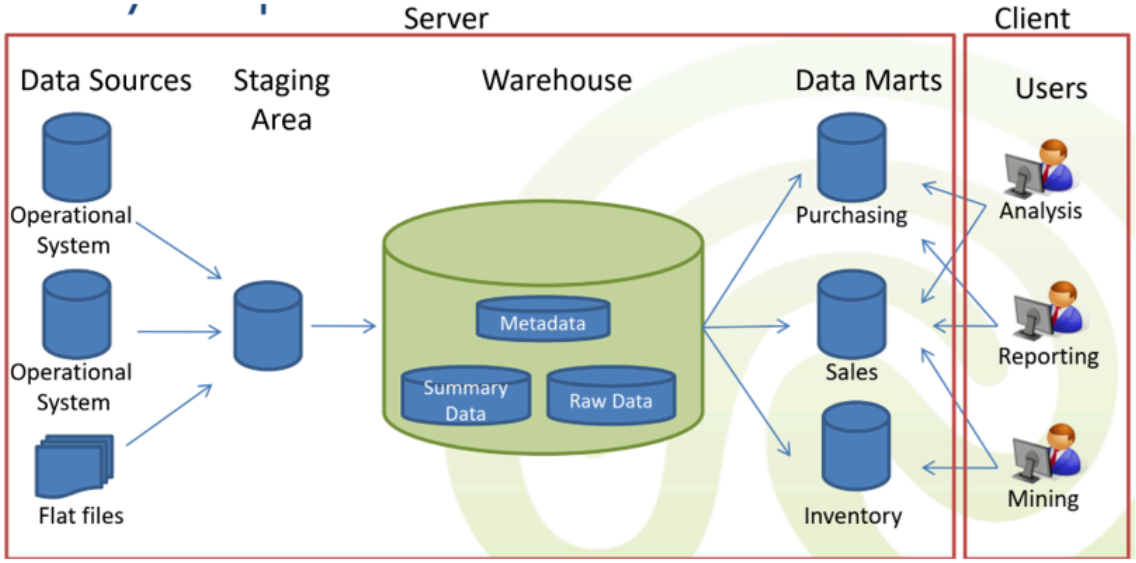
ETL

Created time	@April 17, 2025 6:30 PM
Last updated time	@April 23, 2025 6:29 PM

Các bước ETL:

1. Trích chọn dữ liệu (Extract)
2. Chuyển đổi dữ liệu (Transform)
3. Tải dữ liệu vào Data Warehouse (Load)



Trong quá trình ETL thì chia ra thành 2 loại là:

ETL lần đầu	ETL các lần tiếp theo
Thực hiện trên toàn bộ dữ liệu	Chỉ xem xét sự thay đổi và tính toán thêm trên sự thay đổi đó, cập nhật DW dựa trên sự thay đổi thôi, không cần tính toán hết lại từ đầu

Có 2 phương pháp để kiểm soát sự thay đổi dữ liệu giữa các lần ETL:

	Phương pháp sử dụng bảng Staging làm Snapshot	**Phương pháp sử dụng bảng Snapshot riêng**

Cách thức hoạt động	1. Giữ lại dữ liệu từ lần ETL trước trong bảng Staging 2. Lưu dữ liệu mới vào bảng tạm thời 3. So sánh dữ liệu mới với dữ liệu cũ trong Staging 4. Cập nhật DW dựa trên sự khác biệt 5. Thay thế dữ liệu cũ trong Staging bằng dữ liệu mới	1. Duy trì bảng Snapshot riêng với cấu trúc tinh gọn 1. Lưu dữ liệu mới vào bảng Staging riêng 1. So sánh dữ liệu mới với Snapshot 1. Cập nhật DW dựa trên sự khác biệt 1. Cập nhật Snapshot với dữ liệu mới
Ưu điểm	1. **Đơn giản hơn**: Ít bảng hơn, quản lý đơn giản hơn 2. **Tiết kiệm không gian lưu trữ**: Không cần thêm bảng Snapshot 3. **Kiểm tra toàn diện**: So sánh tất cả các trường dữ liệu 4. **Đơn giản về khái niệm**: Dễ hiểu và triển khai hơn 5. **Quản lý trạng thái dễ dàng**: Tất cả dữ liệu và trạng thái ở một nơi	1. **Hiệu suất tốt hơn**: Snapshot thường có cấu trúc đơn giản, ít trường hơn 2. **Tách biệt quan tâm**: Staging cho biến đổi, Snapshot cho theo dõi thay đổi 3. **Ít phức tạp khi biến đổi**: Biến đổi trong Staging không ảnh hưởng đến Snapshot 4. **Dễ mở rộng**: Có thể giữ nhiều phiên bản Snapshot (lịch sử thay đổi) 5. **Tối ưu hóa cho phát hiện thay đổi**: Sử dụng hash để so sánh nhanh chóng

Tại sao khi tạo bảng Snapshot thì nên có thêm trường Snapshot, LoadDateTime và CurrentFlag so với bảng Stagging?

```
-- 2. Bảng Customers
CREATE TABLE STG.Customers (
    CustomerID nchar(5),
    CompanyName nvarchar(40),
    ContactName nvarchar(30),
    ContactTitle nvarchar(30),
    Address nvarchar(60),
    City nvarchar(15),
    Region nvarchar(15),
    PostalCode nvarchar(10),
    Country nvarchar(15),
    Phone nvarchar(24),
    Fax nvarchar(24)
);
```

```
CREATE TABLE SNP.Customers (
    SnapshotID int IDENTITY(1,1) PRIMARY KEY,
    CustomerID nchar(5),
    CompanyName nvarchar(40),
    ContactName nvarchar(30),
    ContactTitle nvarchar(30),
    Address nvarchar(60),
    City nvarchar(15),
    Region nvarchar(15),
    PostalCode nvarchar(10),
    Country nvarchar(15),
    Phone nvarchar(24),
    Fax nvarchar(24),
    LoadDateTime datetime NOT NULL DEFAULT GETDATE(),
    CurrentFlag bit DEFAULT 1
```

Ví dụ:

Ngày 1: Bạn có 3 khách hàng trong hệ thống:

- Nguyễn Văn A: Sống ở Hà Nội, SĐT: 0901234567
- Trần Thị B: Sống ở TP.HCM, SĐT: 0912345678
- Lê Văn C: Sống ở Đà Nẵng, SĐT: 0923456789

Tất cả đều được đánh dấu CurrentFlag = 1 (thông tin hiện tại) trong bảng Snapshot.

Ngày 2

: Có một số thay đổi:

- Khách hàng A chuyển đến TP.HCM (thay đổi địa chỉ)
- Khách hàng D mới (Phạm Thị D) đến mua hàng (bản ghi mới)
- Khách hàng C yêu cầu xóa thông tin khỏi hệ thống (bản ghi bị xóa)

Quy trình xử lý:

Bước 1: Load dữ liệu mới vào STG

Bảng STG bây giờ chứa:

- Nguyễn Văn A: Sống ở TP.HCM, SĐT: 0901234567 (đã cập nhật)
- Trần Thị B: Sống ở TP.HCM, SĐT: 0912345678 (không đổi)
- Phạm Thị D: Sống ở Cần Thơ, SĐT: 0934567890 (mới)

(không có thông tin của C vì đã bị xóa)

Bước 2: So sánh STG với SNP (chỉ những bản ghi có CurrentFlag = 1)

2.1. Tìm bản ghi mới:

So sánh STG với SNP, tìm những bản ghi có trong STG nhưng không có trong SNP.

→ Tìm thấy: Phạm Thị D (khách hàng mới)

2.2. Tìm bản ghi thay đổi:

So sánh STG với SNP, tìm những khách hàng có cùng ID nhưng thông tin khác nhau.

→ Tìm thấy: Nguyễn Văn A (địa chỉ đã thay đổi từ Hà Nội → TP.HCM)

2.3. Tìm bản ghi bị xóa:

So sánh SNP với STG, tìm những bản ghi có trong SNP (với CurrentFlag = 1) nhưng không có trong STG.

→ Tìm thấy: Lê Văn C (đã bị xóa)

Bước 3: Cập nhật SNP

3.1. Đánh dấu bản ghi cũ:

- Đổi CurrentFlag = 0 cho bản ghi cũ của Nguyễn Văn A (địa chỉ Hà Nội)
- Đổi CurrentFlag = 0 cho bản ghi của Lê Văn C (đã bị xóa)

3.2. Thêm bản ghi mới:

- Thêm bản ghi mới cho Nguyễn Văn A với địa chỉ TP.HCM, CurrentFlag = 1
- Thêm bản ghi mới cho Phạm Thị D, CurrentFlag = 1

Sau khi cập nhật, bảng SNP sẽ có:

- Nguyễn Văn A: Hà Nội, CurrentFlag = 0 (bản ghi cũ)
- Nguyễn Văn A: TP.HCM, CurrentFlag = 1 (bản ghi mới)
- Trần Thị B: TP.HCM, CurrentFlag = 1 (không đổi)
- Lê Văn C: Đà Nẵng, CurrentFlag = 0 (đã bị xóa)
- Phạm Thị D: Cần Thơ, CurrentFlag = 1 (mới thêm)

Bước 4: Chỉ cập nhật Data Warehouse cho những thay đổi

Thay vì tính toán lại dữ liệu cho tất cả khách hàng, bạn chỉ cần xử lý:

- Thêm Phạm Thị D vào DW (khách hàng mới)
- Cập nhật địa chỉ của Nguyễn Văn A thành TP.HCM trong DW
- Đánh dấu Lê Văn C là không còn hoạt động trong DW

Vậy thì, ví dụ bảng Snapshot Customer thì sẽ có nhiều bản ghi có trùng Customer ID phải không?

Trong bảng Snapshot, khóa chính (Primary Key) không phải là ID của khách hàng (như CustomerID) mà thường là **Khóa riêng (Snapshot ID tự tăng)**.

SnapshotID	CustomerID	Tên	Địa chỉ	Thời gian cập nhật	CurrentFlag
1	A	Nguyễn Văn A	Hà Nội	01/01/2023	0
2	B	Trần Thị B	TP.HCM	01/01/2023	1
3	C	Lê Văn C	Đà Nẵng	01/01/2023	0
4	A	Nguyễn Văn A	TP.HCM	02/01/2023	1
5	D	Phạm Thị D	Cần Thơ	02/01/2023	1

Như ta thấy:

- CustomerID = 'A' xuất hiện 2 lần, nhưng có SnapshotID khác nhau
- Ta phân biệt phiên bản hiện tại bằng CurrentFlag = 1

Cách truy vấn

Khi cần thông tin hiện tại, bạn luôn thêm điều kiện CurrentFlag = 1:

```
SELECT * FROM SNP.Customers WHERE CustomerID = 'A' AND CurrentFlag = 1;
```

Vì sao thiết kế như vậy?

1. **Lưu vết lịch sử**: Bạn biết khách hàng A đã chuyển từ Hà Nội đến TP.HCM
2. **Phân tích xu hướng**: Có thể phân tích việc di chuyển của khách hàng theo thời gian
3. **Kiểm tra/Audit**: Khi cần kiểm tra, bạn có thể xem thông tin trước đây
4. **Khôi phục**: Nếu cập nhật sai, có thể khôi phục về phiên bản trước

Tuy nhiên, dùng 1 cột "Thời gian cập nhật" kết hợp với cột CurrentFlag cũng có hạn chế:

- Không phân biệt được, là do người dùng nhập sai trước đó, và cần cập nhật để sửa thông tin hay là vì người dùng thực sự chuyển đổi nơi ở nên cần cập nhật ⇒ Tính truy vết lịch sử chỉ mang tính tham khảo khi cần.

Luồng dữ liệu và tiến trình xử lý ETL:

Quy trình: Nguồn → Stagging → Xử lý cơ bản → Snapshot → Xử lý nâng cao → DW

Mô tả:

Bước 1. Tạo SQL thực hiện tiến trình: Lấy dữ liều từ Nguồn → Stagging

Bước 2. Tạo SQL thực hiện tiến trình: Xử lý cơ bản (xử lý cơ bản được thực hiện trong Stagging)

Bước 3. Tạo SQL thực hiện tiến trình: Đưa dữ liệu đã được xử lý cơ bản vào Snapshot

Bước 6. Tạo SQL thực hiện tiến trình: Xử lý nâng cao (được thực hiện trên các bàng Snapshot)

Bước 7. Tạo SQL thực hiện tiến trình: Đưa dữ liệu đã được xử lý nâng cao vào DW