiết bị kết nối)

- Medical records (Hồ sơ y tế)

- Human genomes (Hệ gen con người)

(c) Công nghệ trích xuất dữ liệu: Dữ liệu có thể được trích xuất bằng nhiều phương pháp khác nhau, tùy thuộc vào loại nguồn và mục tiêu sử dụng. Dưới đây là một số kỹ thuật phổ biến:

- OCR (Optical Character Recognition): Nhận diện ký tự quang học, giúp trích xuất dữ liệu từ tài liệu giấy, hóa đơn, hợp đồng, v.v.

- Web Scraping: Dùng các công cụ như BeautifulSoup, Scrapy, Selenium để thu thập dữ liệu từ website.

- APIs (Application Programming Interfaces): Trích xuất dữ liệu từ hệ thống khác qua các giao diện API (REST, GraphQL, SOAP).

- Database Query: Dùng SQL (Structured Query Language) để trích xuất dữ liệu từ cơ sở dữ liệu như MySQL, PostgreSQL, Oracle.

- Cookies & User Logs: Thu thập hành vi người dùng từ cookies và nhật ký truy cập hệ thống.

- DC, CCD Sampling: Lấy mẫu từ các cảm biến, thiết bị hình ảnh (camera, cảm biến nhiệt).

- Edge Computing: Trích xuất dữ liệu ngay tại thiết bị biên (IoT, cảm biến thời tiết, xe tự hành).

- Biomedical Devices: Trích xuất dữ liệu từ thiết bị y tế như ECG, MRI, xét nghiệm gen.

(d) Giao tiếp giữa 2 DBMS

| **Công nghệ** | **Ứng Dụng Chính** |
| --- | --- |
| Replication | Sao chép dữ liệu giữa hai DBMS |
| ETL | Trích xuất, chuyển đổi và tải dữ liệu giữa DBMS |
| CDC | Đồng bộ chỉ các thay đổi dữ liệu giữa DBMS |
| Database Federation | Truy vấn dữ liệu liên DBMS mà không di chuyển dữ liệu |
| API-based Transfer | Chuyển dữ liệu giữa DBMS thông qua API |

1. **Tìm hiểu về API của Spotify, Facebook (Vy)**

* **API của Facebook:** là nền tảng(platform) mà Facebook cung cấp cho các lập trình viên để có thể dễ dàng trong việc tạo ứng dụng liên kết với tài khoản Facebook và đảm bảo không can thiệp quá sâu vào hệ thống. Nhờ có Facebook API, người tạo ứng dụng có thể lấy thông tin về user, groups, photo ,... trên Facebook mà họ cần. Thông qua API, lập trình viên có thể tích hợp các tính năng của Facebook vào ứng dụng của mình, cung cấp cho người dùng trải nghiệm liền mạch và phong phú hơn
* Các loại Facebook API chính:

1. **Graph API:** Đây là giao diện chính để tương tác với dữ liệu và chức năng của Facebook. Graph API cung cấp một biểu diễn dạng đồ thị của dữ liệu Facebook, cho phép bạn truy cập thông tin về người dùng, trang, nhóm, sự kiện, bài đăng và nhiều hơn nữa.
2. **Marketing API:** API này hỗ trợ các doanh nghiệp trong việc tối ưu hóa và đo lường hiệu quả của các chiến dịch quảng cáo trên Facebook. Nó cho phép bạn tạo, quản lý và theo dõi quảng cáo, cũng như truy xuất thông tin chi tiết về hiệu suất quảng cáo.​
3. **Pages API:** API này cho phép các ứng dụng truy cập và quản lý nội dung cũng như cài đặt của Trang Facebook. Bạn có thể tạo và lấy bài viết, xem bình luận về nội dung trên trang, lấy thông tin chi tiết về trang và cập nhật các hành động mà người dùng có thể thực hiện trên trang.
4. **Conversions API (API Chuyển đổi):** Được thiết kế để tạo kết nối trực tiếp giữa dữ liệu marketing của bạn và hệ thống tối ưu hóa quảng cáo của Meta, giúp cải thiện hiệu quả quảng cáo và đo lường kết quả chính xác hơn.

* Cách thức hoạt động :

Facebook sẽ gửi giao thức POST tới máy chủ của Facebook API. Nó bao gồm một số các thông số theo yêu cầu như api\_key của ứng dụng. Session\_key của người dùng đưa ra yêu cầu. Bên cạnh đó, Facebook còn thêm tham số fb\_sig để thông báo ứng dụng đưa ra yêu cầu. Bằng cách này tất cả các lời gọi API sẽ được đảm bảo an toàn và bảo mật, Facebook có thể xác minh yêu cầu được gửi từ một ứng dụng đã được chấp thuận. Thông tin mà Facebook trả lại là một tài liệu XML, từ đó lập trình viên có thể áp dụng API của Facebook vào ứng dụng của mình.

* Lợi ích của việc sử dụng Facebook API:

1. **Tích hợp tính năng xã hội:** Kết nối ứng dụng của bạn với Facebook giúp tăng tính tương tác và chia sẻ xã hội giữa người dùng.
2. **Tối ưu hóa quảng cáo:** Sử dụng dữ liệu từ Facebook để nhắm mục tiêu chính xác hơn và cải thiện hiệu quả của các chiến dịch quảng cáo.
3. **Truy cập dữ liệu phong phú:** Lấy thông tin về người dùng, trang, sự kiện và nhiều hơn nữa để cung cấp trải nghiệm cá nhân hóa và hấp dẫn hơn.​

* **API của Spotify:** ​Spotify cung cấp Web API, một giao diện lập trình ứng dụng mạnh mẽ cho phép các nhà phát triển tương tác với dịch vụ phát nhạc trực tuyến của họ. Thông qua API này, chúng có thể truy xuất siêu dữ liệu về nghệ sĩ, album, bài hát, quản lý playlist, kiểm soát phát nhạc và nhiều tính năng khác.​
* Các tính năng chính của Spotify Web API:

1. **Truy xuất thông tin nội dung:** Lấy dữ liệu về nghệ sĩ, album, bài hát, playlist, chương trình và tập tin.​
2. **Tìm kiếm nội dung:** Tìm kiếm thông tin trong danh mục của Spotify dựa trên từ khóa
3. **Quản lý thư viện cá nhân:** Thêm hoặc xóa bài hát, album vào thư viện của người dùng.
4. **Kiểm soát phát nhạc:** Điều khiển việc phát nhạc trên các thiết bị của người dùng, như phát, tạm dừng, chuyển bài.
5. **Quản lý playlist:** Tạo, chỉnh sửa và xóa playlist; thêm hoặc xóa bài hát khỏi playlist.

* Cách thức hoạt động:

1. **Đăng ký ứng dụng:** Truy cập [Spotify Developer Dashboard](https://developer.spotify.com/dashboard), đăng nhập bằng tài khoản Spotify của bạn và tạo một ứng dụng mới để nhận Client ID và Client Secret.​[Viblo](https://viblo.asia/p/chia-se-truc-tiep-bai-hat-dang-nghe-len-website-voi-spotify-api-7ymJXPx6Jkq?utm_source=chatgpt.com)
2. **Xác thực và ủy quyền:** Sử dụng OAuth 2.0 để xác thực và nhận access token, cho phép ứng dụng của bạn truy cập vào dữ liệu người dùng.​
3. **Gửi yêu cầu API:** Sử dụng access token để gửi các yêu cầu HTTP đến các endpoint của Spotify Web API và xử lý dữ liệu trả về ở định dạng JSON.
4. **Nghiên cứu về Spark, Kafka (anh Cường)**

a. Kafka - Apache Kafka:

- Khái niệm:

Giải thích đơn giản thì Kafka như một hệ thống bưu điện kỹ thuật số. Giống như một bưu điện xử lý việc nhận, gửi thư từ và gói hàng từ những người gửi đến những người nhận, Kafka quản lý luồng dữ liệu giữa các hệ thống và ứng dụng máy tính. Có thể coi Kafka là nơi thông tin có thể được gửi, lưu trữ và phân phối nhanh chóng mà không mất cấu trúc ban đầu hay bị nghẽn mạch.

Về bản chất, Kafka là một nền tảng truyền phát luồng dữ liệu phân tán với các chức năng quan trọng là: Xuất bản và đăng ký các luồng bản ghi (streams of records), lưu trữ các luồng bản ghi với khả năng chịu lỗi và xử lý các luồng bản ghi theo thời gian thực.

- Cách hoạt động của Kafka:



+ Producers: Như những người gửi thả các gói hàng (dữ liệu) tại bưu điện (Kafka).

+ Broker: Giống như một Cluster, Brokers giống các trung tâm phân phối quản lý toàn bộ quá trình định tuyến, lưu trữ và sao chép dữ liệu trên cluster chung.

+ Topic: Dữ liệu lưu trữ trong Kafka được di chuyển vào các chủ đề riêng biệt giống các tuyến đường bưu chính chuyên dụng. Mỗi Broker chứa nhiều Topic tương tự nhau trên các Broker.

+ Consumers: Như những người nhận và xử lý các gói hàng từ các tuyến đường cụ thể.

- Các Tính Năng Cơ Bản:

+ Hiệu năng cao: Có thể xử lý hàng ngàn thông điệp mỗi giây.

+ Có khả Năng Mở Rộng: Có thể mở rộng một cách liền mạch trên nhiều máy chủ.

+ Tính Bền Vững: Lưu trữ thông điệp trên đĩa, ngăn ngừa thất thoát dữ liệu.

+ Khả Năng Chịu Lỗi: Tiếp tục hoạt động ngay cả khi một số Broker bị lỗi bằng cách các Broker khác sẽ tiếp tục việc lưu trữ và truyền thông tin thay thế cho Broker lỗi.

+ Xử Lý Thời Gian Thực: Cho phép truyền và xử lý dữ liệu ngay lập tức.

- Một số ứng dụng:

+ Event tracking (Theo dõi sự kiện): Các nền tảng mạng xã hội theo dõi tương tác người dùng qua việc ghi lại các lượt nhấp chuột và hành vi người dùng trên trang web hoặc dùng để giám sát sự kiện ứng dụng theo thời gian thực.

+ Log aggregation (Tổng hợp nhật ký): Thu thập nhật ký từ nhiều dịch vụ hoặc nhật ký hệ thống và ứng dụng để hỗ trợ cho việc giám sát và phân tích.

+ Stream processing (Xử lý luồng): Được ứng dụng vào các công việc cần xử lý thời gian thực như: Hệ thống tài chính theo dõi giao dịch chứng khoán hay thu thập và xử lý dữ liệu thiết bị IoT.

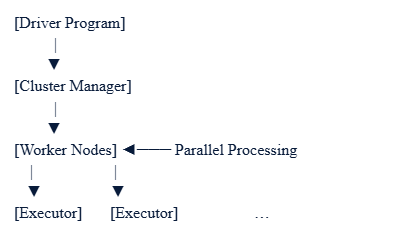
b. Spark - Apache Spark:

- Khái niệm:

Giải thích dễ hiểu thì Spark như một đội ngũ chuyên xử lý dữ liệu có thể đồng thời làm việc trên những câu đố khổng lồ trải rộng trên nhiều bảng lượng dữ liệu rất lớn. Giống như một nhóm chuyên gia có thể chia nhỏ một vấn đề phức tạp thành các phần nhỏ hơn, dễ quản lý và giải quyết nhanh chóng bằng cách làm việc song song, Spark xử lý khối lượng dữ liệu khổng lồ bằng cách phân phối công việc trên nhiều máy tính.

Về bản chất, Spark là một hệ thống điện toán phân tán mã nguồn mở được thiết kế để thực hiện xử lý big data (dữ liệu quy mô lớn) với tốc độ và hiệu quả cao. Nó giống như một trình quản lý dữ liệu thông minh có thể phân tích, chuyển đổi và trích xuất thông tin chi tiết từ các tập dữ liệu khổng lồ gần như theo thời gian thực.

- Cách hoạt động của Spark:



Distributed computing: Spark hoạt động như một nhóm công nhân được điều phối một cách có tổ chức với:

+ Một Driver Program (chương trình điều khiển) có vai trò phân công nhiệm vụ.

+ Nhiều Worker Nodes (nút xử lý) có vai trò xử lý các Executors (trình thực thi) đồng thời.

=> Kết quả sẽ được thu thập và kết hợp một cách hiệu quả.

In-Memory processing: Không giống như các hệ thống truyền thống phải đọc và ghi dữ liệu trên đĩa nhiều lần, Spark giữ dữ liệu trong bộ nhớ máy tính (RAM), làm giảm đáng kể thời gian xử lý.

- Các Tính Năng Cơ Bản:

+ Tốc độ cao: Nhanh hơn rất nhiều lần so với Hadoop MapReduce truyền thống trong một số trường hợp xử lý nhất định, đồng thời giảm thiểu các thao tác nhập, xuất dữ liệu.

+ Hỗ trợ nhiều ngôn ngữ lập trình: Scala (ngôn ngữ gốc), Python, Java, R,...

+ Cung cấp các thư viện nâng cao cũng là thành phần chính của Spark như Spark SQL (xử lý dữ liệu có cấu trúc), Spark Streaming (phân tích dữ liệu thời gian thực), MLlib ( hỗ học machine learning), GraphX (xử lý đồ thị).

+ Khả năng chịu lỗi: Tự động phục hồi từ các Worker Nodes và duy trì tính nhất quán dữ liệu trên các hệ thống phân tán.

- Một số ứng dụng:

+ Business Intelligence: Giúp phân tích hành vi khách hàng, dự đoán xu hướng thị trường và đánh giá rủi ro tài chính theo thời gian thực.

+ Scientific Research: Có thể ứng dụng trong một số nghiên cứu về phân tích dữ liệu gen, nghiên cứu vật lý hạt hay xử lý dữ liệu thiên văn.

+ Machine Learning and AI: Đào tạo các mô hình học máy quy mô lớn, xử lý ngôn ngữ tự nhiên, các thuật toán nhận dạng hình ảnh,...

c. So sánh Kafka và Spark:

| *Đặc điểm* | *Apache Kafka* | *Apache Spark* |
| --- | --- | --- |
| Mục đích sử dụng chính | Trao đổi thông tin trong thời gian thực | Xử lý và phân tích các vấn đề về big data |
| Xử lý dữ liệu | Real-time event streaming (Xử lý dữ liệu thời gian thực) | Batch and stream processing (Xử lý theo lô và theo luồng) |
| Hiệu năng | Truyền phát thông tin hiệu năng cao | Xử lý tính toán tốc độ cao |
| Khả năng lưu trữ | Lưu trữ tin nhắn tạm thời | Can persist and transform data |
| Ngôn ngữ lập trình | Chủ yếu là Java | Rộng rãi (Scala, Python, Java, R,…) |
| Khả năng tính toán phức tạp | Hạn chế | Nhiều lĩnh vực (học máy, xử lý đồ thị) |

1. **Thách thức: dữ liệu nguồn bị thay đổi ảnh hưởng như thế nào đến downstream(Vy)**

* **Ảnh hưởng đến Data Pipeline**
* ETL Process bị gián đoạn:
* Nếu cấu trúc dữ liệu thay đổi(thêm,xóa ,sửa cột), quá trình ETL có thể gặp lỗi, khiến dữ liệu không được cập nhật đúng cách

Ví dụ: Nếu API Facebook thay đổi format trả về hoặc xóa một trường dữ liệu, pipeline của Data Engineer có thể không hoạt động đúng.

* Lỗi Schema Evolution:
* Nếu database nguồn thay đổi schema (ví dụ: đổi kiểu dữ liệu từ int sang string), các hệ thống downstream như Data Warehouse có thể không xử lý được.

Ví dụ: Một cột price ban đầu là số nguyên (INTEGER) nhưng bị chuyển thành chuỗi (VARCHAR), khiến các phép tính tổng, trung bình trên hệ thống phân tích bị sai.

* Mất dữ liệu:
* Nếu upstream xóa hoặc lọc dữ liệu trước khi đẩy xuống downstream mà không có thông báo, có thể gây mất mát dữ liệu quan trọng.

Ví dụ: Nếu API Spotify ngừng cung cấp dữ liệu về số lượt phát (play\_count), các báo cáo phân tích lượt nghe sẽ bị thiếu dữ liệu.

* **Ảnh hưởng đến Hệ thống Phân tích( BI, Dashboard)**
* Dữ liệu lỗi hoặc không đầy đủ
* Nếu dữ liệu nguồn bị lỗi hoặc gián đoạn, các biểu đồ, báo cáo trong Power BI, Tableau, Looker có thể hiển thị sai.  
  Ví dụ: Nếu dữ liệu doanh thu từ đơn hàng bị thiếu, báo cáo doanh thu tháng có thể hiển thị giảm sút không chính xác.
* Không đồng bộ giữa các hệ thống
* Khi dữ liệu ở một số bảng được cập nhật nhưng bảng khác thì không, có thể gây ra sự mâu thuẫn

Ví dụ: Nếu bảng Orders cập nhật đơn hàng mới nhưng bảng Customers không cập nhật thông tin khách hàng, các báo cáo phân tích hành vi khách hàng sẽ bị sai.

* **Ảnh hưởng đến hệ thống AI/ ML**
* Mô hình dự đoán sai
* Nếu dữ liệu training thay đổi format hoặc bị lỗi, mô hình Machine Learning có thể dự đoán sai.

Ví dụ: Nếu hệ thống đề xuất nhạc của Spotify dùng dữ liệu tương tác bị sai lệch, gợi ý bài hát sẽ không chính xác.

* Trôi dữ liệu( Drift)
* Nếu dữ liệu mới có phân phối khác so với dữ liệu cũ, mô hình AI có thể hoạt động kém hiệu quả.

Ví dụ: Nếu trước đây dữ liệu khách hàng chủ yếu từ châu Á nhưng bây giờ có nhiều khách từ châu Âu mà không điều chỉnh mô hình, kết quả phân tích sẽ không còn chính xác.

* **Ảnh hưởng đến Ứng dụng người dùng cuối**
* Lỗi giao diện người dùng:
* Nếu API thay đổi mà frontend không cập nhật, người dùng có thể thấy lỗi hoặc dữ liệu hiển thị sai.

Ví dụ: Nếu API Facebook đổi format dữ liệu ảnh đại diện, ứng dụng hiển thị avatar có thể bị vỡ.

* Lỗi tích hợp dịch vụ:
* Nếu API của bên thứ ba thay đổi, các ứng dụng tích hợp có thể bị lỗi hoặc ngừng hoạt động.

Ví dụ: Nếu API thanh toán đổi phương thức xác thực mà backend không cập nhật, quá trình thanh toán có thể bị lỗi.

1. **Tìm hiểu về Grab engineer (Hậu)**
2. Tổng quan

- Các doanh nghiệp lớn trong lĩnh vực công nghệ ( **Google** với Google Developers & Google Research; **Meta** với Meta Engineering; **Microsoft** với Microsoft Engineering Blog,..) thường duy trì các blog kỹ thuật để

* Xây dựng thương hiệu kỹ thuật (Tech Branding)
* Thu hút và tuyển dụng nhân tài (Tech Talent Recruitment)
* Chia sẻ kiến thức & mã nguồn mở
* Xây dựng quan hệ với đối tác & khách hàng B2B
* Minh bạch về thuật toán & chính sách.

- Grab Engineering là blog kỹ thuật của Grab.

- Giới thiệu ngắn về Grab: Grab là một công ty công nghệ có trụ sở tại Singapore, hoạt động trong lĩnh vực dịch vụ gọi xe, giao hàng, tài chính và công nghệ số tại Đông Nam Á.

- Theo báo cáo của công ty nghiên cứu thị trường Mordor Intelligence thị trường gọi xe công nghệ tại Việt Nam năm 2023 có quy mô 727,73 triệu USD, trong đó Grab chiếm đến 58,68% thị phần.

1. Grab Engineering (engineering.grab.com):

- Là nền tảng web công nghệ của Grab với những bài viết đầu tiên xuất hiện từ rất sớm (28/12/2025) .

- Các chủ đề chia sẻ chính:

+ **Engineering** – Các bài viết về kỹ thuật phần mềm, kiến trúc hệ thống, DevOps, CI/CD, hiệu suất ứng dụng, v.v.

**+ Data Science** – Phân tích dữ liệu, AI/ML, xử lý dữ liệu lớn, triển khai mô hình AI/ML, tối ưu hóa thuật toán, v.v

+ **Design** – Trải nghiệm người dùng (UX/UI), thiết kế sản phẩm số, nghiên cứu người dùng, v.v.

+ **Product** – Quản lý sản phẩm, tối ưu hóa quy trình phát triển, cải tiến sản phẩm dựa trên dữ liệu, v.v.

+ **Security** – Bảo mật hệ thống, bảo vệ dữ liệu người dùng, chống gian lận, bảo vệ API, v.v.

- Ở footer có các thông tin trang web github, linked in, grab.careers (tuyển dụng) dành cho các bạn nếu muốn đồng hành cùng Grab.

**A2: Từ A1 : (anh Sinh)**

1. **Phân biệt các loại dữ liệu: Structured, unstructured, semi-structured trong case A1**

* Structured: thông tin người dùng (mã số, họ tên, ngày sinh, tuổi, email, điện thoại, địa chỉ, ngày đăng ký), nhà hàng (mã số, tên, địa chỉ, số sao trung bình, giờ mở/đóng cửa, trạng thái), món ăn (mã số, tên, mã nhà hàng, giá, danh mục), đơn hàng (mã đơn, mã khách, mã nhà hàng, tổng giá, trạng thái, thời gian), chi tiết đơn hàng (mã số, mã đơn hàng, mã món, số lượng, giá), thanh toán (mã khách, mã đơn hàng, phí ship, voucher, giá phải trả, đã thanh toán, hình thức thanh toán), gợi ý món ăn (mã gợi ý, mã khách, mã món, lý do gợi ý) được lưu ở dạng bảng (có lược đồ quan hệ giữa các bảng)
* Unstructured: hình ảnh nhà hàng, món ăn, đánh giá, bình luận của khách hàng dưới dạng văn bản, dữ liệu tìm kiếm bằng văn bản của người dùng, đồ thị biểu diễn mức độ tương đồng giữa các món ăn dùng cho thuật toán gợi ý theo thói quen người dùng
* Semi-structured: dữ liệu log theo dõi hành vi người dùng gồm mã khánh hàng, các hành vi (thời gian, hoạt động (search, view, cart abandonment, order), mã nhà hàng, mã món ăn) (ở dạng json), dữ liệu phản hồi của khách hàng về một nhà hàng, món ăn bao gồm mã khách hàng, mã nhà hàng, mã món ăn, đánh giá (số sao, văn bản nhận xét, tích cực/tiêu cực) (ở dạng xml/json)

1. **Phân biệt các nguồn dữ liệu: OLTP, external, internal. Phân cấp mức độ bảo mật**

* OLTP: hệ thống xử lý giao dịch trực tuyến. Hướng tới xử lý các lượng lớn dữ liệu giao dịch real-time, đảm bảo tính ACID (nguyên tử, nhất quán, độc lập, bền vững) và hiệu suất cao (tốc độ xử lý, khả năng mở rộng) của việc thêm, sửa, xóa dữ liệu từ CSDL. Dữ liệu có cấu trúc, lưu trữ trong cơ sở dữ liệu quan hệ (SQL), cập nhật liên tục. OLTP xử lý, lưu trữ thông tin người dùng, đơn hàng, nhà hàng, thanh toán. Ví dụ: khi người dùng đặt món, hệ thống OLTP xử lý đơn hàng ngay lập tức, khi thanh toán thành công, dữ liệu sẽ được ghi vào bảng Thanh toán trong CSDL. Dữ liệu này rất nhạy cảm, nếu bị rò rỉ sẽ ảnh hưởng đến uy tín doanh nghiệp hoặc vi phạm luật bảo mật, chỉ người có quyền hạn mới được truy cập vì nó lưu trữ hoạt động của khách hàng và dữ liệu thanh toán.
* External data là dữ liệu đến từ nguồn bên ngoài hệ thống, không thuộc quyền kiểm soát của ứng dụng. Có thể ở dạng có cấu trúc, bán cấu trúc hoặc phi cấu trúc. Cần tích hợp thông qua API, web scraping hoặc nhập dữ liệu thủ công. Ví dụ: API thời tiết để gợi ý món ăn theo thời tiết, dữ liệu mạng xã hội để bắt kịp xu hướng ẩm thực, dữ liệu của đối thủ, dữ liệu Google Maps API sử dụng để định vị nhà hàng, tính toán thời gian giao hàng, gợi ý món ăn ở nhà hàng gần nhất. Dữ liệu này công khai, ai cũng có quyền truy cập.
* Internal data là dữ liệu được thu thập và sở hữu nội bộ hệ thống, không đến từ bên ngoài. Bao gồm dữ liệu có cấu trúc (khách hàng, món ăn, nhà hàng, hóa đơn, thanh toán), bán cấu trúc (hành vi người dùng, phản hồi dịch vụ) hoặc phi cấu trúc (hình ảnh món ăn, đánh giá bằng văn bản, đồ thị tương đồng giữa các món ăn), sử dụng cho phân tích hành vi khách hàng, tối ưu trải nghiệm người dùng. Dữ liệu tối mật vì chỉ có một số ít người chịu trách nhiệm mới có quyền truy cập, bao gồm mật khẩu của người dùng, API key của hệ thống thanh toán., dữ liệu chiến lược kinh doanh, thuật toán AI gợi ý món ăn, nếu bị lộ gây nguy cơ thiệt hại nghiêm trọng về tài chính, pháp lý.

1. **Khi nào thì gọi APIs ? khi nào sử dụng Web scraping**

* Gọi APIs trong hệ thống khi cần trao đổi dữ liệu giữa các thành phần của ứng dụng, từ bên ngoài khi cần lấy dữ liệu công khai có hỗ trợ API. API có độ chính xác cao, cập nhật thường xuyên, request nhanh, bảo mật, kiểm soát tốt, không cần phân tích cú pháp (parsing) HTML phức tạp như Web Scraping. Nhưng cũng có nhược điểm là một số API yêu cầu trả phí để truy cập đầy đủ và các API thường giới hạn số lần gọi.
* Web scraping khi hệ thống bên ngoài không cung cấp API, có nhu cầu tổng hợp thông tin từ nhiều website và không bị giới hạn số lần gọi. Nhược điểm là có thể vi phạm điều khoản của website, dữ liệu gặp lỗi nếu website thay đổi giao diện, tốc độ cũng chậm hơn so với gọi API, vì phải tải toàn bộ trang web và phân tích cú pháp (parsing) HTML.

1. **Dữ liệu có liên quan gì đến 5V của Big Data:**

-Big Data được đặc trưng bởi 5 yếu tố chính, gọi là 5V:

* Volume (Khối lượng): Big Data có kích thước cực lớn, có thể lên đến petabyte hoặc exabyte.
* Velocity (Tốc độ): Dữ liệu được tạo ra và xử lý với tốc độ rất nhanh theo thời gian (real time) thực hoặc gần thời gian thự(near real time).
* Variety (Đa dạng): Dữ liệu có nhiều định dạng khác nhau: có cấu trúc (structured), bán cấu trúc (semi-structured) và phi cấu trúc (unstructured).
* Veracity (Tính xác thực): Cần làm sạch dữ liệu, loại bỏ nhiễu để đảm bảo độ tin cậy cho phân tích.
* Value (Giá trị): Mục tiêu cuối cùng của Big Data là tạo ra giá trị từ dữ liệu.
* Vai trò của 5V đã khẳng định tiềm năng to lớn của hệ thống trong việc trở thành một nền tảng Big Data
* Volume: GrabFood là hệ thống với hàng triệu người dùng và không ngừng tăng lên do mức sống và nhu cầu tăng cao nên số lượng dữ liệu được tạo và lưu trữ là vô cùng khổng lồ đặt biệt là các dữ liệu unstructured ( hình ảnh, đánh giá, phản hồi, dữ liệu tìm kiếm,..) và semi structured(log hành vi khách hàng, dữ liệu phản hồi,...)
* Variety: thể hiện rõ ở việc dữ liệu cung cấp cho Data Scientist có đầy đủ định dạng: Structured, Unstructured, Semi-structured.
* Velocity (Tốc độ sinh dữ liệu): GrabFood là hệ thống có cả OLTP và OLAP nhưng chủ yếu vẫn là OLTP bên cạnh đó kết hợp với dữ liệu đa dạng và khối lượng người dùng khổng lồ nên hệ thống cần có khả năng xử lý tốc độ cao, cần cập nhật theo thời gian thực.
* Veracity: Dữ liệu đa dạng, từ nhiều nguồn nên có nhiều thông tin gây nhiễu, chưa phù hợp nên cần làm sạch, chuẩn hóa trước khi tiến hành phân tích.
* Value: Dữ liệu giúp tạo nền tảng cho cải thiện hệ thống gợi ý và phân tích hành vi đặt hàng từ đó giúp tăng doanh số, thúc đẩy người dùng đặt hàng.
* Một số ví dụ về 5V:
* Volume (Khối lượng)
* Facebook xử lý khoảng **4 petabyte dữ liệu mỗi ngày**, bao gồm bài đăng, bình luận, hình ảnh và video.
* Các hệ thống IoT trong thành phố thông minh thu thập **hàng tỷ dữ liệu** từ cảm biến giao thông, thời tiết, và năng lượng mỗi ngày.
* Velocity ( Tốc độ )
* Sàn chứng khoán NASDAQ xử lý hàng **triệu giao dịch trong một giây**.
* Các ứng dụng như Google Maps cập nhật **tình trạng giao thông theo thời gian thực** từ dữ liệu GPS của người dùng.
* Variety ( Đa dạng)
* Một bệnh viện có dữ liệu từ nhiều nguồn khác nhau: Dữ liệu có cấu trúc: Hồ sơ bệnh nhân (SQL), Dữ liệu bán cấu trúc: Kết quả xét nghiệm lưu dưới dạng XML/JSON, Dữ liệu phi cấu trúc: Ảnh X-ray, MRI, ghi âm hội thoại của bác sĩ.
* YouTube xử lý video, bình luận văn bản, âm thanh và metadata liên quan.
* Veracity ( Độ tin cậy )
* Các công ty tài chính cần **phát hiện giao dịch gian lận**, vì có thể có dữ liệu không chính xác từ lỗi nhập liệu hoặc hacker.
* Fake news lan truyền trên mạng xã hội có thể gây hiểu lầm, vì vậy cần **AI kiểm tra độ xác thực của thông tin**.
* Value (Giá trị)
* Netflix sử dụng dữ liệu xem phim của người dùng để **đề xuất nội dung phù hợp**, giúp tăng tỷ lệ giữ chân khách hàng.
* Các siêu thị như Walmart phân tích dữ liệu mua sắm để **dự đoán nhu cầu**, từ đó tối ưu hóa kho hàng.

**Key word:**

1. **External, internal (Hậu)**
2. Internal Data (Dữ liệu nội bộ)

Là dữ liệu được thu thập từ bên trong tổ chức, doanh nghiệp. Đây là dữ liệu mà công ty sở hữu và có thể sử dụng để ra quyết định kinh doanh.

Ví dụ về Internal Data:

* Dữ liệu bán hàng: Số lượng sản phẩm bán ra, doanh thu theo tháng/quý.
* Dữ liệu khách hàng: Thông tin khách hàng, lịch sử mua hàng.
* Dữ liệu nhân sự: Lương, hiệu suất làm việc của nhân viên.
* Dữ liệu sản xuất: Chi phí nguyên vật liệu, số lượng sản phẩm tồn kho.

1. External Data (Dữ liệu bên ngoài)

Là dữ liệu được thu thập từ các nguồn bên ngoài tổ chức, không do doanh nghiệp tự tạo ra nhưng có thể được sử dụng để phân tích và đưa ra quyết định.

Ví dụ về External Data:

* Dữ liệu thị trường: Xu hướng tiêu dùng, mức giá trung bình của đối thủ.
* Dữ liệu kinh tế: Lãi suất ngân hàng, tỷ giá hối đoái, lạm phát.
* Dữ liệu mạng xã hội: Đánh giá của khách hàng về sản phẩm trên Facebook, Twitter.
* Dữ liệu thời tiết: Ảnh hưởng đến chuỗi cung ứng hoặc hành vi mua hàng theo mùa.

(c) Sự khác nhau

| **Tiêu chí** | **Internal Data** | **External Data** |
| --- | --- | --- |
| Nguồn gốc | Từ nội bộ doanh nghiệp | Từ bên ngoài doanh nghiệp |
| Độ kiểm soát | Doanh nghiệp có toàn quyền sở hữu và kiểm soát | Doanh nghiệp không kiểm soát trực tiếp |
| Chi phí thu thập | Thường có sẵn, ít tốn chi phí | Có thể cần mua hoặc thuê từ bên thứ ba |
| Tính cập nhật | Có thể cập nhật liên tục theo thời gian thực | Có thể bị chậm trễ tùy thuộc vào nguồn cung cấp |
| Ứng dụng | Phân tích hiệu suất nội bộ, tối ưu vận hành | Đánh giá thị trường, dự báo xu hướng |

1. **CDC, material view (Vy)**

* **CDC:**
* Khái niệm: là một quá trình theo dõi và xác định những thay đổi trong dữ liệu từ nhiều nguồn dữ liệu khác nhau như cơ sở dữ liệu và kho dữ liệu, nhằm đảm bảo tính toàn vẹn và nhất quán của dữ liệu giữa các hệ thống khác nhau.
* Phương pháp thực hiện CDC:
* **Sử dụng cột dấu thời gian (timestamp):** Thêm một cột trong bảng để ghi lại thời điểm thay đổi cuối cùng của mỗi hàng. Bằng cách này, các hàng có dấu thời gian mới hơn lần kiểm tra trước đó được coi là đã thay đổi.
* **Sử dụng số phiên bản (versioning):** Mỗi hàng trong bảng được gán một số phiên bản. Khi dữ liệu thay đổi, số phiên bản được tăng lên, giúp xác định các thay đổi dựa trên sự khác biệt về số phiên bản.
* **Sử dụng triggers trong cơ sở dữ liệu:** Thiết lập triggers để ghi lại các sự kiện thay đổi vào một bảng hàng đợi, sau đó có thể được xử lý để phản ánh các thay đổi trong hệ thống đích.
* **Đọc log giao dịch:** Phân tích log giao dịch của cơ sở dữ liệu để xác định các thay đổi, sau đó áp dụng chúng vào hệ thống đích. Phương pháp này thường ít ảnh hưởng đến hiệu suất của cơ sở dữ liệu nguồn.
* Lợi ích của CDC:
* **Đồng bộ hóa dữ liệu theo thời gian thực:** Giúp các hệ thống luôn cập nhật với những thay đổi mới nhất, đảm bảo tính nhất quán của dữ liệu.
* **Tích hợp dữ liệu hiệu quả:** Cho phép tích hợp dữ liệu từ nhiều nguồn khác nhau mà không cần tải lại toàn bộ dữ liệu, giảm tải cho hệ thống và tăng hiệu suất.
* **Hỗ trợ phân tích và báo cáo:** Cung cấp dữ liệu cập nhật liên tục, hỗ trợ các hoạt động phân tích và ra quyết định dựa trên dữ liệu mới nhất.
* Thách thức khi triển khai:
* **Phức tạp trong việc thiết lập:** Tùy thuộc vào hệ quản trị cơ sở dữ liệu và cấu trúc hệ thống, việc triển khai CDC có thể đòi hỏi kỹ thuật phức tạp.
* **Ảnh hưởng đến hiệu suất:** Một số phương pháp CDC, như sử dụng triggers, có thể ảnh hưởng đến hiệu suất của cơ sở dữ liệu nguồn nếu không được cấu hình cẩn thận.
* **Xử lý dữ liệu lớn:** Với các hệ thống có lưu lượng dữ liệu thay đổi lớn, việc theo dõi và xử lý các thay đổi có thể gặp khó khăn và đòi hỏi hạ tầng mạnh mẽ.
* Materialized View

- Khái niệm: ​Materialized View (MV) là một đối tượng trong hệ quản trị cơ sở dữ liệu, lưu trữ kết quả của một truy vấn SQL dưới dạng bảng vật lý. Khác với View thông thường (là bảng ảo không lưu trữ dữ liệu), Materialized View giữ lại dữ liệu đã truy vấn, giúp tăng tốc độ truy xuất thông tin, đặc biệt hữu ích trong các hệ thống yêu cầu hiệu năng cao hoặc xử lý dữ liệu phức tạp.

- Đặc điểm chính:

* **Lưu trữ vật lý:** Kết quả của truy vấn được lưu trữ trực tiếp trên đĩa cứng, cho phép truy xuất dữ liệu nhanh hơn so với View thông thường, vốn phải tính toán lại mỗi khi được gọi.
* **Tiêu tốn không gian lưu trữ:** Do lưu trữ dữ liệu thực tế, Materialized View yêu cầu thêm không gian bộ nhớ để duy trì thông tin đã truy vấn. ​
* **Cần làm mới (refresh) dữ liệu:** Dữ liệu trong Materialized View có thể trở nên lỗi thời nếu không được cập nhật thường xuyên. Việc làm mới có thể được thực hiện theo lịch trình hoặc khi cần thiết, tùy thuộc vào yêu cầu của hệ thống.
* Ưu điểm:
* **Tăng tốc độ truy vấn:** Do dữ liệu đã được tính toán và lưu trữ sẵn, việc truy xuất thông tin từ Materialized View nhanh hơn đáng kể so với việc thực hiện lại truy vấn gốc, đặc biệt với các truy vấn phức tạp hoặc tổng hợp dữ liệu từ nhiều bảng.
* **Giảm tải cho hệ thống:** Bằng cách sử dụng Materialized View, hệ thống có thể giảm số lượng truy vấn trực tiếp đến các bảng cơ sở, giúp tối ưu hóa hiệu suất và giảm tải cho cơ sở dữ liệu.
* Thách thức:
* **Cập nhật dữ liệu:** Dữ liệu trong Materialized View không tự động cập nhật khi dữ liệu trong các bảng nguồn thay đổi. Do đó, cần có cơ chế làm mới để đảm bảo tính nhất quán và độ chính xác của thông tin. ​
* **Tiêu tốn tài nguyên lưu trữ:** Việc lưu trữ kết quả truy vấn đòi hỏi không gian bộ nhớ bổ sung, điều này có thể là một yếu tố cần cân nhắc trong các hệ thống có hạn chế về tài nguyên.
* Ứng dụng:
* **Hệ thống báo cáo và phân tích:** Trong các hệ thống yêu cầu báo cáo nhanh chóng và hiệu quả, Materialized View giúp cung cấp dữ liệu đã được tổng hợp sẵn, giảm thời gian chờ đợi và tăng trải nghiệm người dùng.
* **Kho dữ liệu (Data Warehouse):** Materialized View thường được sử dụng để lưu trữ các tập dữ liệu đã được tổng hợp hoặc chuyển đổi, hỗ trợ cho các truy vấn phân tích phức tạp mà không ảnh hưởng đến hiệu suất của hệ thống chính.
* Lưu ý khi sử dụng:
* **Lên kế hoạch làm mới dữ liệu:** Xác định tần suất và phương pháp làm mới phù hợp để cân bằng giữa độ chính xác của dữ liệu và hiệu suất hệ thống.
* **Quản lý không gian lưu trữ:** Theo dõi và quản lý không gian bộ nhớ để đảm bảo rằng việc sử dụng Materialized View không ảnh hưởng tiêu cực đến tài nguyên hệ thống.

1. **Data brick (Vy)**

* Data brick là gì?

Databricks là một giải pháp toàn diện dựa trên đám mây, và duy nhất có thể xử lý mọi yêu cầu về quản lý và phân tích dữ liệu. Hiểu đơn giản là tất cả nhóm dữ liệu của bạn được cộng tác trên đó. Ngoài ra, với khả năng xử lý dữ liệu ở quy mô lớn – Databricks không những mang lại hiệu suất cao mà còn đảm bảo tính hiệu quả về chi phí.

Được hỗ trợ trên trên các dịch vụ đám mây hàng đầu như Microsoft Azure, Amazon Web Services (AWS) và Google Cloud – Databricks cung cấp sự linh hoạt tối đa, cho phép doanh nghiệp dễ dàng tích hợp và mở rộng trên nền tảng đám mây sẵn có hoặc thậm chí kết hợp nhiều dịch vụ đám mây, đáp ứng nhu cầu đa dạng của các tổ chức.

* Các thành phần của Databrick

1. **Databases & Tables** – Quản lý dữ liệu theo mô hình bảng, tương tự DataFrame.
2. **Libraries** – Hỗ trợ thư viện bên thứ ba cho Spark, có thể cài đặt ở cấp notebook hoặc cụm.
3. **Jobs** – Tự động hóa thực thi notebook hoặc tệp JAR trên Spark theo lịch trình.
4. **Notebook** – Môi trường viết mã đa ngôn ngữ (Python, SQL, Scala, R, Java).
5. **Workspace** – Quản lý tài nguyên như notebook, thư viện, bảng điều khiển trực quan.
6. **Cluster** – Cụm Spark để xử lý dữ liệu, gồm cụm tương tác và cụm công việc.

* Các tính năng nổi bật:

1. Databricks Production:

Bạn có thể tổ chức một quy trình làm việc hoàn chỉnh bằng cách sắp xếp các notebook. Những quy trình này được triển khai như các công việc Spark và sử dụng lịch công việc.

Qua đó, bạn có thể theo dõi và kiểm soát chúng, từ việc kiểm tra nhật ký và thiết lập cảnh báo linh hoạt. Ngoài việc tương tác với dữ liệu, bạn cũng có thể chuyển chúng vào môi trường sản xuất để tiết kiệm thời gian và nâng cao hiệu suất.

2. Interactive workspace (Môi trường làm việc hợp tác)

Trong môi trường này, bạn có thể khám phá, tương tác và phân tích dữ liệu một cách linh hoạt. Tương tự như làm việc trên một tệp Excel, bạn có thể áp dụng công thức và xem kết quả ngay lập tức. Ngoài ra, bạn cũng có thể thực hiện các phép tính phức tạp và xem kết quả ngay trong không gian làm việc.

Interactive workspace cho phép nhiều người viết mã trong cùng một notebook. Đồng thời họ có thể theo dõi thay đổi của các mã và di chuyển chúng vào quản lý mã nguồn sau khi hoàn thành. Các tập dữ liệu đã được xử lý sẽ được tổ chức trong bảng điều khiển, để giám sát hệ thống hoặc phục vụ cho người dùng cuối.

3. Databricks Workspace

Trong Databricks Workspace được tạo ra 2 phần riêng biệt gồm:

Không gian làm việc tương tác, được thiết kế để thao tác và thử nghiệm các công việc.

Databricks Production, được sử dụng để chạy và quản lý các công việc Spark trong môi trường sản xuất

Đây là 2 phần cốt lõi của nền tảng Databricks, mang lại sự linh hoạt và hiệu suất cho các dự án phân tích dữ liệu

4. Tự động hóa các tác vụ

Thực thi các tác vụ và Tự động hóa là yếu tố quan trọng trong việc xử lý dữ liệu, giúp tránh lỗi con người và giảm thiểu thời gian triển khai.

Sau quá trình khám phá và xây dựng đường ống dữ liệu, tính năng này sẽ tự động thực hiện kế hoạch theo yêu cầu của người dùng, bằng cách lên lịch tự động thực thi

5. Bảo mật tích hợp

Bảo mật là một phần không thể thiếu của mọi công cụ liên quan đến dữ liệu. Databricks cung cấp sẵn tính năng kiểm soát truy cập và bảo mật cấp doanh nghiệp, tuân thủ các tiêu chuẩn bảo mật hệ thống cao. Điều này giúp bạn triển khai ứng dụng của mình một cách an toàn trên nền tảng Databricks.

6. Quản lý Cơ sở hạ tầng

Ngoài các tính năng thông thường của Spark, Databricks cung cấp một loạt tính năng quản lý hạ tầng giúp đơn giản hóa quy trình triển khai và quản lý cụm.

1. **Hadoop(A Cường)**

a. Giới thiệu sơ lược: Hadoop là một Apache framework mã nguồn mở cho phép phát triển các ứng dụng phân tán (distributed processing) để lưu trữ và quản lý các tập dữ liệu lớn. Hadoop hiện thực mô hình MapReduce, mô hình mà ứng dụng sẽ được chia nhỏ ra thành nhiều phân đoạn khác nhau được chạy song song trên nhiều node khác nhau (hay nhiều máy khác nhau).

b. Các chức năng chính của Hadoop: Nếu ví Hadoop như một hệ thống thư viện thông minh khổng lồ thì nó có thể thực hiện các nhiệm vụ sau:

- Phân phối sách trên nhiều phòng và nhanh chóng truy xuất thông tin từ bất kỳ khu vực nào của thư viện: Hệ thống tập tin phân tán của Hadoop HDFS (Hadoop Distributed File System) là chia các tệp lớn thành các khối nhỏ hơn và lưu trữ trên nhiều máy khác nhau, giúp cho việc truy xuất dữ liệu nhanh chóng và có khả năng chịu lỗi của dữ liệu.

- Cho phép nhiều thủ thư làm việc đồng thời và tự động thay thế sách: Xử lý các tập dữ liệu khổng lồ song song bằng cách chia vấn đề phức tạp thành các tác vụ nhỏ và tổng hợp kết quả từ các tác vụ riêng lẻ đó.

c. Một số ưu điểm và ứng dụng:

- Ưu điểm: Dễ dàng mở rộng, linh hoạt với nhiều loại dữ liệu và có khả năng chịu lỗi với hiệu quả chi phí thấp do hoạt động trên phần cứng thông thường mà không cần các hệ thống phức tạp.

- Ứng dụng: Phân tích hành vi người dùng trên trình duyệt web, sử dụng trong nghiên cứu khoa học, phân tích hành vi khách hàng và tài chính.

d. Một số hạn chế: Cài đặt phức tạp, không phù hợp cho các vấn đề xử lý thời gian thực,...

1. **Map reduce (Hậu)**

* MapReduce là mô hình được thiết kế độc quyền bởi Google, nó có khả năng lập trình xử lý các tập dữ liệu lớn song song và phân tán thuật toán trên 1 cụm máy tính. MapReduce trở thành một trong những thành ngữ tổng quát hóa trong thời gian gần đây.
* MapReduce hoạt động dựa trên nguyên tắc “chia để trị”

1. Bước Map:

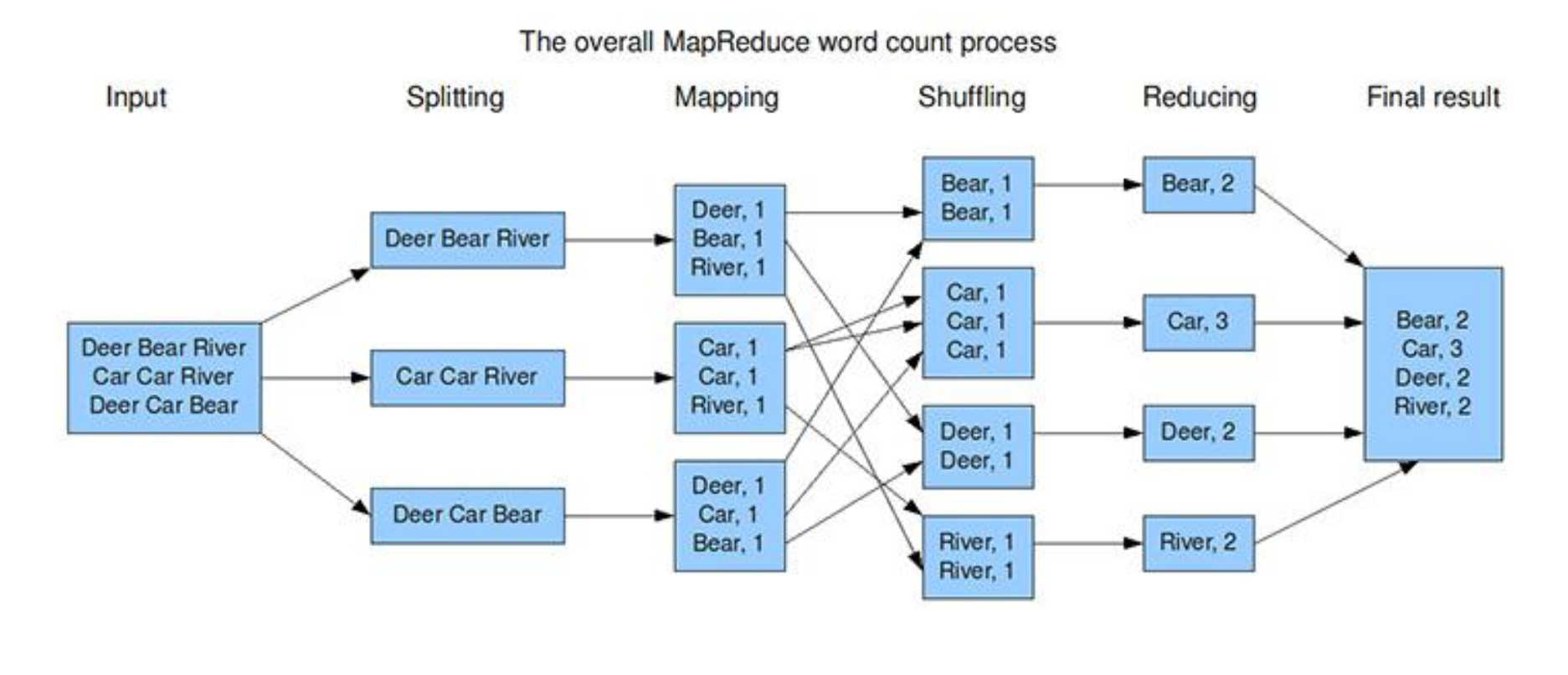
* Chia nhỏ dữ liệu thành nhiều phần.
* Xử lý từng phần độc lập.
* Xuất ra các cặp (key, value).

2. Bước Shuffle & Sort (bước đệm):

* Shuffle: Gom nhóm dữ liệu có cùng key.
* Sort: Sắp xếp lại dữ liệu để dễ xử lý.

3. Bước Reduce:

* Nhận dữ liệu đã nhóm từ Shuffle & Sort.
* Xử lý tổng hợp và tạo kết quả cuối cùng
* Ví dụ:



Nguồn: <https://itnavi.com.vn/blog/mapreduce-nhung-uu-diem-va-cach-thuc-hoat-dong-cua-nen-tang-nay>

- Kết luận: MapReduce là một mô hình mạnh mẽ giúp xử lý dữ liệu lớn một cách hiệu quả bằng cách chia nhỏ công việc, xử lý song song và tổng hợp kết quả, giúp tối ưu hóa tốc độ và tài nguyên trong các hệ thống dữ liệu phân tán.

- Ứng dụng:

+ Xử lý dữ liệu lớn (Big Data Analytics) – Phân tích dữ liệu khổng lồ nhanh chóng.

+ Công cụ tìm kiếm (Google Search, Bing) – Lập chỉ mục, xếp hạng trang web.

+ Mạng xã hội (Facebook, YouTube, TikTok) – Đếm lượt thích, bình luận, chia sẻ.

+ Thương mại điện tử (Amazon, Shopee, Tiki) – Đề xuất sản phẩm dựa trên hành vi người dùng.

+ Phân tích log hệ thống (Apache Hadoop, AWS EMR) – Kiểm tra lỗi, giám sát hoạt động máy chủ.

+ Khoa học dữ liệu & AI – Huấn luyện mô hình Machine Learning với dữ liệu lớn.

1. **Thư viện python: NumPy, Pandas, Matplotlib (A Cường)**

a. NumPy:

- NumPy (Numerical Python): là một thư viện cung cấp nhiều công cụ tính toán mạnh mẽ với các phép toán đại số và thống kê. NumPy nổi trội trong việc làm việc với mảng đa chiều (n-dimensional array), là nền tảng cho tính toán khoa học, phân tích dữ liệu và học máy.

- NumPy có thể nhanh chóng thực hiện các phép toán như:

+ Các phép tính trên từng phần tử.

+ Các phép biến đổi đại số tuyến tính.

+ Các phép tính thống kê.

+ Sinh số ngẫu nhiên.

b. Pandas:

- Pandas được thiết kế để xử lý dữ liệu có cấu trúc, tương tự như bảng tính hoặc bảng SQL. Pandas giới thiệu hai cấu trúc dữ liệu chính: Series (một chiều) và DataFrame (hai chiều), giúp việc thao tác dữ liệu trở nên trực quan và dễ dàng.

- Một số tính năng cơ bản của Pandas:

+ Tải dữ liệu từ nhiều nguồn khác nhau (CSV, Excel, SQL database).

+ Làm sạch và chuyển đổi các loại dữ liệu.

+ Điền dữ liệu thiếu theo quy tắc, lọc và lựa chọn dữ liệu.

+ Thực hiện các thao tác nhóm và tổng hợp phức tạp.

c. Matplotlib:

- Matplotlib là thư viện hỗ trợ người dùng trong việc trực quan hóa dữ liệu thông qua các đồ thị, biểu đồ và hình vẽ. Việc tùy chỉnh các hình ảnh cho Matplotlib tạo ra cũng rất linh hoạt và dễ dàng. Thư viện này là một công cụ hữu ích cho việc biểu diễn dữ liệu cho các database.

- Nó cho phép bạn tạo các loại biểu đồ, hình ảnh sau:

+ Đồ thị đường.

+ Biểu đồ phân tán, biểu đồ cột, biểu đồ tần suất, biểu đồ tròn,..

+ Các hình trực quan hóa phức tạp với nhiều bảng.

1. **DBT(Data Build Tool) transform data (Vy)**

* Khái niệm: DBT là viết tắt cho Data Build Tool ,là một công cụ mã nguồn mở giúp các nhà phân tích và kỹ sư dữ liệu chuyển đổi dữ liệu trong kho dữ liệu bằng cách viết các câu lệnh SQL. dbt tập trung vào phần "Transform" trong quy trình ELT (Extract, Load, Transform), cho phép biến các câu lệnh SELECT thành các bảng và views trong kho dữ liệu. ​
* Lợi ích của dbt:
* **Chuyển đổi dữ liệu hiệu quả:** dbt cho phép viết và quản lý các truy vấn SQL để chuyển đổi dữ liệu một cách có cấu trúc và dễ dàng tái sử dụng.
* **Kiểm thử và tài liệu hóa:** Công cụ này hỗ trợ kiểm thử dữ liệu và tự động tạo tài liệu, giúp đảm bảo chất lượng và tính minh bạch của dữ liệu. ​
* **Quản lý phiên bản:** dbt tích hợp với hệ thống kiểm soát phiên bản như Git, cho phép theo dõi và quản lý các thay đổi trong mã SQL. ​
* **Tái sử dụng mã:** Khả năng tạo các mô-đun và macro giúp tái sử dụng mã SQL, giảm thiểu lỗi và tăng hiệu suất làm việc.
* Cách hoạt động của dbt:

**Viết các mô hình (models):** Người dùng viết các truy vấn SQL để xác định cách dữ liệu nên được chuyển đổi và lưu trữ trong kho dữ liệu.​

**Tổ chức và quản lý mã nguồn:** dbt sử dụng cấu trúc thư mục và tệp YAML để tổ chức các mô hình, tài liệu và cấu hình dự án.​

**Thực thi và kiểm thử:** Sau khi viết các mô hình, dbt sẽ thực thi chúng, tạo ra các bảng hoặc views trong kho dữ liệu, đồng thời chạy các kiểm thử để đảm bảo dữ liệu chính xác.​

**Tài liệu hóa:** dbt tự động tạo tài liệu dựa trên mô hình và kiểm thử, giúp người dùng hiểu rõ hơn về dữ liệu và quy trình chuyển đổi.

* Tại sao nên sử dụng dbt:
* **Dễ học và sử dụng:** dbt sử dụng SQL, ngôn ngữ quen thuộc với nhiều nhà phân tích dữ liệu, giúp họ dễ dàng tiếp cận và sử dụng công cụ này.
* **Cộng đồng hỗ trợ mạnh mẽ:** Là một dự án mã nguồn mở, dbt có một cộng đồng lớn mạnh, cung cấp tài liệu, hướng dẫn và hỗ trợ cho người dùng. ​
* **Tích hợp linh hoạt:** dbt có thể tích hợp với nhiều kho dữ liệu phổ biến như Snowflake, BigQuery, Redshift và PostgreSQL, giúp linh hoạt trong việc triển khai và sử dụng.