TRƯỜNG ĐẠI HỌC Y TẾ CÔNG CỘNG
CHUYÊN NGÀNH: KHOA HỌC DỮ LIỆU

BÀI TIỂU LUẬN CHỦ ĐỀ: DATA EXTRACTION

Danh sách thành viên: Nguyễn Hải An - 2211090001

Đinh Diệu Linh - 2211090022

Đinh Lê Quỳnh Phương - 2211090031

Lớp : **CNCQ KHDL1-1A**

Nhóm : **6**

Giảng viên : **Trần Lâm Quân**

Môn học : **Cơ sở dữ liệu (nâng cao)**

Năm 2025

MỤC LỤC

DANH SÁCH HÌNH ẢNH

TIỂU LUẬN HẾT MÔN CƠ SỞ DỮ LIỆU NÂNG CAO

Chương 7: DATA EXTRACTION

I. Giới thiệu tổng quan về Data Extraction:

Chương 7 trong tài liệu "Building a Data Warehouse With Examples in SQL Server" đã cung cấp những nhận thức sâu sắc về quy trình trích xuất dữ liệu trong bối cảnh xây dựng kho dữ liệu. Phân tích này nhấn mạnh đến các nguyên tắc cốt lõi, đánh giá ưu và nhược điểm của các phương pháp, và đề xuất các biện pháp khắc phục nhằm tối ưu hóa hiệu quả trong quy trình ETL (Extract, Transform, Load).

Trích xuất dữ liệu là bước đầu tiên và then chốt trong quy trình ETL, có vai trò chính là trích xuất dữ liệu từ các hệ thống nguồn đa dạng để đưa vào kho dữ liệu (Data Warehouse). Mục tiêu chính của quy trình này bao gồm:

- **Thu thập dữ liệu đa dạng:** Trích xuất dữ liệu từ nhiều hệ thống nguồn khác nhau, bao gồm cơ sở dữ liệu quan hệ (RDBMS), tệp (files), giao diện lập trình ứng dụng (API), hoặc nhật ký web (web logs).
- **Đảm bảo hiệu suất và tốc độ:** Thực hiện trích xuất dữ liệu một cách hiệu quả và nhanh chóng, đồng thời giảm thiểu tối đa ảnh hưởng đến hiệu suất hoạt động của hệ thống nguồn.
- **Đảm bảo tính toàn vẹn dữ liệu:** Giảm thiểu rủi ro phát sinh lỗi dữ liệu, chẳng hạn như thiếu dữ liệu (data loss) hoặc rò rỉ dữ liệu (data leakage) trong quá trình trích xuất.

Như vậy, trích xuất dữ liệu đóng vai trò thiết yếu trong việc xây dựng và vận hành kho dữ liệu hiện đại. Bên cạnh vai trò thu thập thông tin, quy trình này còn đảm bảo dữ liệu được chuẩn hóa, làm sạch và có khả năng tương thích cao với các quy trình phân tích tiếp theo. Điều này đặc biệt quan trọng trong các ngành công nghiệp như tài chính, chăm sóc sức khỏe và thương mại điện tử, nơi dữ liệu cần được xử lý nhanh chóng và chính xác để hỗ trợ quá trình ra quyết định.

Vai trò và ý nghĩa của Trích xuất Dữ liệu:

Trích xuất dữ liệu không chỉ đơn thuần là thu thập dữ liệu từ các nguồn mà còn là bước quan trọng để chuẩn hóa, làm sạch và đảm bảo tính tương thích cho các bước phân tích tiếp theo. Trong bối cảnh các ngành công nghiệp hiện đại như tài chính, chăm sóc sức

khỏe và thương mại điện tử, dữ liệu cần được xử lý nhanh chóng và chính xác để hỗ trợ ra quyết định. Do đó, trích xuất dữ liệu là nền tảng giúp kho dữ liệu hoạt động ổn định và đáng tin cậy, đồng thời tạo điều kiện thuận lợi cho việc triển khai các hệ thống phân tích dữ liệu tiên tiến.

II. Nguyên tắc cơ bản của Data Extraction

2.1. Tối ưu hóa tốc độ và kích thước dữ liệu truy xuất

- Giảm thời gian truy xuất: Tài liệu đề cập rằng các hệ thống OLTP (Online Transaction Processing) được thiết kế để xử lý dữ liệu trong các khối nhỏ, vì vậy việc trích xuất dữ liệu cần nhanh chóng. Một cách để thực hiện điều này là chỉ lấy dữ liệu thay đổi gần đây bằng cách sử dụng timestamp như "created" và "last_updated" trong truy vấn SQL, điều này giúp giảm tải và thời gian xử lý, tối ưu hóa hiệu suất truy xuất.

```
SELECT * FROM order_header

WHERE (created >= LSET AND created < CET) OR (last_updated >= LSET AND last_updated < CET)
```

- Lập lịch truy xuất hợp lý: Nếu hệ thống nguồn có lịch chạy batch dài qua đêm, thì có thể lập kế hoạch truy xuất vào ban ngày nhưng chia thành các phần nhỏ để không làm chậm hệ thống OLTP. Ví dụ, bạn có thể thiết lập các khoảng thời gian ngắn hơn trong ngày để trích xuất từng phần dữ liệu một cách liên tục mà không gây ảnh hưởng lớn đến hiệu suất hệ thống chính.

2.2. Hạn chế ảnh hưởng đến hệ thống nguồn (OLTP)

- Truy xuất nhẹ nhàng: Để giảm tải cho hệ thống OLTP, nên sử dụng các server phụ (read-only replica). Ví dụ, bạn có thể đọc dữ liệu từ log files hoặc từ các server phụ thay vì truy vấn trực tiếp vào hệ thống chính. Điều này giúp bảo vệ hiệu suất của hệ thống chính và đảm bảo các giao dịch không bị gián đoạn.

- Hạn chế can thiệp: Thay vì cài đặt các trigger hoặc chạy các script nặng, có thể sử dụng phương pháp đọc incremental dựa vào timestamp hoặc ID tăng dần. Ví dụ:

```
SELECT * FROM order_header

WHERE order_id > last_processed_id;
```

2.3. Đảm bảo chất lượng và tính toàn vẹn của dữ liệu

- Kiểm tra tính hợp lệ: Trước khi lưu dữ liệu vào hệ thống đích, cần thực hiện các kiểm tra để đảm bảo dữ liệu hợp lệ. Ví dụ, kiểm tra checksum để đảm bảo không có hàng bị thay đổi hoặc bị thiếu:

```
SELECT order_id, MD5(CONCAT(order_id, created, last_updated, ...)) as checksum
FROM order_header;
```

Đảm bảo các giá trị như email hoặc số điện thoại tuân thủ định dạng hợp lệ để tránh lỗi trong quá trình liên lạc.

- Phục hồi khi gặp lỗi: Trong trường hợp ETL thất bại, cần có cơ chế phục hồi để đảm bảo dữ liệu không bị mất hoặc trùng lặp. Một cách để thực hiện điều này là sử dụng cơ chế "watermark" bằng cách sử dụng LSET và CET để theo dõi các bản ghi đã được xử lý và cần phục hồi. Điều này giúp đảm bảo tính toàn vẹn của dữ liệu và khả năng khôi phục sau sự cố.

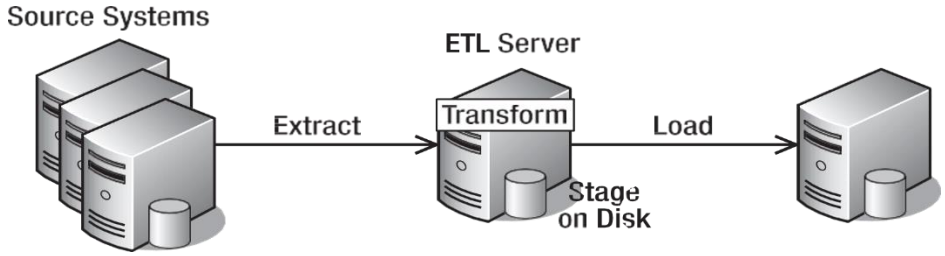
III. Các cách tiếp cận và kiến trúc ETL

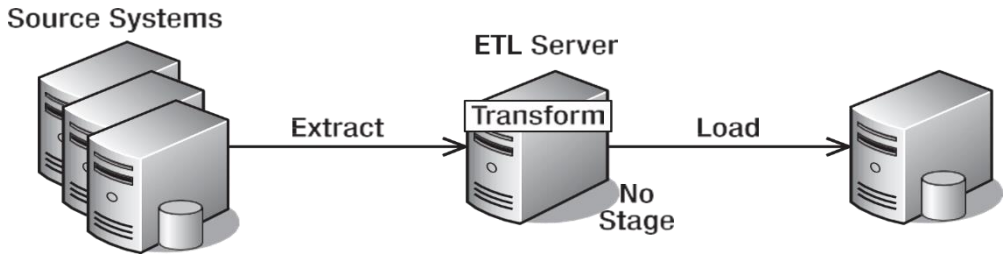
Những nguyên tắc cơ bản của Data Extraction không chỉ giúp tối ưu hóa quy trình mà còn hỗ trợ việc lựa chọn cách tiếp cận và kiến trúc ETL phù hợp. Dưới đây là các cách tiếp cận và kiến trúc chính dựa trên loại dữ liệu, yêu cầu kinh doanh và nguồn lực sẵn có.

3.1. Các cách tiếp cận ETL

3.1.1. ETL truyền thống (Extract, Transform, Load)

Quy trình: Dữ liệu được trích xuất từ hệ thống nguồn, lưu trữ tạm thời tại vùng trung gian (staging area) để làm sạch và biến đổi, sau đó tải vào kho dữ liệu đích.





Hình 1. Để xử lý trung gian trên đĩa hoặc thực hiện biến đổi trong bộ nhớ

Ưu điểm:

- Cung cấp khả năng xử lý dữ liệu mạnh mẽ nhờ sử dụng phần mềm ETL chuyên dụng.
- Cho phép thực hiện các phép biến đổi phức tạp trước khi lưu trữ dữ liệu.

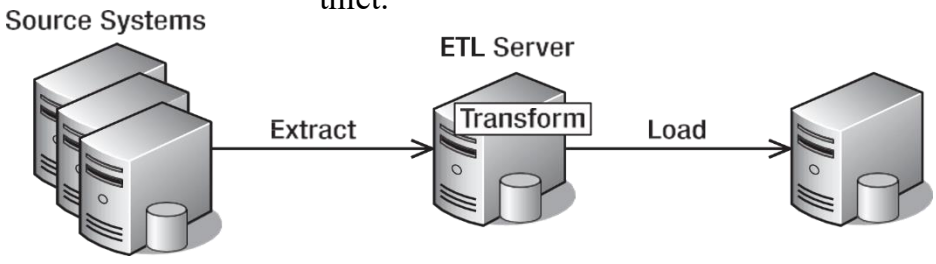
Nhược điểm:

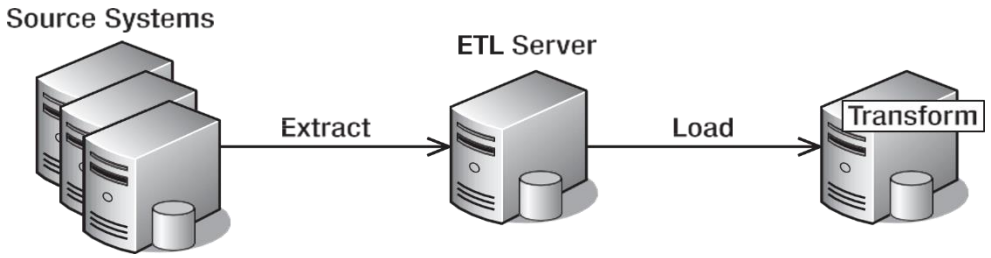
- Đòi hỏi tài nguyên và thời gian để xử lý, đặc biệt khi khối lượng dữ liệu lớn.
- Có thể gây ảnh hưởng đến hiệu suất hệ thống nguồn do phải truy xuất dữ liệu toàn bộ.

Ví dụ: Dữ liệu từ bảng order_header được trích xuất, biến đổi qua các timestamp created và last_updated để đảm bảo chỉ xử lý dữ liệu mới hoặc thay đổi trước khi tải vào kho dữ liệu.

3.1.2. ELT (Extract, Load, Transform):

Quy trình: Dữ liệu được trích xuất từ hệ thống nguồn và có thể lưu trữ tạm thời tại vùng trung gian (staging area) để làm sạch và biến đổi. Sau đó, dữ liệu được tải vào kho dữ liệu đích, nơi các bước biến đổi bổ sung có thể được thực hiện trực tiếp trong kho dữ liệu nếu cần thiết.





Hình 2. ETL và ELT: các lựa chọn về nơi thực hiện các phép biến đổi

Ưu điểm:

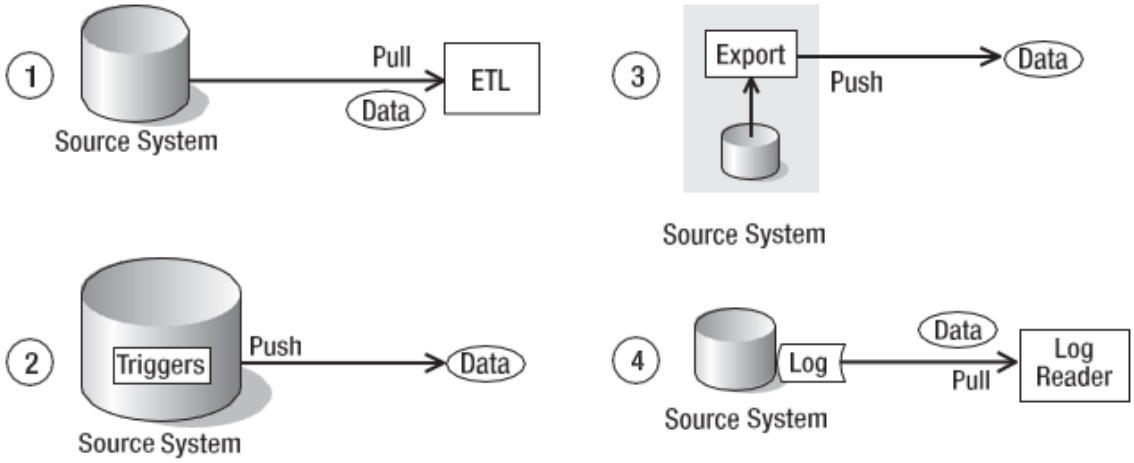
- Phù hợp với kho dữ liệu có khả năng xử lý mạnh mẽ: Như hệ thống MPP (Massively Parallel Processing) với khả năng xử lý dữ liệu lớn và phức tạp.
- Giảm tải cho hệ thống nguồn: Vì dữ liệu không cần phải biến đổi ngay lập tức trong quá trình trích xuất.

Nhược điểm:

- Đòi hỏi hệ thống đích mạnh mẽ: Để thực hiện các phép biến đổi dữ liệu, hệ thống đích (data warehouse) phải có khả năng xử lý mạnh mẽ.
- Khó khăn trong quản lý và theo dõi**:** Quá trình biến đổi dữ liệu diễn ra ngay trong kho dữ liệu có thể khó khăn trong việc quản lý và theo dõi.

Ví dụ: Hệ thống Teradata và Netezza hỗ trợ tốt cho cách tiếp cận này nhờ kiến trúc "share nothing" của hệ thống MPP, cho phép biến đổi dữ liệu nhanh chóng và hiệu quả. Trong các hệ thống MPP, mỗi nút (node) có bộ nhớ, bộ xử lý và đĩa riêng, giúp tăng hiệu suất xử lý dữ liệu theo cách tuyến tính.

3.1.3. Các phương pháp ETL dựa trên việc di chuyển dữ liệu ra khỏi hệ thống nguồn:



Hình 3. Bốn phương pháp ETL dựa trên việc ai di chuyển dữ liệu ra khỏi hệ thống nguồn

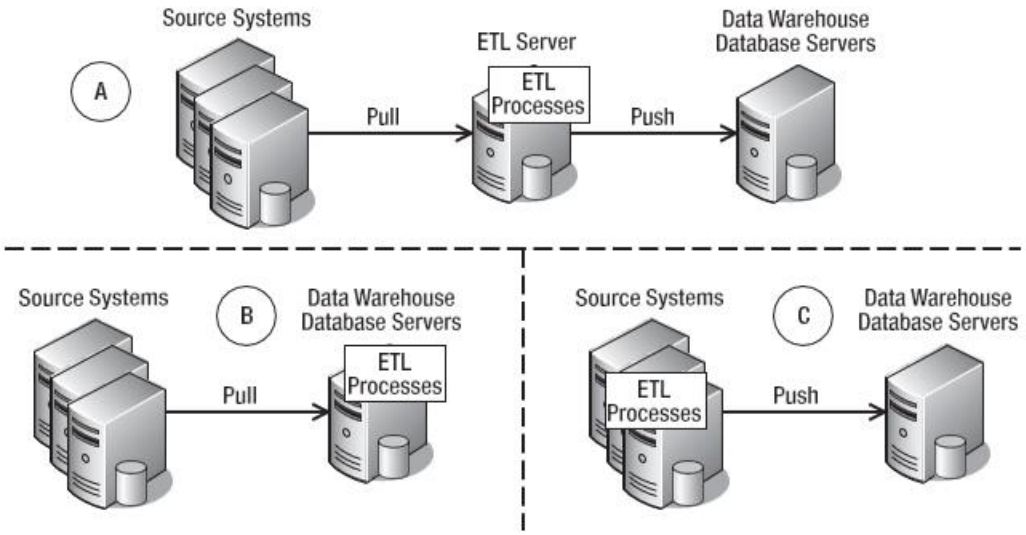
Bước 1: Truy vấn trực tiếp (Pull): Hệ thống ETL thường xuyên truy vấn cơ sở dữ liệu nguồn để lấy dữ liệu. Hệ thống ETL kết nối với cơ sở dữ liệu nguồn, thực hiện các truy vấn để lấy dữ liệu mới hoặc thay đổi, sau đó chuyển dữ liệu ra ngoài.

Bước 2: Đẩy dữ liệu qua Trigger (Push): Các trigger trong cơ sở dữ liệu nguồn tự động đẩy các thay đổi dữ liệu ra ngoài. Trigger là tập hợp các câu lệnh SQL được thực thi mỗi khi có thao tác thêm, cập nhật hoặc xóa trên một bảng. Trigger ghi lại các bản ghi thay đổi vào một bảng khác hoặc xuất dữ liệu ra ngoài.

Bước 3: Xuất dữ liệu theo lịch trình (Scheduled Export): Một quy trình được lên lịch trong hệ thống nguồn để xuất dữ liệu thường xuyên. Chương trình xuất dữ liệu là chương trình nội bộ chạy trên máy chủ hệ thống nguồn, truy vấn cơ sở dữ liệu và xuất dữ liệu ra tệp hoặc gửi đến hệ thống đích.

Bước 4: Đọc log file (Log Reader): Chương trình đọc log file của cơ sở dữ liệu để xác định các thay đổi dữ liệu. Log file chứa các bản ghi của các giao dịch được thực hiện trên cơ sở dữ liệu. Chương trình đọc log file phải hiểu định dạng của log file, trích xuất các thay đổi và lưu trữ dữ liệu ở nơi khác.

3.1.4. Các phương pháp ETL dựa trên nơi thực hiện quá trình ETL:



Hình 4. Các lựa chọn về nơi thực hiện các quy trình ETL

- Trên máy chủ ETL riêng biệt: Các quy trình ETL chạy trên máy chủ riêng biệt nằm giữa hệ thống nguồn và kho dữ liệu. Cách tiếp cận này cung cấp hiệu suất cao nhất nhưng tốn kém hơn do phải mua thêm phần cứng và phần mềm.

- **Trên máy chủ kho dữ liệu:** Các quy trình ETL chạy trên máy chủ kho dữ liệu. Phù hợp khi có khả năng dư thừa trong kho dữ liệu hoặc có khung giờ không sử dụng (ví dụ vào ban đêm).

- **Trên máy chủ hệ thống nguồn:** Các quy trình ETL chạy trên máy chủ hệ thống nguồn. Cách tiếp cận này thích hợp cho yêu cầu kho dữ liệu thời gian thực, khi thay đổi dữ liệu trong hệ thống nguồn sẽ được truyền ngay đến kho dữ liệu.

3.2. Kiến trúc ETL:

3.2.1. Sử dụng vùng trung gian (Staging Area):

Quy trình: Dữ liệu từ nguồn được lưu tạm thời tại một cơ sở dữ liệu trung gian (Staging), sau đó thực hiện biến đổi trước khi tải vào kho dữ liệu.

Ưu điểm:

- Cho phép kiểm tra và biến đổi dữ liệu kỹ lưỡng trước khi tải.
- Dễ dàng xử lý lỗi.

Ví dụ: Lưu dữ liệu từ nguồn vào staging database, thực hiện các phép biến đổi như chuẩn hóa hoặc kiểm tra giá trị, trước khi tải sang kho dữ liệu chính.

3.2.2. Biến đổi trong bộ nhớ (In-Memory Transformation):

Quy trình: Biến đổi dữ liệu trực tiếp trong bộ nhớ thay vì ghi vào vùng trung gian.

Ưu điểm: Phù hợp khi kích thước dữ liệu nhỏ và tài nguyên phần cứng mạnh mẽ.

Ví dụ: Dữ liệu nhỏ có thể biến đổi ngay trong bộ nhớ của máy chủ ETL mà không cần ghi ra đĩa.

3.2.3. Kiến trúc ETL tập trung (Centralized ETL Server):

Quy trình: Tất cả các tác vụ ETL được thực hiện trên một máy chủ riêng biệt.

Ưu điểm: Đảm bảo không làm ảnh hưởng đến hệ thống nguồn hoặc kho dữ liệu.

Ví dụ: Một tổ chức thiết lập máy chủ ETL riêng để chạy các quy trình phức tạp mà không ảnh hưởng đến hiệu suất hệ thống chính.

3.2.4. Kiến trúc phân tán (Distributed ETL):

Quy trình: Dữ liệu được xử lý tại từng hệ thống nguồn trước khi hợp nhất tại kho dữ liệu trung tâm.

Ưu điểm: Phù hợp với tổ chức lớn, có nhiều hệ thống nguồn khác nhau.

Ví dụ: Các công ty toàn cầu có thể sử dụng hệ thống message queue để thu thập dữ liệu từ nhiều chi nhánh và đồng bộ về kho dữ liệu trung tâm.

IV. Các phương pháp trích xuất dữ liệu ETL (trích xuất, chuyển đổi ,tải):

4.1. Truy xuất toàn bộ bảng (Full Table Extraction)

Đặc điểm: Lấy toàn bộ dữ liệu từ bảng nguồn mỗi lần trích xuất, bất kể dữ liệu có thay đổi hay không.

Ưu điểm:

- Đơn giản, dễ triển khai.
- Phù hợp với các bảng nhỏ hoặc khi không có cột nhận diện thay đổi (ví dụ: không có cột timestamp hoặc ID gia tăng).

Nhược điểm:

- Gây lãng phí tài nguyên nếu bảng lớn hoặc ít thay đổi.
- Tốn thời gian và có thể làm chậm hệ thống nguồn.

Ví dụ : Sử dụng khi bảng nhỏ và không có cột timestamp hoặc ID, như bảng mã code-decode.

4.2. Truy xuất gia tăng (Incremental Extraction)

Đặc điểm: Chỉ trích xuất dữ liệu đã thay đổi kể từ lần trích xuất trước (bao gồm thêm, sửa, hoặc xóa).

Cách thực hiện:

- Dựa trên cột timestamp (dấu mốc thời gian) hoặc incremental ID.
- So sánh dữ liệu hiện tại với LSET (Last Successful Extraction Time) và CET (Current Extraction Time).

Ưu điểm:

- Hiệu quả và nhanh hơn khi làm việc với bảng lớn.
- Giảm tải cho hệ thống nguồn.

Nhược điểm:

- Phụ thuộc vào cột timestamp hoặc ID gia tăng đáng tin cậy.
- Có thể cần thêm xử lý để phát hiện các bản ghi bị xóa.

Ví dụ:

```
SELECT *
FROM order_header
WHERE (created >= LSET AND created < CET)
    OR (last_updated >= LSET AND last_updated < CET);
```

4.3. Phạm vi cố định (Fixed Range Extraction)

Đặc điểm: Trích xuất dữ liệu theo một phạm vi cố định (ví dụ: dữ liệu của 6 tháng gần nhất hoặc 100,000 bản ghi mới nhất).

Ưu điểm:

- Phù hợp khi không có cột nhận diện thay đổi.
- Giảm thời gian trích xuất cho các bảng rất lớn.

Nhược điểm:

- Có thể bỏ sót dữ liệu thay đổi nằm ngoài phạm vi đã chọn.
- Cần thêm xử lý để phát hiện bản ghi bị xóa.

Ví dụ trong tài liệu: Truy xuất các bản ghi dựa trên order_date với thời gian cố định:

```
SELECT *
FROM order_header
WHERE order_date >= (LSET- INTERVAL '6 months') AND order_date < CET;
```

4.4. ELT (Extract, Load, Transform)

Đặc điểm: Khác với ETL truyền thống, dữ liệu được trích xuất và tải trực tiếp vào kho dữ liệu, sau đó thực hiện biến đổi ngay trong kho dữ liệu.

Ưu điểm: Tận dụng hiệu suất xử lý mạnh mẽ của các hệ thống kho dữ liệu hiện đại (như Teradata hoặc Netezza).

Nhược điểm: Yêu cầu hệ thống kho dữ liệu phải mạnh mẽ và được thiết kế tốt.

Ví dụ trong tài liệu: Sử dụng kiến trúc MPP để xử lý song song, tăng hiệu quả xử lý các bảng lớn.

V. Các công cụ và kĩ thuật trích xuất:

5.1. Công cụ trích xuất dữ liệu:

5.1.1. SQL Server Integration Services (SSIS):

- Là một công cụ ETL mạnh mẽ của Microsoft, tích hợp tốt với SQL Server.
- Đặc điểm nổi bật:
  + Giao diện trực quan, dễ sử dụng.
  + Hỗ trợ kết nối đa dạng: cơ sở dữ liệu, tệp tin (CSV, XML), và dịch vụ web.
  + Khả năng tự động hóa và giám sát toàn bộ quy trình trích xuất.
- Ứng dụng: Trích xuất và chuyển đổi dữ liệu trong môi trường doanh nghiệp sử dụng SQL Server.
- Ví dụ: Tạo nguồn dữ liệu, ánh xạ và chạy luồng dữ liệu trong SSIS để trích xuất dữ liệu từ bảng order_header:

```
SELECT * FROM order_header
WHERE (created > ? AND created <= ?)
  OR (last_updated > ? AND last_updated <= ?);
```

5.1.2. Apache Kafka:

- Nền tảng xử lý dữ liệu thời gian thực, phù hợp với dữ liệu thay đổi liên tục.
- Đặc điểm nổi bật:
  + Hỗ trợ xử lý song song và có khả năng mở rộng cao.

+ Tích hợp với các hệ thống nguồn và đích qua các connector.

- Ứng dụng: Lý tưởng cho các doanh nghiệp yêu cầu trích xuất dữ liệu theo thời gian thực.

5.1.3. Talend Open Studio:

- Công cụ ETL mã nguồn mở, phù hợp với doanh nghiệp có ngân sách hạn chế.
- Đặc điểm nổi bật:

  + Linh hoạt trong việc tích hợp nhiều nguồn dữ liệu (REST API, NoSQL, JSON).

  + Không yêu cầu phí giấy phép, dễ dàng mở rộng.

- Ứng dụng: Các dự án nhỏ đến trung bình yêu cầu ETL đơn giản nhưng linh hoạt.

5.1.4. Informatica:

- Công cụ ETL cao cấp, phù hợp cho xử lý dữ liệu lớn và phức tạp.
- Đặc điểm nổi bật:

  + Hỗ trợ bảo mật tốt, phù hợp với các ngành như tài chính và y tế.

  + Xử lý dữ liệu nhanh chóng với khả năng tích hợp sâu vào hạ tầng doanh nghiệp.

- Ứng dụng: Các tổ chức lớn với nhu cầu xử lý dữ liệu khối lượng lớn.

5.2. Kĩ thuật trích xuất dữ liệu:

Việc lựa chọn kỹ thuật trích xuất dữ liệu phù hợp là rất quan trọng trong quá trình xây dựng kho dữ liệu. Mỗi kỹ thuật có những ưu và nhược điểm riêng, và việc lựa chọn phụ thuộc vào yêu cầu cụ thể của dự án, đặc điểm của dữ liệu nguồn và hiệu suất mong muốn. Dưới đây là 6 kỹ thuật trích xuất dữ liệu phổ biến:

5.2.1. Toàn bộ bảng (Full Table):

Mô tả: Phương pháp đơn giản nhất, trích xuất *toàn bộ* dữ liệu từ bảng nguồn mỗi lần. Dữ liệu được sao chép hoàn toàn vào kho dữ liệu.

Ưu điểm: Dễ triển khai, không yêu cầu logic phức tạp, phù hợp cho các bảng nhỏ hoặc khi lần đầu tiên khởi tạo kho dữ liệu.

Nhược điểm: Tốn rất nhiều tài nguyên hệ thống (CPU, bộ nhớ, băng thông mạng) nếu bảng dữ liệu có kích thước lớn. Không hiệu quả cho dữ liệu thay đổi thường xuyên, vì mỗi lần trích xuất sẽ sao chép toàn bộ dữ liệu, gây lãng phí tài nguyên và thời gian.

Ví dụ (trong ngữ cảnh SQL Server): Sao chép toàn bộ bảng Customers từ cơ sở dữ liệu OLTP (ví dụ: một cơ sở dữ liệu quản lý bán hàng) vào bảng DimCustomer (bảng chiều

khách hàng) trong kho dữ liệu hàng đêm bằng một câu lệnh SELECT INTO hoặc INSERT INTO SELECT.

5.2.2. Gia tăng (Incremental):

Mô tả: Chỉ trích xuất dữ liệu *mới* hoặc *đã thay đổi* kể từ lần trích xuất gần nhất. Giúp giảm tải cho hệ thống nguồn và tăng tốc độ xử lý.

Ưu điểm: Giảm đáng kể tải cho hệ thống nguồn, tăng tốc độ trích xuất và xử lý dữ liệu, tiết kiệm tài nguyên.

Nhược điểm: Yêu cầu bảng nguồn phải có cột dấu thời gian (timestamp, ví dụ: last_updated, modified_date) hoặc ID tự tăng (auto-increment) để xác định dữ liệu mới hoặc đã thay đổi. Cần quản lý trạng thái của lần trích xuất cuối cùng.

Ví dụ SQL:

SELECT * FROM orders WHERE last_updated > '2023-01-01 00:00:00' (chọn tất cả đơn hàng được cập nhật sau ngày 01/01/2023).

Các phương pháp triển khai Incremental:

- Dựa trên timestamp**:** Sử dụng cột timestamp để so sánh thời gian.
- Dựa trên ID tự tăng: Sử dụng ID tự tăng để xác định các bản ghi mới.
- Sử dụng Change Data Capture (CDC) trong SQL Server: Một tính năng của SQL Server giúp theo dõi các thay đổi dữ liệu một cách hiệu quả.

5.2.3. Dùng trigger:

Mô tả: Sử dụng trigger trong cơ sở dữ liệu nguồn để tự động theo dõi các thay đổi (INSERT, UPDATE, DELETE) và ghi lại thông tin về các thay đổi này vào một bảng sự kiện (event table) hoặc bảng nhật ký (log table).

Ưu điểm: Đảm bảo dữ liệu được trích xuất đầy đủ và chính xác, dễ dàng quản lý lịch sử thay đổi, có thể trích xuất dữ liệu gần như ngay lập tức sau khi có thay đổi.

Nhược điểm: Tăng tải xử lý trên hệ thống nguồn do trigger phải được thực thi mỗi khi có thay đổi dữ liệu. Có thể ảnh hưởng đến hiệu suất của hệ thống nguồn nếu số lượng thay đổi lớn. Khó khăn trong việc bảo trì và gỡ lỗi trigger.

Ví dụ: Tạo một trigger TR_Orders_AfterUpdate trên bảng Orders để ghi lại các thay đổi vào bảng OrderChanges mỗi khi có bản ghi trong bảng Orders bị cập nhật.

5.2.4. Đọc log giao dịch (Transaction Log):

Mô tả: Phân tích log giao dịch (transaction log) của cơ sở dữ liệu để lấy thông tin về các thay đổi dữ liệu. Log giao dịch ghi lại tất cả các hoạt động thay đổi dữ liệu trong cơ sở dữ liệu.

Ưu điểm: Không tác động trực tiếp đến cơ sở dữ liệu nguồn, do đó ít ảnh hưởng đến hiệu suất của hệ thống nguồn. Có thể trích xuất dữ liệu với độ trễ thấp.

Nhược điểm: Yêu cầu công cụ chuyên dụng để đọc và phân tích log giao dịch, phức tạp hơn so với các phương pháp khác. Định dạng log giao dịch phụ thuộc vào từng hệ quản trị cơ sở dữ liệu (DBMS). Khó khăn trong việc xử lý các log bị hỏng.

Ví dụ: Sử dụng các công cụ như fn_dblog hoặc các công cụ của bên thứ ba để đọc và phân tích log giao dịch. Hoặc sử dụng Change Data Capture (CDC), một tính năng dựa trên transaction log, nhưng dễ sử dụng hơn.

5.2.5. Phương pháp theo lô (Batch Extraction):

Mô tả: Chia dữ liệu thành các lô (batch) nhỏ hơn để trích xuất theo từng đợt, thường theo lịch trình định kỳ (ví dụ: hàng giờ, hàng ngày).

Ưu điểm: Giảm tải tức thời cho hệ thống nguồn, tránh quá tải hệ thống. Dễ dàng quản lý và kiểm soát quá trình trích xuất.

Nhược điểm: Dữ liệu trong kho dữ liệu không được cập nhật liên tục, có độ trễ nhất định. Phức tạp hơn trong việc quản lý và điều phối các lô, đặc biệt khi số lượng lô lớn.

Ví dụ: Sử dụng SQL Server Integration Services (SSIS) để tạo một package trích xuất dữ liệu bán hàng từ cơ sở dữ liệu OLTP vào kho dữ liệu vào cuối mỗi ngày.

5.2.6. Dữ liệu thời gian thực (Real-time Extraction):

Mô tả: Trích xuất dữ liệu ngay lập tức khi có thay đổi trong hệ thống nguồn. Thường sử dụng các công nghệ như message queue (ví dụ: Apache Kafka, RabbitMQ) hoặc Change Data Capture (CDC).

Ưu điểm: Cung cấp dữ liệu gần như ngay lập tức, phù hợp cho các ứng dụng yêu cầu cập nhật dữ liệu liên tục, ví dụ như giám sát hệ thống, phân tích hành vi người dùng trực tuyến.

Nhược điểm: Yêu cầu cơ sở hạ tầng mạnh và phần mềm chuyên dụng, phức tạp trong việc triển khai và quản lý. Yêu cầu kiến trúc hệ thống phức tạp hơn.

Ví dụ: Sử dụng CDC để ghi lại các thay đổi trong SQL Server và sau đó sử dụng một connector để đẩy các thay đổi này vào Kafka.

Lưu ý thực tiễn:

- Trong thực tế, nên kết hợp nhiều kỹ thuật để tối ưu hiệu suất và đáp ứng các yêu cầu khác nhau. Ví dụ: sử dụng trigger cho các bảng nhỏ và batch extraction cho dữ liệu lớn. Sử dụng CDC kết hợp với Kafka cho dữ liệu cần cập nhật theo thời gian thực.
- Luôn kiểm tra và giám sát dữ liệu trích xuất để đảm bảo tính chính xác và đầy đủ của dữ liệu trong kho dữ liệu. Sử dụng các công cụ kiểm tra chất lượng dữ liệu (Data Quality) trong SQL Server Integration Services (SSIS) để phát hiện và xử lý các vấn đề về dữ liệu.

VI . Công cụ và ví dụ thực tiễn:

6.1. Quy trình trích xuất với SQL Server Integration Services (SSIS):

SQL Server Integration Services (SSIS) là một thành phần quan trọng của nền tảng SQL Server, được thiết kế để xây dựng các giải pháp tích hợp dữ liệu và chuyển đổi dữ liệu mạnh mẽ, thường được biết đến với tên gọi ETL (Extract, Transform, Load). Quy trình trích xuất dữ liệu trong SSIS bao gồm các giai đoạn chính sau

6.1.1. Giai đoạn thiết lập kết nối dữ liệu:

Giai đoạn này tập trung vào việc thiết lập kết nối đến các nguồn dữ liệu khác nhau, bao gồm cả cơ sở dữ liệu nguồn (thường là hệ thống OLTP) và khu vực dàn dựng (staging area) hoặc trực tiếp đến kho dữ liệu đích. Trong SSIS, việc này được thực hiện thông qua các *Connection Manager*. Các loại Connection Manager phổ biến bao gồm OLE DB, ADO.NET và Flat File, cho phép kết nối đến nhiều loại nguồn dữ liệu như SQL Server, Oracle, tệp văn bản, v.v. Việc thiết lập kết nối chính xác là tiền đề quan trọng cho các giai đoạn tiếp theo.

6.1.2. Giai đoạn thiết kế luồng dữ liệu:

Đây là giai đoạn cốt lõi của quy trình ETL trong SSIS, được thực hiện trong *Data Flow Task*. Giai đoạn này bao gồm hai bước chính:

- **Trích xuất dữ liệu:** Dữ liệu được trích xuất từ nguồn bằng cách sử dụng các *Source component*, ví dụ như *OLE DB Source* hoặc *SQL Server Source*. Câu lệnh SQL được sử dụng để xác định dữ liệu cần trích xuất. Ví dụ, để trích xuất dữ liệu gia tăng từ bảng order_header dựa trên dấu thời gian, câu lệnh SQL có thể được xây dựng như sau: SELECT * FROM order_header WHERE (created > ? AND created <= ?) OR (last_updated > ? AND last_updated <= ?). Các tham số ? sẽ được gán giá trị bởi các biến trong SSIS.

- **Biến đổi và tải dữ liệu:** Sau khi dữ liệu được trích xuất, các *Transformation component* (nếu cần) được sử dụng để thực hiện các biến đổi dữ liệu như làm sạch, chuyển đổi kiểu dữ liệu, tổng hợp dữ liệu, v.v. Cuối cùng, dữ liệu đã được biến đổi sẽ được tải vào bảng đích trong kho dữ liệu bằng cách sử dụng các *Destination component*, ví dụ như *OLE DB Destination* hoặc *SQL Server Destination*.

6.1.3. Giai đoạn thực thi và kiểm tra:

Sau khi luồng dữ liệu được thiết kế, package SSIS sẽ được thực thi. Giai đoạn này bao gồm việc chạy package và kiểm tra dữ liệu trong bảng đích để đảm bảo tính chính xác và đầy đủ của dữ liệu đã được trích xuất và tải. Việc kiểm tra có thể bao gồm so sánh số lượng bản ghi, kiểm tra tính toàn vẹn dữ liệu và kiểm tra các giá trị cụ thể.

6.2. Ví dụ minh họa: Trích xuất dữ liệu gia tăng từ bảng order_header:

Ví dụ trong nghiên cứu này tập trung vào việc trích xuất dữ liệu gia tăng từ bảng order_header dựa trên logic dấu thời gian. Câu lệnh SQL được sử dụng là:

```
SELECT * FROM order_header
WHERE (created > ? AND created <= ?) OR (last_updated > ? AND last_updated <= ?).
```

Để quản lý việc trích xuất gia tăng, hai biến thời gian được sử dụng:

- CET (Current Extraction Time): Thời gian trích xuất hiện tại, được lấy trước khi bắt đầu quá trình trích xuất.
- LSET (Last Successful Extraction Time): Thời gian trích xuất thành công lần trước, được lưu trữ và cập nhật sau mỗi lần trích xuất thành công.

Trong SSIS, các biến và biểu thức được sử dụng để quản lý CET và LSET và gán chúng vào câu lệnh SQL. Logic này đảm bảo chỉ có dữ liệu mới hoặc đã thay đổi kể từ lần trích xuất trước được trích xuất, giúp tối ưu hiệu suất.

VII. Các vấn đề và thách thức trong trích xuất dữ liệu:

Trích xuất dữ liệu, mặc dù là bước khởi đầu trong quy trình ETL, lại tiềm ẩn nhiều thách thức và vấn đề cần được xem xét cẩn thận để đảm bảo tính chính xác, hiệu quả và ổn định của toàn bộ hệ thống kho dữ liệu. Một số vấn đề và thách thức chính bao gồm:

7.1. Rò rỉ dữ liệu (Data Leakage):

Rò rỉ dữ liệu xảy ra khi dữ liệu bị mất mát hoặc không được trích xuất đầy đủ trong quá trình chuyển đổi từ hệ thống nguồn sang kho dữ liệu. Điều này có thể dẫn đến phân tích sai lệch và đưa ra quyết định không chính xác. Các nguyên nhân chính gây ra rò rỉ dữ liệu bao gồm:

- Lỗi logic trong quy trình trích xuất: Các lỗi trong truy vấn SQL, logic xử lý dữ liệu hoặc cấu hình ETL có thể dẫn đến việc bỏ sót một số bản ghi hoặc trường dữ liệu. Ví dụ, một điều kiện WHERE không chính xác trong truy vấn có thể loại bỏ nhầm các bản ghi cần thiết.
- Sai sót trong việc xác định dữ liệu đã thay đổi (đối với trích xuất gia tăng): Trong các hệ thống trích xuất gia tăng, việc không phát hiện chính xác các thay đổi trong dữ liệu nguồn (ví dụ: do thiếu trigger hoặc log không đầy đủ) có thể dẫn đến việc bỏ qua các bản ghi đã được cập nhật hoặc chèn mới.

Để khắc phục vấn đề rò rỉ dữ liệu, các biện pháp sau có thể được áp dụng:

- Kiểm tra đối chiếu dữ liệu (Data Reconciliation): So sánh dữ liệu giữa hệ thống nguồn và kho dữ liệu sau khi trích xuất để phát hiện sự khác biệt.
- Sử dụng các công cụ kiểm tra chất lượng dữ liệu (Data Quality Check): Áp dụng các quy tắc và ràng buộc dữ liệu để kiểm tra tính đầy đủ, nhất quán và chính xác của dữ liệu được trích xuất.

7.2. Ảnh hưởng đến hệ thống nguồn:

Quá trình trích xuất dữ liệu có thể tạo ra tải cho hệ thống nguồn, đặc biệt là các hệ thống OLTP đang hoạt động. Các truy vấn trích xuất, đặc biệt là các truy vấn phức tạp hoặc truy vấn toàn bộ bảng, có thể tiêu tốn tài nguyên hệ thống (CPU, bộ nhớ, I/O) và làm chậm hiệu suất của hệ thống nguồn, ảnh hưởng đến hoạt động của người dùng.

Để giảm thiểu ảnh hưởng đến hệ thống nguồn, các giải pháp sau có thể được áp dụng:

- Trích xuất vào thời điểm ít người dùng (ngoài giờ làm việc): Lên lịch trích xuất vào thời điểm hệ thống ít chịu tải nhất.
- Sử dụng cơ sở dữ liệu phụ hoặc bản sao (Snapshot/Replication): Tạo một bản sao của cơ sở dữ liệu nguồn và thực hiện trích xuất trên bản sao này, tránh ảnh hưởng trực tiếp đến hệ thống chính.

VIII. Đánh giá ưu và nhược điểm của phương pháp trích xuất dữ liệu:

Các phương pháp trích xuất dữ liệu khác nhau mang lại những ưu và nhược điểm riêng. Việc lựa chọn phương pháp phù hợp phụ thuộc vào yêu cầu cụ thể của từng dự án và đặc điểm của hệ thống nguồn.

8.1. Ưu điểm:

- Tải gia tăng (Incremental Load): Chỉ trích xuất dữ liệu đã thay đổi, giúp tiết kiệm tài nguyên, giảm tải cho hệ thống nguồn và tối ưu hóa hiệu suất trích xuất.

- Khả năng tích hợp linh hoạt: Hỗ trợ nhiều loại nguồn dữ liệu khác nhau, từ cơ sở dữ liệu quan hệ, tệp tin, API đến nhật ký giao dịch.

- Tự động hóa: Sử dụng các công cụ ETL như SQL Server Integration Services (SSIS) giúp tự động hóa và đơn giản hóa quy trình trích xuất, giảm thiểu công sức thủ công.

- Khả năng mở rộng (Scalability): Hỗ trợ xử lý lượng dữ liệu lớn với khả năng xử lý song song trong các hệ thống MPP (Massively Parallel Processing).

8.2. Nhược điểm:

- Tính phức tạp: Việc thiết lập trigger hoặc đọc log giao dịch để thực hiện trích xuất gia tăng đòi hỏi kỹ năng chuyên môn cao. Các hệ thống ETL cũng yêu cầu kiến trúc và cấu hình kỹ thuật phức tạp.

- Phụ thuộc vào công cụ: Yêu cầu người dùng phải có kiến thức và kỹ năng sử dụng các công cụ ETL như SSIS và driver cơ sở dữ liệu.

- Hiệu suất giới hạn (trong một số trường hợp): Việc xử lý dữ liệu trong bộ nhớ có thể gây khó khăn khi trích xuất và biến đổi lượng dữ liệu quá lớn.

- Rủi ro bỏ lỡ dữ liệu (đối với trích xuất gia tăng): Các quy trình trích xuất gia tăng có thể bỏ qua các thay đổi nếu không được thiết kế và kiểm tra kỹ lưỡng.

VII. Kết luận:

Trích xuất dữ liệu là một bước cốt lõi trong quy trình ETL, đóng vai trò quan trọng trong việc xây dựng và duy trì kho dữ liệu hiệu quả. Việc lựa chọn phương pháp trích xuất phù hợp đòi hỏi sự cân nhắc kỹ lưỡng về các yếu tố như yêu cầu nghiệp vụ, đặc điểm của hệ thống nguồn, nguồn lực kỹ thuật và ngân sách. Đầu tư vào công nghệ, kỹ năng và quy trình trích xuất bài bản là yếu tố then chốt để đảm bảo thành công của dự án kho dữ liệu.

MỘT SỐ TÀI LIỆU THAM KHẢO

1. Schmidt, L., Finnerty Mutlu, A. N., Elmore, R., Olorisade, B. K., Thomas, J., & Higgins, J. P. T. (2021). *Data extraction methods for systematic review (semi)automation: Update of a living systematic review*. doi.org/10.12688/f1000research.51117.2

2. Mykowiecka, A., Marciniak, M., & Kupść, A. (2009). *Rule-based information extraction from patients' clinical data*. Journal of Biomedical Informatics, [1] *42*(6), 1021–1032. https://doi.org/10.1016/j.jbi.2009.07.007