

# LÝ THUYẾT VÀ THỰC HÀNH THIẾT KẾ KHO DỮ LIỆU

Tháng 10 – 2019



Mở đầu .....	4
Phần 1: Kho dữ liệu là gì.....	6
1.1 Kho dữ liệu – Data Warehouse .....	6
1.2 Mô hình hoá kho dữ liệu .....	8
1.2.1 Bảng fact .....	8
1.2.2 Bảng dimension .....	9
1.3 Kiến trúc hệ thống kho dữ liệu .....	12
1.3.1 Tầng Xử lý dữ liệu – ETL .....	14
1.3.2 Tầng kho dữ liệu .....	16
1.3.3 Tầng khai thác dữ liệu .....	17
1.4 Tổng kết .....	17
Phần 2: Thiết kế tổng thể Kho dữ liệu .....	18
2.1 Mô tả hệ thống nghiệp vụ .....	18
2.2 Mô tả yêu cầu người dùng.....	24
2.3 Thiết kế tổng thể.....	25
Phần 3: Xây dựng bảng fact.....	31
3.1 Nguyên tắc chung .....	31
3.2 Bảng fact giao dịch.....	34
3.3 Bảng fact snapshot.....	36
3.4 Bảng fact tổng hợp.....	37
Phần 4: Xây dựng bảng dimension .....	39
4.1 Cấu trúc bảng dimension.....	39
4.2 Phân cấp trong bảng dimension .....	40
4.3 Cập nhật dữ liệu bảng dimension .....	41
4.4 Nguyên tắc thiết kế.....	46
4.5 Một số bảng dimension đặc biệt .....	48
4.5.1 Dimension Ngày tháng.....	48
4.5.2 Dimension có thông tin thay đổi quá nhanh.....	49
4.5.3 Dimension rác.....	50
4.5.4 Dimension từ điển.....	51
4.5.5 Dimension có nhiều bảng vật lý .....	51
4.5.6 Dimension có nhiều thuộc tính đặc thù .....	52
4.5.7 Dimension có phân cấp không giới hạn .....	54

Phần 5: Kinh nghiệm xây dựng – vận hành kho dữ liệu .....	55
5.1 Kết hợp dữ liệu từ nhiều nguồn khác nhau.....	55
5.2 Xử lý bản ghi dimension về trễ .....	55
5.3 Kiểm tra tính đúng đắn của dữ liệu.....	56
5.4 Quản trị dữ liệu.....	56
5.5 Một số kinh nghiệm khác.....	57
Phần 6: Tổng kết.....	58
Tài liệu tham khảo.....	59

## Mở đầu

Chúng ta đang sống trong thời đại của nền kinh tế tri thức. Mọi hoạt động của ta muốn đạt hiệu quả cao thì nhất thiết phải có những phương pháp để có được những thông tin, tri thức cần thiết một cách nhanh chóng, chính xác.

Việc áp dụng công nghệ thông tin vào thực tiễn sản xuất nghiệp vụ đã mang lại những hiệu quả và lợi ích to lớn. Công nghệ ngày càng được phát triển, hoàn thiện hơn để đáp ứng những yêu cầu ngày càng cao của thực tế nghiên cứu, quản lý sản xuất và nghiệp vụ. Sự mở rộng quy mô áp dụng từ những ứng dụng đơn lẻ đến các hệ thống thông tin cỡ lớn đã dẫn đến những thành công vượt bậc trong nghiệp vụ. Các hệ thống thông tin từ chỗ chỉ giải quyết các công việc sự vụ hàng ngày đã tiến tới đáp ứng được các yêu cầu ở mức độ cao hơn, giúp các nhà quản lý điều hành không những biết công việc đang diễn ra thế nào mà còn biết cái gì sẽ xảy ra, nghĩa là thông tin mang tính phân tích và hệ thống thông tin có khả năng hỗ trợ ra quyết định. Tuy nhiên việc xây dựng một hệ thống như vậy vấp phải nhiều hạn chế về mặt kỹ thuật, đặc biệt khi kích thước cũng như độ phức tạp của môi trường thông tin tăng lên. Những hệ thống thông tin xây dựng theo phương pháp truyền thống không làm hài lòng người sử dụng và các nhà quản lý hệ thống thông tin. Giải pháp tốt đẹp nhất đáp ứng được khả năng hỗ trợ kinh doanh đồng thời không làm ảnh hưởng đến hoạt động của hệ thống nghiệp vụ chính là xây dựng một kho dữ liệu.

Ở Việt Nam, nhiều công ty đã có ý thức xây dựng hệ thống kho dữ liệu tập trung phục vụ hỗ trợ công tác dự báo, phân tích và ra quyết định kinh doanh. Không may, một phần không nhỏ trong số đó tiếp cận công nghệ kho dữ liệu như một hệ thống báo cáo thông thường: dữ liệu được tập trung, xử lý đơn giản thành các bảng báo cáo hiển thị cho người dùng cuối. Cách làm này có nhược điểm là khó mở rộng, không mềm dẻo và gây bất tiện cho nhân viên khai thác dữ liệu sau một thời gian sử dụng.

Tài liệu này được viết ra nhằm giới thiệu sơ lược các khái niệm chung nhất của một hệ thống kho dữ liệu, các bước thiết kế một kho dữ liệu bao gồm ví dụ minh họa mô hình hoá một hệ thống. Tài liệu này dành cho các bạn lập trình viên, business analyst, data analyst, devops... có kiến thức cơ bản về database muốn tìm hiểu kỹ hơn về kho dữ liệu. Trong tài liệu có thể có các thuật ngữ, câu lệnh SQL..., sẽ được trình bày đơn giản nhất có thể.

Trong tài liệu này, tác giả sử dụng hình tượng con ếch luộc làm ẩn dụ cho việc thiết kế không tốt gây khó khăn cho người sử dụng kho dữ liệu. Con ếch cho vào nồi nước nóng quá sẽ nhảy ra ngay lập tức, nhưng nếu cho con ếch vào nồi nước lạnh rồi luộc từ từ, con ếch sẽ nằm im chịu chết. Trong ngữ cảnh tài liệu này, người phân tích dữ liệu chính là con ếch và nguyên tắc người sử dụng phải ghi nhớ là nhiệt độ cái nồi. Nếu thiết kế yêu cầu người dùng phải ghi nhớ quá nhiều nguyên tắc, sớm muộn gì họ cũng mắc sai lầm (con ếch đã chết!).

Bố cục tài liệu chia làm 4 phần chính. Phần 1 (Kho dữ liệu là gì?) cung cấp khái niệm một hệ thống kho dữ liệu, các module cơ bản nhất của một hệ thống kho dữ liệu và tính năng của từng module. Phần 2 (Thiết kế tổng thể kho dữ liệu) giới thiệu các bước thiết kế tổng quan một kho dữ liệu. Nội dung về bảng fact, các loại bảng fact và các phương án thiết kế bảng fact được giới thiệu ở phần 3 (Xây dựng bảng fact). Cuối cùng, phần 4 (Xây dựng bảng dimension) đi vào chi tiết việc thiết kế các bảng dimension.

Tác giả rất mong muốn nhận được đóng góp, phản biện của bạn đọc. Mọi ý kiến vui lòng gửi đến địa chỉ email [DF.thangld@hotmail.com](mailto:DF.thangld@hotmail.com).

# Phần 1: Kho dữ liệu là gì

## 1.1 Kho dữ liệu – Data Warehouse

Kinh tế khó khăn, đối thủ càng nhiều thì việc phân tích dữ liệu càng trở nên quan trọng đối với doanh nghiệp nhằm hỗ trợ ra quyết định, gia tăng lợi thế cạnh tranh. Tuy nhiên cơ sở dữ liệu thông thường lại không thỏa mãn các yêu cầu về phân tích dữ liệu, chỉ lưu trữ dữ liệu chi tiết cho thời điểm hiện tại, có hiệu năng kém khi truy vấn phức tạp (join nhiều bảng dữ liệu với nhau) hoặc khối lượng dữ liệu lớn. Thêm nữa, cơ sở dữ liệu thông thường không giải quyết được bài toán tổng hợp dữ liệu từ nhiều nguồn khác nhau, ở những format khác nhau. Để giải quyết được các vấn đề trên cần một hướng tiếp cận khác, công nghệ khác.

Có cung thì tất có cầu, ngay từ những năm 70 nhiều công ty đã bán các hệ thống cơ sở dữ liệu hỗ trợ phân tích, báo cáo như Teradata, MAPPER... Nhưng thuật ngữ “kho dữ liệu” chỉ được sử dụng vào năm 1988 trong một bài báo kỹ thuật của IBM có tiêu đề “Kiến trúc hệ thống thông tin và kinh doanh” (An architecture for a business and information system<sup>1</sup>). Phần này sẽ dành riêng để nói về khái niệm kho dữ liệu.

Theo wikipedia<sup>2</sup>, kho dữ liệu (data warehouse) chính là **hệ quản trị cơ sở dữ liệu chuyên dùng** cho tạo **báo cáo** và **phân tích dữ liệu**. Nó vừa hỗ trợ các truy vấn phức tạp, vừa là điểm tập trung dữ liệu từ nhiều nguồn khác nhau để có được thông tin phân tích đầy đủ nhất. Theo đó, kho dữ liệu là một tập hợp dữ liệu hướng chủ đề, toàn vẹn, không bị rò rỉ mất mát và có giá trị lịch sử. Cụ thể các tính chất đó như sau:

- **Tính hướng chủ đề (Subject – oriented)** nghĩa là kho dữ liệu tập trung vào việc phân tích các yêu cầu quản lý ở nhiều cấp độ khác nhau trong quy trình ra quyết định. Các yêu cầu phân tích này thường rất cụ thể, và xoay quanh loại hình kinh doanh của doanh nghiệp, ví dụ các công ty phân phối sẽ quan tâm đến tình hình kinh doanh, doanh nghiệp viễn thông quan tâm đến lưu lượng dịch vụ... Tuy nhiên một doanh nghiệp thường quan tâm đến vài chủ đề khác nhau, như công ty phân phối còn phải quan tâm đến kho bãi, chuỗi cung ứng...
- **Tính toàn vẹn (Integrated)** giải quyết các khó khăn trong việc kết hợp dữ liệu từ nhiều nguồn dữ liệu khác nhau, giải quyết các sai khác về tên trường dữ liệu (dữ liệu khác nhau nhưng tên giống nhau), ý nghĩa dữ liệu (tên giống nhau nhưng dữ liệu khác nhau), định dạng dữ liệu (tên và ý nghĩa giống nhau nhưng kiểu dữ liệu khác nhau).
- **Tính bất biến (Nonvolatile)** quy định rằng dữ liệu phải thống nhất theo thời gian (bằng cách hạn chế tối đa sửa đổi hoặc xóa dữ liệu), từ đó làm tăng quy mô dữ liệu lên đáng kể so với hệ thống nghiệp vụ (5-10 năm so với 2 đến 6 tháng như cơ sở dữ liệu thông thường)
- **Giá trị lịch sử (time – varying)** nói về khả năng lấy các giá trị khác nhau của cùng một thông tin và thời điểm xảy ra thay đổi. Ví dụ thông tin địa chỉ, email, số điện thoại của

<sup>1</sup> <http://altaplana.com/ibmsj2701G.pdf>

<sup>2</sup> [http://en.wikipedia.org/wiki/Data\\_warehouse](http://en.wikipedia.org/wiki/Data_warehouse)

khách hàng có thể thay đổi, nhưng việc thay đổi đó không được phép tác động đến giá trị báo cáo, phân tích thực hiện trước khi sự thay đổi xảy ra.

Tính chất	Cơ sở dữ liệu nghiệp vụ	Kho dữ liệu
Người dùng	Nhân viên vận hành	Cán bộ quản lý, nhân viên phân tích số liệu
Loại hình sử dụng	Dự đoán được, tác vụ lặp đi lặp lại	Truy vấn đột xuất, không xác định trước
Dữ liệu	Hiện tại, ở mức chi tiết	Lịch sử, ở mức tổng hợp
Tổ chức dữ liệu	Theo yêu cầu nghiệp vụ	Theo vấn đề cần phân tích
Cấu trúc dữ liệu	Tối ưu cho các giao dịch nhỏ	Tối ưu cho truy vấn phức tạp, trên lượng dữ liệu lớn
Tần suất truy cập	Tần suất cao	Tần suất từ trung bình đến thấp
Loại truy cập	Đọc, ghi, cập nhật, xoá	Đọc, ghi
Số lượng bản ghi mỗi phiên truy cập	Ít	Rất lớn
Thời gian truy cập xử lý hoặc truy vấn dữ liệu	Ngắn	Tương đối dài (đến mức phút hoặc tiếng đồng hồ)
Mức độ xử lý song song	Cao, các tác vụ xử lý đồng thời trên một bản ghi nhất định xảy ra thường xuyên	Thấp
Khoá	Cần thiết	Không cần thiết
Tần suất cập nhật dữ liệu	Thường xuyên	Không cập nhật
Dư thừa dữ liệu	Thấp (bảng đã chuẩn hoá)	Cao (bảng dữ liệu thường phẳng hoá về dạng chuẩn 2)
Mô hình dữ liệu	Mô hình quan hệ thực thể (Entity Relational)	Mô hình dữ liệu đa chiều (multidimensional)
Mô hình triển khai	Toàn bộ hệ thống	Tăng dần theo phân khu dữ liệu

**Bảng 1.1:** So sánh cơ sở dữ liệu nghiệp vụ và kho dữ liệu

Kho dữ liệu cho phép người dùng ở mức quản lý, ra quyết định thực hiện các phép phân tích tương tác với data bằng hệ thống **xử lý phân tích trực tuyến (online analytical processing – OLAP)**. Ngoài ra kho dữ liệu cũng được dùng cho báo cáo, data mining và phân tích thống kê. Database và kho dữ liệu, do đó chỉ khác nhau về mặt khái niệm, một cơ sở dữ liệu nếu dùng riêng cho các mục đích trên cũng được coi là kho dữ liệu.

Như vậy, nếu như cơ sở dữ liệu được ví như cái tủ sách cá nhân, nơi người ta thường xuyên tra cứu, cập nhật, hiệu đính, ghi chú vào lề, thêm mới hoặc chuyển sách đi, thì kho dữ liệu lại được so sánh với thư viện quốc gia, nơi các tài liệu kinh điển được đưa đến liên tục để lưu trữ và tham khảo, không ai sửa chữa hoặc chuyển chúng qua chỗ nào khác cả. Theo wikipedia, *kho dữ liệu là một bộ máy hoạt động thu thập, làm sạch, chuẩn hoá dữ liệu nghiệp vụ thành dữ liệu đa chiều, đồng thời tạo ra các câu truy vấn giúp phân tích dữ liệu nhằm hỗ trợ ra quyết định.*

Nên phân biệt hệ thống kho dữ liệu và hệ thống kinh doanh thông minh. Hệ thống kinh doanh thông minh (**Business Intelligence System**) là tập hợp các lý thuyết, các phương pháp, quy trình, kiến trúc và công nghệ để chuyển đổi dữ liệu thô từ hệ thống nghiệp vụ thành thông tin có ý nghĩa, hữu ích cho các mục đích kinh doanh. Từ hai định nghĩa trên, có thể thấy hệ thống kho dữ liệu chính là hệ thống kinh doanh thông minh nhưng được bổ sung một kho dữ liệu làm nơi lưu trữ.

## 1.2 Mô hình hoá kho dữ liệu

Mô hình dữ liệu trong kho dữ liệu khác biệt rất nhiều so với mô hình hệ thống nghiệp vụ. Có 2 loại thực thể chính trong kho dữ liệu là bảng **fact** và bảng **dimension**.

### 1.2.1 Bảng fact

Bảng fact là thực thể quan trọng nhất trong mô hình kiến trúc kho dữ liệu. Nó là nơi lưu trữ các số liệu về một sự kiện nào đó hoặc một phép đo tình hình hoạt động của tổ chức theo những chu kỳ thời gian nhất định. Ví dụ một bảng fact được thể hiện trên hình 1.1:

**Bảng fact dạng giao dịch**

Transaction Fact	
FK	date_dim_id
FK	time_dim_id
FK	shop_dim_id
FK	client_dim_id
	transaction_code
	sale_amount
	tax_amount
	total_amount

**Bảng fact dạng snapshot**

Balance Fact	
FK	date_dim_id
FK	client_dim_id
	balance

**Bảng fact tổng hợp**

Daily Sale Fact	
FK	date_dim_id
FK	shop_dim_id
	transaction_count
	total_sale_amount
	total_tax_amount
	total_amount
	percent_revenue

**Hình 1.1:** Ví dụ bảng fact

Các cột dữ liệu trong bảng fact có thể phân ra làm 3 nhóm khác nhau:

- Các cột số liệu là các cột có ích và dễ sử dụng nhất. Các cột này là các cột số liệu như thành tiền của hoá đơn, tiền khách thanh toán, tiền thuế...
- Các cột khoá phụ liên kết đến dimension



- Các cột có liên quan nhưng không phải thông tin fact, ví dụ bảng fact giao dịch có thêm trường kiểu giao dịch `transaction_type`, trường mô tả `description`... Các cột dữ liệu này không có nhiều ý nghĩa trong hoạt động báo cáo, thống kê, machine learning... nên không được khuyến khích lưu vào bảng fact. Tuy nhiên các cột này có ích trong trường hợp kho dữ liệu có thêm nhiệm vụ hỗ trợ hệ thống nghiệp vụ.

Có 3 loại bảng fact chính: bảng fact giao dịch, bảng fact snapshot và bảng fact tổng hợp (aggregated fact):

- Bảng **fact giao dịch (transactional fact)** mô tả một sự việc nào đó trong hoạt động nghiệp vụ của tổ chức. Các bảng fact này cung cấp dữ liệu ở mức chi tiết nhất, là đầu vào cho các bảng fact tổng hợp có độ chi tiết thấp hơn.
- Bảng **fact snapshot (snapshot fact)** định kỳ lưu trạng thái nào đó của đối tượng quan tâm theo những khoảng thời gian nhất định như số dư tài khoản, cấp độ tài khoản, các phép đo nhiệt độ, chiều cao...
- Bảng **fact tổng hợp (aggregated fact)** tính toán sẵn các chỉ tiêu có liên quan đến một nhóm dimension nào đó, giúp thuận lợi hơn cho quá trình phân tích và hiển thị báo cáo. Không như hai loại bảng fact giao dịch và fact snapshot, bảng fact tổng hợp được thiết kế theo yêu cầu và nhu cầu sử dụng của người phân tích.

Ngoài 3 loại bảng fact trên, đôi khi ta bắt gặp một loại bảng thứ tư gọi là factless fact. Các bảng fact dạng này không có các cột cộng dồn hay bán cộng dồn mà chỉ có các cột không cộng dồn được. Các bảng fact dạng này thường dùng để ghi nhận số lần xuất hiện một hành động nào đó (tương tự transactional fact) như thông tin đăng ký làm thành viên trang web; hoặc ghi nhận các đối tượng có sẵn trên hệ thống (tương tự snapshot fact) như danh sách thành viên đã đăng ký trang web. Phép toán thống kê thực hiện trên bảng này là phép đếm.

### 1.2.2 Bảng dimension

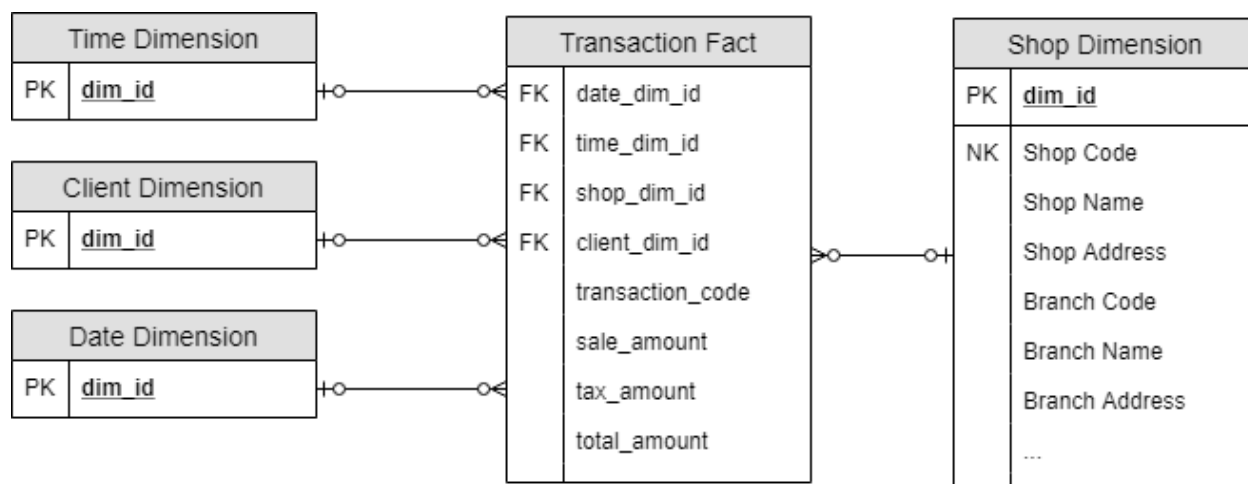
Trong hệ thống kho dữ liệu, bảng dimension đóng vai trò trợ giúp, cung cấp ngữ cảnh cho các bảng fact. Nếu như bảng fact giao dịch thể hiện các chỉ tiêu tiền hoá đơn, tiền khách trả, tiền thuế... thì bảng dimension cung cấp thông tin trả lời cho câu hỏi ai mua, mua ở đâu, mua lúc nào... Bảng dimension thường có số lượng cột rất lớn, lên đến hàng trăm hoặc vài trăm cột dữ liệu.

Product Dimension	
PK	<u>dim_id</u>
NK	SKU
	Name
	Description
	Brand
	Package Type
	Package Size
	Weight
	Unit type
	---

**Hình 1.2:** Ví dụ bảng dimension

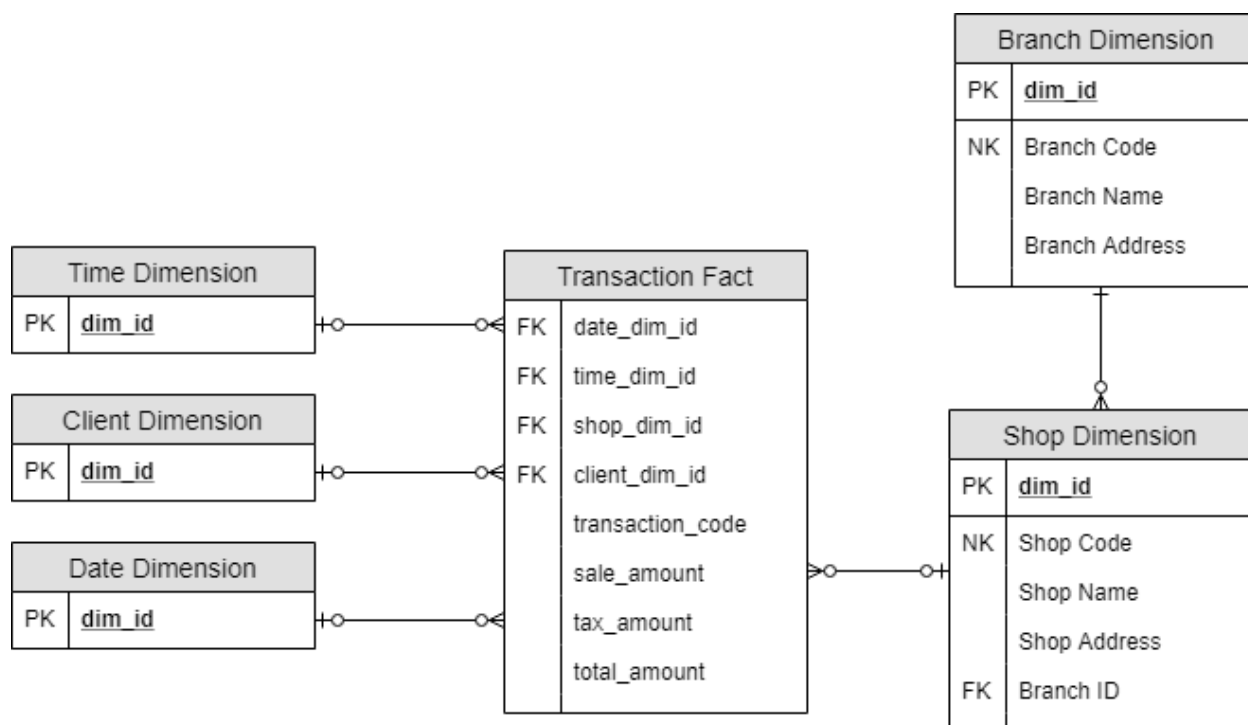
Dữ liệu trong bảng dimension cũng thường hàm chứa một hoặc nhiều quan hệ phân cấp. Trong hình 1.2 ở trên, nhiều sản phẩm có thể được gom nhóm theo nhãn hàng (Brand), theo cách đóng gói (Package Type) hoặc theo đơn vị tính (Unit type). Tùy theo phương án thiết kế, các cấp bậc dữ liệu cao hơn có thể được lưu trực tiếp vào bảng dimension hoặc chia nhỏ ra thành nhiều bảng dimension khác nhau. Hai phương án thiết kế đó được gọi là thiết kế hình ngôi sao (star schema) và thiết kế hình bông tuyết (snowflake schema).

Theo **thiết kế hình sao (star schema)**, các bảng dimension được phi chuẩn hoá về dạng chuẩn 2 của thiết kế cơ sở dữ liệu, các thực thể có cấu trúc phân cấp được nhập chung vào làm một. Khi đó mô hình dữ liệu của một bảng fact trông có vẻ giống hình ngôi sao. Ví dụ thiết kế hình sao được thể hiện ở hình 1.3, trong đó dimension Cửa hàng (Shop Dimension) vừa có các thông tin về cửa hàng (trường dữ liệu có tiền tố Shop), vừa có các thông tin về chi nhánh (trường dữ liệu có tiền tố Branch). Khi đó các cửa hàng được quản lý chung bởi một chi nhánh sẽ có giá trị các trường thông tin chi nhánh giống hệt nhau.



Hình 1.3: Lược đồ hình sao

**Thiết kế hình bông tuyết (snowflake schema)** giảm bớt dư thừa dữ liệu trong lược đồ hình sao bằng cách chuẩn hoá các bảng dimension về dạng chuẩn 3 thiết kế cơ sở dữ liệu. Khi đó, một thực thể dimension có phân cấp sẽ được thể hiện thành nhiều bảng dữ liệu khác nhau, mỗi bảng đại diện cho một cấp. Hình 1.4 thể hiện ví dụ một thiết kế theo hình bông tuyết, trong đó dimension Cửa hàng được thể hiện qua 2 bảng dữ liệu Cửa hàng (Shop Dimension) và Chi nhánh (Branch Dimension).



Hình 1.4: Lược đồ bông tuyết

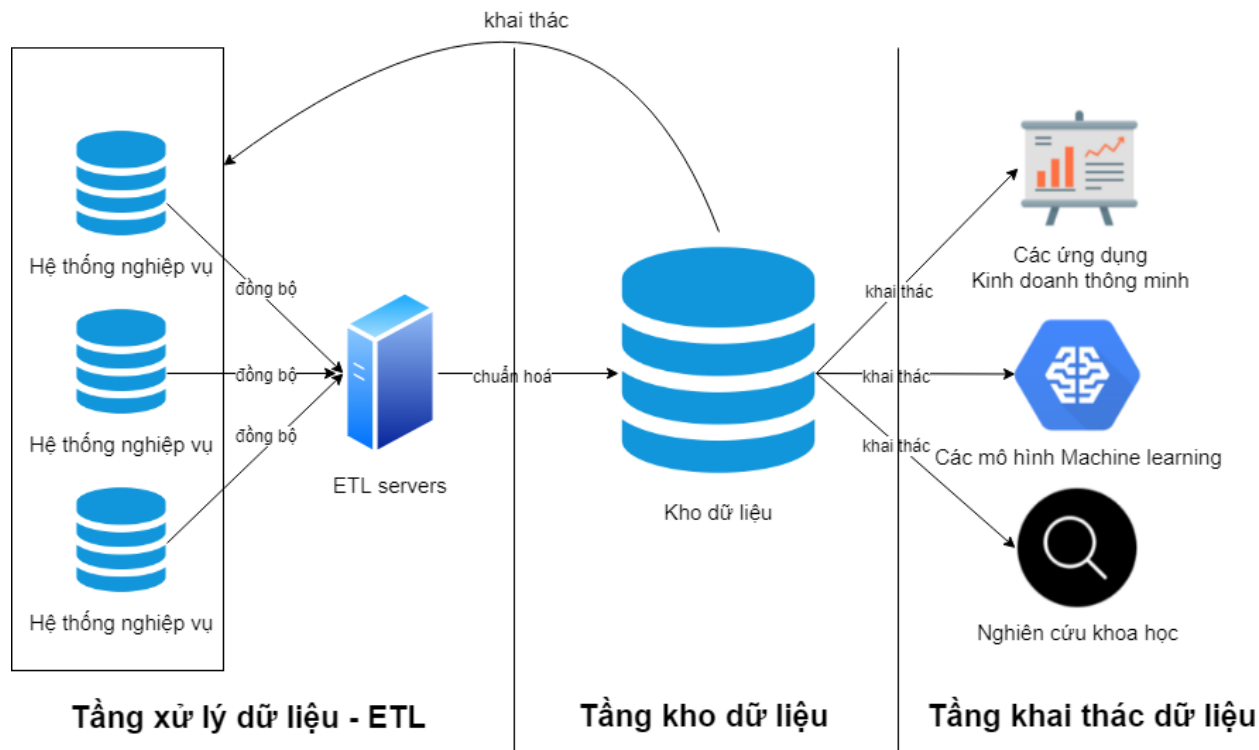
Dựa vào mô tả 2 thiết kế trên, ta có thể thấy ngay thiết kế theo hướng hình sao có số lượng bảng dimension nhỏ nên rất đơn giản, dễ hiểu, hiệu năng truy vấn cũng cao hơn vì chỉ cần thực hiện một phép join để lấy tất cả thông tin dimension. Ngược lại, thiết kế hình bông tuyết được chuẩn hoá chặt chẽ nên tránh được tình trạng dư thừa dữ liệu, hạn chế tình trạng không nhất quán về dữ liệu.

Theo quan điểm tác giả, các bảng dimension có thể có số lượng cột dữ liệu rất lớn, nhưng số lượng bản ghi lại rất nhỏ so với bảng fact. Thông thường tổng dung lượng các bảng dimension chỉ chiếm khoảng 10% dung lượng kho dữ liệu. Do có dung lượng ít như vậy, việc chuẩn hoá để tiết kiệm là không cần thiết, nhất là trong thời đại hiện nay, chi phí cho việc lưu trữ đã rất rẻ so với những năm 1990. Chính vì thế tác giả luôn ưu tiên áp dụng thiết kế hình sao cho công việc của mình.

### 1.3 Kiến trúc hệ thống kho dữ liệu

Một hệ thống kho dữ liệu bao gồm 3 thành phần chính sau:

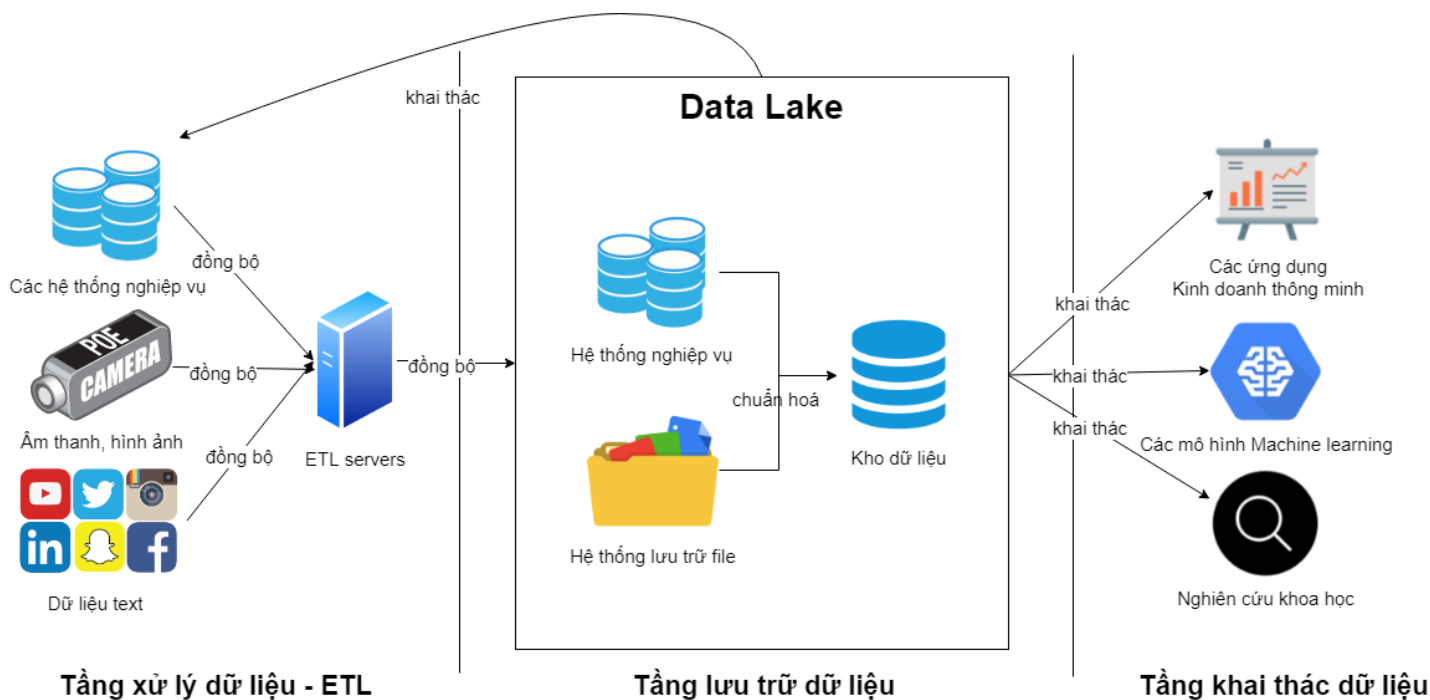
- Một bộ công cụ để **thu thập** dữ liệu từ hệ thống nghiệp vụ, **chuẩn hoá** chúng về định dạng dữ liệu đa chiều,  **nạp** vào kho dữ liệu (**Extract-Transformation-Loading – xử lý dữ liệu dữ liệu**).
- Một cơ sở dữ liệu dùng làm **kho dữ liệu** để lưu trữ dữ liệu
- Một loạt các công cụ khai thác dữ liệu từ kho dữ liệu như hệ thống OLAP, hệ thống báo cáo tính, hệ thống data mining...



Hình 1.5: Kiến trúc hệ thống Data warehouse

Khoảng 10 năm trở lại đây mọi người dần dần nói nhiều đến thuật ngữ Hồ Dữ liệu (Data Lake). Thuật ngữ do ông James Dixon, Giám đốc kỹ thuật công ty Pentaho đưa ra trong một bài phỏng vấn khoảng giữa năm 2011. Theo ông James, tại thời điểm thu thập dữ liệu, thường chúng ta chỉ mô hình hoá dữ liệu theo nhu cầu tại thời điểm thu thập, đôi khi làm hạn chế hoặc không cho phép khai thác theo các nhu cầu phát sinh sau này. Do đó phát sinh ra hệ thống data lake, là hệ thống có tác dụng lưu trữ dữ liệu ở định dạng gốc, bao gồm cả dữ liệu có cấu trúc (dòng/cột của hệ thống nghiệp vụ), bán cấu trúc (file XML, JSON, HTML...), không có cấu trúc (text), dữ liệu binary (hình ảnh, video, file âm thanh...).

Theo quan điểm của tác giả, một hệ thống data lake như vậy rất khó khai thác và làm giảm hiệu quả làm việc của người phân tích dữ liệu và người xây dựng mô hình machine learning. Một công việc rất quan trọng của 2 nhóm người trên là chuẩn hoá dữ liệu trước khi tiến hành phân tích hoặc train mô hình, tuy nhiên công việc này thiên về mặt engineer, đòi hỏi những kỹ năng hoàn toàn khác. Do đó ngoài việc lưu trữ nguyên gốc, dữ liệu trên hệ thống data lake nên được xử lý, chuẩn hoá trước khi đến tay nhóm phân tích. Như vậy dữ liệu trong data lake có thể dùng làm đầu vào cho kho dữ liệu được mô hình hoá, và người dùng ngoài có thêm lựa chọn khai thác dữ liệu thô từ đầu nguồn. Mô hình kiến trúc hệ thống data lake theo quan điểm tác giả được mô tả trong hình 1.6



**Hình 1.6:** Kiến trúc hệ thống Data lake

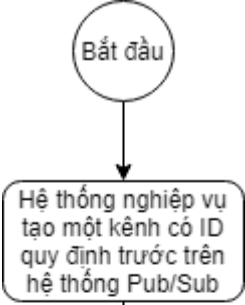
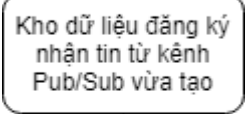
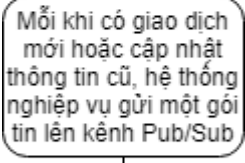
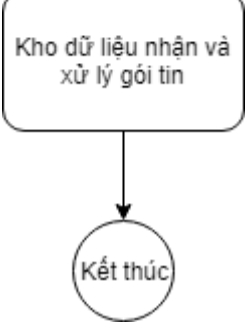
Trong phạm vi tài liệu này chỉ tập trung nghiên cứu hệ thống kho dữ liệu. Chi tiết các module trong hệ thống sẽ được mô tả chi tiết hơn trong các phần sau.

### 1.3.1 Tầng Xử lý dữ liệu – ETL

Tầng Xử lý dữ liệu (**ETL: Extract – Transform – Load**) là tầng thấp nhất, ẩn đi với người dùng cuối, bao gồm 3 bước:

- Bước **thu thập (extract)** gom góp dữ liệu từ nhiều khác nhau về. Các nguồn này có thể là cơ sở dữ liệu hệ thống nghiệp vụ (MS SQL, mySQL, Oracle, DB2...), cũng có thể là file ở các định dạng khác nhau (CSV, fix-length, excel, XML...), có thể là dữ liệu nội bộ doanh nghiệp hoặc từ bên ngoài. Một hệ thống xử lý dữ liệu tốt phải đảm bảo tương thích với các nguồn dữ liệu thông dụng này.
- Bước **chuẩn hoá (transform)** biến đổi dữ liệu từ định dạng nguồn sang định dạng của kho dữ liệu (định dạng dữ liệu đa chiều đã nói ở bước trên), bao gồm các bước nhỏ:
  - o Bước **dọn dẹp (cleaning)** xoá các bản ghi bị sai, lỗi và chuyển hoá dữ liệu về định dạng chuẩn chung.
  - o Bước **tập hợp (integration)** cắt gọt dữ liệu có chung ý nghĩa từ nhiều nguồn khác nhau về một khung duy nhất.
  - o Bước **tổng hợp (aggregation)** tổng hợp dữ liệu dựa vào độ chi tiết của kho dữ liệu.
- Bước  **nạp dữ liệu (load)** ghi dữ liệu đã được chuẩn hoá vào kho dữ liệu. Bước này bao gồm cả quá trình cập nhật thay đổi từ hệ thống nghiệp vụ vào kho dữ liệu, đảm bảo số liệu báo cáo luôn được cập nhật. Tùy thuộc vào chính sách công ty, việc cập nhật này có thể phải thực hiện theo thời gian thực, cập nhật theo giờ, theo ngày hoặc thậm chí theo tháng.

Trong những hệ thống kho dữ liệu hiện đại, việc xử lý chuẩn hoá không chỉ thực hiện theo lô mà còn chạy trong thời gian thực (hoặc tương đương, độ trễ dữ liệu ở mức giây). Khi đó hệ thống nghiệp vụ làm thêm việc gửi các bản ghi thay đổi sang kho dữ liệu bằng cách sử dụng một hệ thống truyền tin nào đó, có thể là phương pháp Publish – Subscribe (Pub/Sub). Quy trình thực hiện như sau:

STT	Nơi thực hiện	Bước thực hiện	Mô tả
1	Hệ thống nghiệp vụ	 <pre> graph TD     Start((Bắt đầu)) --&gt; Step1[Hệ thống nghiệp vụ tạo một kênh có ID quy định trước trên hệ thống Pub/Sub] </pre>	Trước khi xây dựng hệ thống cần xác định danh sách các kênh Pub/Sub sẽ tạo. Tùy phương án xử lý có thể tạo một hoặc nhiều kênh khác nhau, mỗi kênh truyền một loại đối tượng dữ liệu nhất định
2	Kho dữ liệu	 <pre> graph TD     Step1 --&gt; Step2[Kho dữ liệu đăng ký nhận tin từ kênh Pub/Sub vừa tạo] </pre>	Kho dữ liệu cần được subscribe với tất cả các kênh đã tạo ở bước 1
3	Hệ thống nghiệp vụ	 <pre> graph TD     Step2 --&gt; Step3[Mỗi khi có giao dịch mới hoặc cập nhật thông tin cũ, hệ thống nghiệp vụ gửi một gói tin lên kênh Pub/Sub] </pre>	Dữ liệu trên hệ thống truyền tin ở dạng text, có thể format theo chuẩn JSON hoặc XML hoặc một định dạng nào đó quy định trước, có thể được mã hoá. Nội dung dữ liệu thường gồm tên đối tượng được cập nhật, kiểu hành động cập nhật (thêm/sửa/xoá...), thời gian phát sinh hành động và các thông tin liên quan đến đối tượng.
4	Kho dữ liệu	 <pre> graph TD     Step3 --&gt; Step4[Kho dữ liệu nhận và xử lý gói tin]     Step4 --&gt; End((Kết thúc)) </pre>	Sau khi giải mã (nếu cần), kho dữ liệu thực hiện các phép chuẩn hoá tương ứng với đối tượng và kiểu hành động nhận được rồi ghi dữ liệu mới vào kho.

**Bảng 1.2:** Phương án xây dựng ETL theo hướng real-time

Có một số chú ý khi xây dựng hệ thống kho dữ liệu:

- Hệ thống Pub/Sub chỉ đảm bảo các ứng dụng subscribe nhận được dữ liệu trong một khoảng thời gian nhất định (theo cấu hình), sau đó không cố gửi nữa. Nếu hệ thống kho dữ liệu ngừng xử lý trong thời gian “dài” (lớn hơn mức cấu hình), các bản ghi được gửi sẽ bị mất.

- Hệ thống Pub/Sub không đảm bảo được số thứ tự nhận bản ghi, đôi khi giao dịch thực hiện sau lại được xử lý trước, đôi khi giao dịch xử lý không liên kết được với dimension vì bản ghi dimension chưa đến.
- Đôi khi một bản tin publish được gửi đến subscriber 2 lần làm bản ghi bị lặp.
- Hệ thống nghiệp vụ được ưu tiên tối đa cho các hoạt động nghiệp vụ. Trong trường hợp cao tải, hệ thống nghiệp vụ có thể hạn chế tài nguyên hoặc ngừng hẳn việc gửi dữ liệu.

Do hạn chế về mặt thời gian, hệ thống kho dữ liệu thời gian thực thường không thực hiện các phép chuẩn hoá phức tạp mà chỉ xử lý đơn giản. Các tính toán phức tạp hơn vẫn sẽ được xử lý theo lô theo định kỳ.

### 1.3.2 Tầng kho dữ liệu

Tầng kho dữ liệu đứng ở trung tâm một hệ thống kho dữ liệu làm nhiệm vụ lưu trữ dữ liệu bao quanh tất cả các hoạt động nghiệp vụ, các phòng ban của doanh nghiệp. Kho dữ liệu thường bao gồm một hoặc nhiều **phân khu dữ liệu (data mart)**, với mỗi phân khu dữ liệu chính là kho dữ liệu thu nhỏ tập trung vào một nghiệp vụ nhất định nào đó của doanh nghiệp (ví dụ phân khu dữ liệu về bán hàng, phân khu dữ liệu về kho bãi, phân khu dữ liệu về nhân sự...).

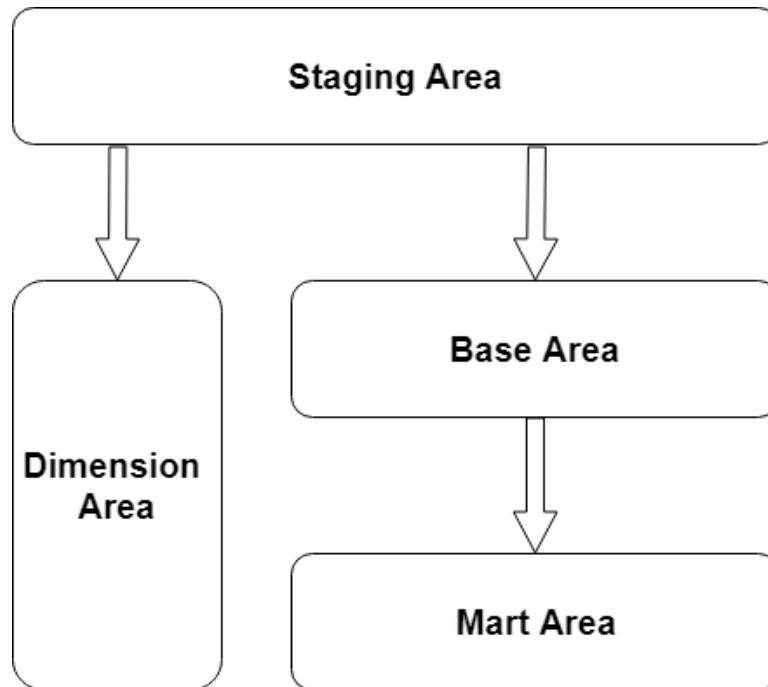
Có 3 loại bảng khác nhau trong một hệ thống kho dữ liệu:

- Các bảng chứa dữ liệu đồng bộ từ hệ thống nghiệp vụ: Tùy phương án thiết kế, các bảng này có thể lưu toàn bộ dữ liệu hoặc chỉ dữ liệu mới phát sinh trong một khoảng thời gian nhất định. Các bảng ở khu vực này dùng làm đầu vào cho tiến trình xử lý chuẩn hoá dữ liệu.
- Các bảng fact lưu các giá trị đo đếm được của một giao dịch, một sự kiện, một đối tượng nào đó. Chi tiết về bảng fact và cách thiết kế bảng fact sẽ được mô tả ở các phần sau.
- Các bảng dimension lưu các thông tin làm ngữ cảnh cho bảng fact. Chi tiết về bảng dimension và cách thiết kế bảng dimension sẽ được mô tả ở các phần sau.

Về mặt logic, một kho dữ liệu được chia làm 4 khu vực khác nhau như hình 1.9:

- Staging Area: chứa các bảng tạm, các bảng đồng bộ dữ liệu từ hệ thống nghiệp vụ. Đây là nguồn đầu vào cho các bảng ở khu vực Dimension Area và Base Area
- Dimension Area: chứa các bảng dimension
- Base Area: chứa các bảng fact đã được chuẩn hoá ở cấp độ thấp, ghi nhận dữ liệu ở mức sự kiện, phép đo. Các bảng trong khu vực này nhận dữ liệu từ khu vực Staging Area và làm đầu vào cho khu vực Mart Area
- Mart Area: chứa các bảng fact đã được tổng hợp ở mức cao, thường là các bảng tổng hợp. Các bảng trong khu vực này nhận dữ liệu từ khu vực Base Area và làm đầu vào cho các hệ thống báo cáo.





Hình 1.7: Mô hình logic các thực thể trong kho dữ liệu

### 1.3.3 Tầng khai thác dữ liệu

Tầng khai thác dữ liệu chứa các công cụ cho người dùng cuối khai thác, sử dụng các dữ liệu trong kho dữ liệu. Một số công cụ chính:

- **Các ứng dụng Kinh doanh thông minh (Business Intelligence)** thường bao gồm một hệ thống báo cáo có độ linh hoạt cao, cho phép người dùng tự xây dựng biểu mẫu báo cáo, biểu đồ... theo nhu cầu phân tích một cách đơn giản, tiện lợi.
- Bộ công cụ **khai phá dữ liệu (machine learning)** cho phép người dùng phân tích dữ liệu để tìm ra các quy luật nào đó ẩn trong dữ liệu chưa được mọi người phát hiện ra. Công cụ cho người khai phá dữ liệu là các thuật toán machine learning, deep learning.
- Ngoài ra, dữ liệu trên hệ thống data warehouse có thể làm đầu vào, tác động ngược lại hệ thống nghiệp vụ.

## 1.4 Tổng kết

Trong phần này, chúng ta đã tìm hiểu qua về một hệ thống kho dữ liệu, các cấu phần hình thành nên kho dữ liệu. Ngoài các định nghĩa cổ điển, chúng ta cũng đã thảo luận qua các ý tưởng tương đối mới như khái niệm Data Lake và phương án xây dựng ETL theo thời gian thực. Chúng ta cũng mô tả được mô hình dữ liệu hệ thống phần mềm một doanh nghiệp hoạt động bán lẻ để thực hành thiết kế.

Thiết kế tổng thể hệ thống kho dữ liệu sẽ được trình bày chi tiết hơn ở phần 2.

## Phần 2: Thiết kế tổng thể Kho dữ liệu

Phần 1 của tài liệu đã mô tả kho dữ liệu, mục đích, ý nghĩa và kiến trúc kho dữ liệu. Phần 2 này sẽ giới thiệu chi tiết hơn các khái niệm và mô tả cách thiết kế tổng thể một kho dữ liệu hoàn chỉnh.

Khi xây dựng hệ thống data warehouse, việc mô hình hoá dữ liệu là cực kỳ quan trọng. Việc thiết kế tốt kho dữ liệu trợ giúp rất nhiều trong quá trình sử dụng kho dữ liệu, giúp cho việc sử dụng dễ dàng hơn, giảm thiểu các bước xử lý của nhân viên phân tích, góp phần hạn chế sai sót. Quan trọng hơn, việc thiết kế mô hình làm giảm sự phụ thuộc của kho dữ liệu vào hệ thống nghiệp vụ, tránh được việc phải thay đổi thiết kế khi hệ thống nghiệp vụ có cập nhật. Một bản thiết kế kho dữ liệu phải đảm bảo được 2 vấn đề sau:

- Việc thay đổi cấu trúc hệ thống nghiệp vụ không làm ảnh hưởng đến kiến trúc và mô hình kho dữ liệu. Nếu tổ chức thay thế hệ thống phần mềm, nhân viên phân tích không cần viết nhiều câu truy vấn khác nhau cho những khoảng thời gian khác nhau (một câu cho hệ thống cũ, một câu cho hệ thống mới).
- Việc thay đổi thông tin một bản ghi dimension không làm ảnh hưởng đến dữ liệu fact có liên quan đến bản ghi đó tại thời điểm trước khi thay đổi.

Như đã trình bày trong mục 1.3.2, các thực thể trong kho dữ liệu được chia làm 4 nhóm khác nhau. Trong đó, các bảng thuộc vùng Staging Area được đồng bộ từ hệ thống nghiệp vụ, các bảng thuộc vùng Dimension Area và Base Area được xây dựng bám theo quy trình nghiệp vụ và mô hình dữ liệu hệ thống nghiệp vụ, có tham khảo ý kiến người sử dụng. Riêng các bảng thuộc vùng Mart Area được thiết kế chủ yếu theo nhu cầu người dùng cuối. Các bảng ở vùng này chính là dữ liệu đầu vào cho hệ thống báo cáo/phân tích/dựng model machine learning.

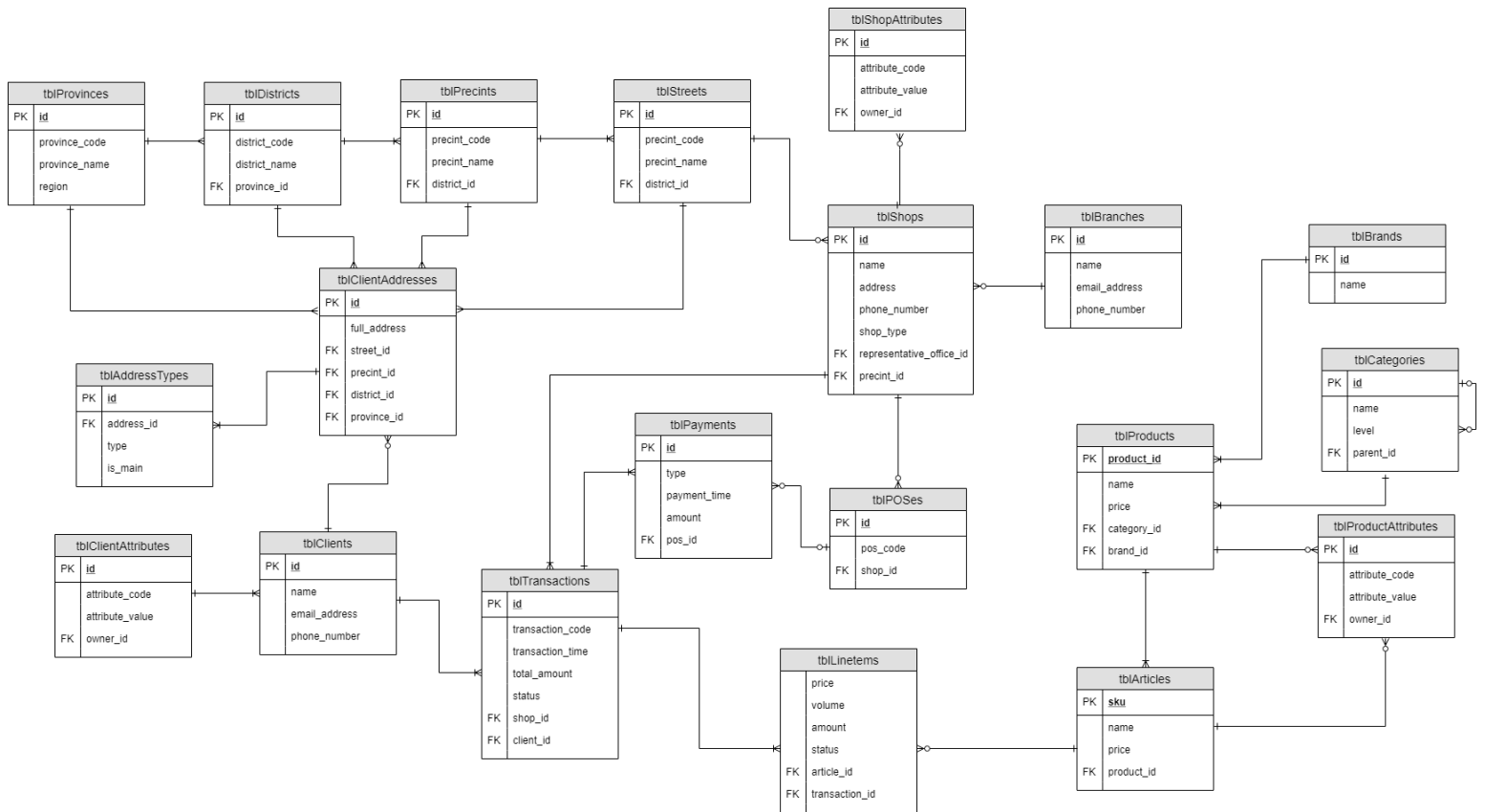
Trước khi thiết kế tổng thể, có 2 công việc quan trọng cần làm là mô tả yêu cầu người dùng và phân tích quy trình nghiệp vụ cũng như mô hình dữ liệu các hệ thống sẽ được mô hình hoá. Các thông tin này có thể được mô tả chi tiết ở tài liệu riêng nhưng cũng nên trình bày sơ lược trong tài liệu này để người đọc nắm được.

### 2.1 Mô tả hệ thống nghiệp vụ

DF Corp là một công ty thương mại vừa kinh doanh siêu thị vừa kinh doanh cửa hàng tiện lợi. Ngoài hệ thống cửa hàng, DF Corp còn có thêm một trang web thương mại điện tử để người dùng có thể đặt hàng tại nhà. Khách hàng sau khi sử dụng dịch vụ ở mức độ nhất định sẽ được nhân viên chăm sóc khách hàng mời đăng ký làm thẻ thành viên để được nhận rất nhiều ưu đãi. Cụ thể 2 ưu đãi chính là tích điểm và hoàn tiền 10% các ngày thứ sáu của tháng nếu tiêu dùng tháng trước đó lớn hơn 5 triệu VNĐ. Ngoài đối tượng khách hàng lẻ, DF Corp còn cung cấp đồ dùng văn phòng cho các công ty và thực phẩm cho các quán ăn, nhà hàng.

Vì nhiều lý do, hệ thống phần mềm của DF Corp bao gồm nhiều phần mềm nhỏ, mỗi phần mềm phục vụ mục đích khác nhau. Mảng quản lý bán hàng của công ty được quản lý bằng một hệ thống ERP nổi tiếng. Trong khi đó, hệ thống khách hàng trung thành lại vận hành trên hệ thống phần mềm do Trung tâm Công nghệ DF Corp tự phát triển.

Để đơn giản, ví dụ chỉ tập trung vào mảng bán hàng, bỏ qua mảng kho bãi, giao vận, nhân sự... Mô hình dữ liệu hệ thống bán hàng DF Corp được mô tả trong hình 1.10



**Hình 1.8:** Mô hình dữ liệu hệ thống quản lý bán hàng DF Corp

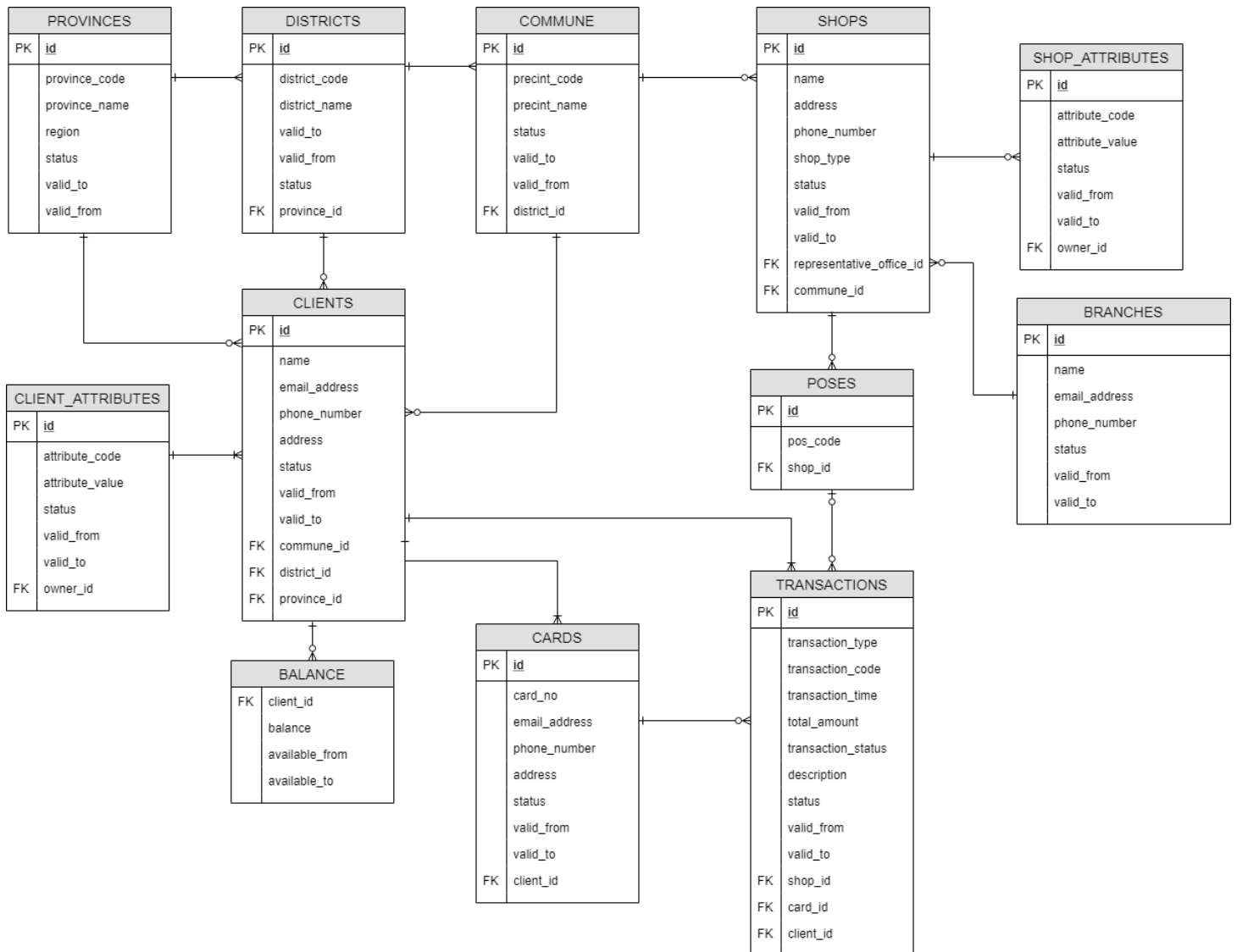
STT	Tên bảng	Mô tả
1	tblStreets	Lưu thông tin địa giới hành chính Việt Nam mức đường phố
2	tblPrecints	Lưu thông tin địa giới hành chính Việt Nam mức phường/xã
3	tblDistricts	Lưu thông tin địa giới hành chính Việt Nam mức quận/huyện
4	tblProvinces	Lưu thông tin địa giới hành chính Việt Nam mức tỉnh/thành phố
5	tblPOSes	Lưu thông tin máy POS tại các siêu thị/cửa hàng tiện lợi. Chú ý: trang thương mại điện tử của DF Corp không có máy POS.
6	tblShops	Lưu thông tin các siêu thị/cửa hàng tiện lợi/trang thương mại điện tử trên hệ thống DF Corp.

7	tblShopAttributes	Lưu thuộc tính của các siêu thị/cửa hàng tiện lợi/trang thương mại điện tử trên hệ thống DF Corp. Bảng thiết kế theo phương pháp Entity – Attribute - Value (EAV)
8	tblBranches	Lưu thông tin văn phòng đại diện quản lý cửa hàng/siêu thị/trang web tương ứng. Ở DF Corp có 4 đơn vị như vậy, 1 đơn vị quản lý trang web, 3 đơn vị đại diện cho 3 miền Bắc, Trung, Nam quản lý các cửa hàng, siêu thị ở miền của mình.
9	tblProducts	Lưu thông tin các sản phẩm bán trên hệ thống DF Corp
10	tblArticles	Ứng với mỗi sản phẩm lại có nhiều lựa chọn khác nhau (màu sắc, kích thước, khối lượng...). Bảng tblArticles lưu thông tin các lựa chọn này.
11	tblProductAttributes	Lưu thuộc tính của các siêu thị/cửa hàng tiện lợi/trang thương mại điện tử trên hệ thống DF Corp. Bảng thiết kế theo phương pháp EAV. Chú ý một thuộc tính có thể liên kết đến bảng tblProducts hoặc bảng tblArticles.
12	tblBrands	Nhãn hiệu hàng hoá
13	tblCategories	Chủng loại hàng hoá. Mỗi chủng loại có thể bao gồm nhiều chủng loại con, phân cấp sâu nhất lên đến 7 tầng. Chú ý: hàng hoá có thể thuộc chủng loại ở tầng thứ tư, thứ năm, thứ sáu hoặc thứ bảy
14	tblClients	Lưu thông tin khách hàng.
15	tblClientAddresses	Lưu các địa chỉ của khách hàng phục vụ thanh toán, giao vận hoặc thống kê. Trên hệ thống có nhiều khách hàng lẻ không cung cấp địa chỉ hoặc chỉ cung cấp rất hạn chế (mức tỉnh thành, phường xã) hoặc không tìm thấy tên đường phố trên hệ thống nên bảng địa chỉ khách hàng cần tham chiếu đến đủ 4 cấp địa giới hành chính.
16	tblAddressTypes	Khách hàng có thể dùng 1 địa chỉ cho 3 mục đích khác nhau là để liên lạc, để thanh toán, để nhận hàng. Bảng này lưu các mục đích sử dụng của khách.
17	tblClientAttributes	Lưu các thuộc tính của khách hàng. Bảng thiết kế theo phương pháp EAV
18	tblTransactions	Lưu thông tin giao dịch của khách. Chú ý: bảng dữ liệu bao gồm cả giao dịch huỷ. Giao dịch huỷ được hệ thống cập nhật trạng thái trường status.

		Khi thanh toán, không phải khách hàng nào cũng ghi lại thông tin cá nhân hoặc đăng ký lên hệ thống. Trường hợp không có khách hàng, trường client_id nhận giá trị null.
19	tblLineItems	Lưu thông tin các sản phẩm của giao dịch. Chú ý: bảng dữ liệu bao gồm cả giao dịch huỷ và trả hàng-hoàn tiền. Giao dịch huỷ được hệ thống cập nhật trạng thái trường status. Giao dịch trả hàng-hoàn tiền được hệ thống thêm một bản ghi ghi nhận số lượng hàng bị hoàn trả với trạng thái trên trường status là “Trả hàng/hoàn tiền”
20	tblPayment	Ghi nhận các khoản thanh toán đơn hàng của khách. Chú ý một đơn hàng có thể được chia nhỏ để thanh toán bằng nhiều phương pháp khác nhau như thẻ ATM, thẻ tín dụng, tiền mặt, điểm khách hàng trung thành.

**Bảng 2.1:** Danh sách bảng dữ liệu hệ thống bán hàng DF Corp

Mô hình dữ liệu hệ thống Khách hàng trung thành (KHTT) được mô tả trong hình 1.11



**Hình 2.1:** Mô hình dữ liệu hệ thống Khách hàng trung thành DF Corp

STT	Tên bảng	Mô tả
1	COMMUNES	Lưu thông tin địa giới hành chính Việt Nam mức phường/xã. Dữ liệu bảng này được đồng bộ từ bảng tblPrecints hệ thống bán hàng.
2	DISTRICTS	Lưu thông tin địa giới hành chính Việt Nam mức quận/huyện. Dữ liệu bảng này được đồng bộ từ bảng tblDistricts hệ thống bán hàng.
3	PROVINCES	Lưu thông tin địa giới hành chính Việt Nam mức tỉnh/thành phố. Dữ liệu bảng này được đồng bộ từ bảng tblProvinces hệ thống bán hàng.

4	SHOPS	Lưu thông tin các siêu thị/cửa hàng tiện lợi/trang thương mại điện tử trên hệ thống DF Corp. Dữ liệu bảng này được đồng bộ từ bảng tblShops hệ thống bán hàng.
5	POSES	Lưu thông tin máy POS trên hệ thống DF Corp. Dữ liệu bảng này được đồng bộ từ bảng tblPOSes hệ thống bán hàng.
6	SHOP_ATTRIBUTES	Lưu thuộc tính của các siêu thị/cửa hàng tiện lợi/trang thương mại điện tử trên hệ thống DF Corp. Bảng thiết kế theo phương pháp EAV. Dữ liệu bảng này được đồng bộ từ bảng tblShopAttributes hệ thống bán hàng.
7	BRANCHES	Lưu thông tin văn phòng đại diện quản lý cửa hàng/siêu thị/trang web tương ứng. Ở DF Corp có 4 đơn vị như vậy, 1 đơn vị quản lý trang web, 3 đơn vị đại diện cho 3 miền Bắc, Trung, Nam quản lý các cửa hàng, siêu thị ở miền của mình. Dữ liệu bảng này được đồng bộ từ bảng tblBranches hệ thống bán hàng.
8	CLIENTS	Lưu thông tin khách hàng. Dữ liệu bảng này được đồng bộ từ bảng tblClients hệ thống bán hàng
9	CLIENT_ATTRIBUTES	Lưu các thuộc tính của khách hàng. Bảng thiết kế theo phương pháp EAV. Dữ liệu bảng này được đồng bộ từ bảng tblClientAttributes hệ thống bán hàng
10	TRANSACTIONS	<p>Lưu thông tin các tác động làm thay đổi số dư điểm của khách trên hệ thống. Trên hệ thống có 3 loại giao dịch khác nhau: hoàn điểm do khách thanh toán hoá đơn, quy đổi điểm thành tiền thanh toán hoá đơn, hệ thống tự cộng trừ điểm cho khách theo chính sách.</p> <p>Ứng với mỗi giao dịch cộng điểm, hệ thống có thể sinh ra nhiều bản ghi cộng điểm tương ứng với nhiều chương trình khuyến mãi khác nhau, ví dụ hệ thống sinh ra 1 bản ghi hoàn điểm 3% cho mặt hàng thực phẩm sạch và 1 bản ghi hoàn điểm 2% cho các mặt hàng khác của đơn hàng. Khi đó trường transaction_type nhận giá trị "AWARD", trường total_amount nhận giá trị dương đúng bằng số điểm được cộng, trường description ghi nhận mã đơn hàng của hệ thống bán hàng.</p>

		<p>Trong quá trình giao dịch, nếu khách hàng thanh toán bằng điểm, hệ thống sinh thêm một bản ghi có transaction_type nhận giá trị 'SPENT' và trường total_amount nhận giá trị âm (thể hiện việc tiêu làm trừ điểm) , trường description ghi nhận mã đơn hàng của hệ thống bán hàng.</p> <p>Đôi khi DF corp có chương trình khuyến mãi cộng điểm cho khách, hoặc tiến trình tự động trừ điểm hết hạn, hoặc sửa sai giao dịch lỗi, hệ thống tạo một bản ghi có transaction_type = 'SYSTEM'.</p> <p>Các thông tin liên quan (nếu có) của cả 3 loại tác động trên đều được ghi nhận vào trường description dưới dạng text.</p>
11	BALANCE	<p>Lưu thông tin số dư điểm của khách trên hệ thống. Theo quy định điểm số có thời gian sử dụng là 365 ngày kể từ ngày được cộng. Một khách hàng có thể có nhiều bản ghi balance trên hệ thống, mỗi bản ghi tương ứng với một lần cộng điểm. Tổng giá trị trường balance chính là điểm số của khách.</p>

**Bảng 2.2:** Danh sách bảng dữ liệu hệ thống Khách hàng trung thành của DF Corp

Mỗi khi có cập nhật giá trị hoặc xóa bản ghi, hệ thống KHTT không chạy lệnh update hoặc delete. Với lệnh xóa, hệ thống vô hiệu hoá bản ghi bằng cách cập nhật trường status về giá trị 'Inactive'. Với lệnh cập nhật, hệ thống vô hiệu hoá bản ghi cũ bằng cách cập nhật trường status về giá trị 'Inactive', sau đó thêm một bản ghi vào bảng dữ liệu với giá trị status là 'Active'.

Chỉ các giao dịch có thẻ thành viên mới được ghi nhận vào hệ thống KHTT, các giao dịch không có thông tin không được ghi nhận.

## 2.2 Mô tả yêu cầu người dùng

Kho dữ liệu được tổng hợp từ nhiều nguồn khác nhau, phục vụ các bộ phận khác nhau, mỗi bộ phận lại có cách hiểu khác nhau về một thuật ngữ hoặc công thức tính một chỉ tiêu nào đó. Để tránh nhầm lẫn, tài liệu thiết kế cần xác định rõ danh sách chỉ tiêu và mô tả công thức tính từng chỉ tiêu vừa xác định.

Khi xây dựng phần này, tác giả thường dựng một bảng có 3 cột thông tin: mã chỉ tiêu, tên chỉ tiêu và mô tả ý nghĩa, công thức của chỉ tiêu tương ứng. Để tiện theo dõi, các chỉ tiêu có thể được sắp xếp theo data mart. Trên một số hệ thống có nhiều chỉ tiêu có tên giống nhau nhưng ý nghĩa/công thức khác nhau, cần phân biệt bằng mã. Ví dụ chỉ tiêu "Số hoá đơn" trên hệ thống bán hàng bao gồm tất cả các đơn hàng trên 3 hệ thống siêu thị / cửa hàng tiện lợi / trang web thương mại điện tử, còn trên hệ thống Khách hàng trung thành lại có nghĩa là các đơn hàng có khách lựa chọn dùng thẻ định danh.



ID	Tên chỉ tiêu	Mô tả
<b>Hệ thống bán hàng</b>		
A001	Số lượng đơn hàng	
A002	Tổng tiền bán hàng	
...	...	...

**Bảng 2.3:** Bảng mô tả chỉ tiêu có thể khai thác qua hệ thống kho dữ liệu

## 2.3 Thiết kế tổng thể

Theo Kimball, việc thiết kế một kho dữ liệu tuân theo quy trình 4 bước:

### 1) Xác định các nghiệp vụ cần được mô hình hoá:

Các hệ thống nghiệp vụ thường bao gồm rất nhiều thông tin đôi khi không cần thiết cho yêu cầu phân tích, khi đó việc lưu trữ dữ liệu không cần thiết đầy trở nên dư thừa, các tiến trình xử lý dữ liệu có thể làm ảnh hưởng đến các nghiệp vụ quan trọng hơn. Nếu không có nhu cầu sử dụng hoặc chưa thấy thấy nhu cầu trong tương lai “gần”, không nên mô hình hoá nghiệp vụ đó vào kho dữ liệu. Mỗi nghiệp vụ này thường được gọi là một Data Mart.

Mô hình dữ liệu mô tả trong mục 2.1 chỉ tập trung vào 2 mảng là nghiệp vụ bán hàng và nghiệp vụ hoàn tiền/đổi điểm của khách hàng. Trên thực tế hệ thống phần mềm bán hàng còn có thể có thêm các nghiệp vụ như kho bãi, giao vận, quản lý nhân viên... Về mặt quan điểm, tất cả nghiệp vụ đều được mô hình hoá để tạo thành kho dữ liệu tổng thể của doanh nghiệp, tuy nhiên tại giai đoạn này việc xây dựng kho dữ liệu chỉ tiến hành cho 2 nghiệp vụ nói trên. Các nghiệp vụ còn lại sẽ được bổ sung vào tài liệu thiết kế trong quá trình xây dựng sau này.

### 2) Xác định độ chi tiết của dữ liệu:

Một sự kiện, theo nhu cầu, có thể được chia làm nhiều sự kiện nhỏ hơn, mỗi sự kiện nhỏ có thể được chia làm nhiều sự kiện nhỏ hơn nữa. Khi phân tích yêu cầu người dùng, người thiết kế nên chú ý đến độ chi tiết nhỏ nhất người dùng cần hoặc muốn để xây dựng hệ thống hợp lý. Thể hiện rõ nhất của việc tăng giảm độ chi tiết dữ liệu là số lượng bậc trong cây phân cấp bảng dimension hoặc số lượng bảng fact: độ chi tiết càng cao thì cây phân cấp bảng dimension và số lượng bảng fact càng nhiều. Việc thiết kế độ chi tiết dữ liệu ở mức nhỏ nhất có thể gây khó khăn cho người phân tích trong khi nhu cầu của họ không đến mức đó. Để xác định độ chi tiết dữ liệu cần bám theo các nghiệp vụ trong data mart. Nên cố gắng thể hiện thông tin độ chi tiết dữ liệu hết mức có thể: các thông tin tưởng là hiển nhiên tại thời điểm thiết kế bỗng trở nên khó hiểu sau vài lần nâng cấp hệ thống nghiệp vụ.

Qua khảo sát nắm bắt được độ chi tiết dữ liệu của hệ thống bán hàng như sau:

- Địa chỉ khách hàng và địa chỉ cửa hàng chỉ quan tâm đến mức phường xã, không cần chi tiết đến đường phố.

- Việc phân tích địa chỉ khách hàng chỉ dừng lại ở địa chỉ liên lạc chính. Các địa chỉ liên lạc phụ (nếu có), địa chỉ thanh toán và địa chỉ nhận hàng không có giá trị phân tích trong data mart bán hàng.
- Sản phẩm có độ chi tiết đến mức SKU, chủng loại hàng hoá từ mức thấp nhất đến mức cao nhất
- Hoá đơn bán hàng có độ chi tiết đến dòng đơn hàng và các phương pháp / lần thanh toán. Thông tin đơn hàng chỉ thể hiện đến mức cửa hàng, thông tin thanh toán thể hiện đến mức máy POS.

Tương tự với độ chi tiết dữ liệu của data mart Khách hàng trung thành:

- Địa chỉ khách hàng và địa chỉ cửa hàng chỉ quan tâm đến mức phường xã, không cần chi tiết đến đường phố.
- Việc phân tích địa chỉ khách hàng chỉ dừng lại ở địa chỉ liên lạc chính. Các địa chỉ liên lạc phụ (nếu có), địa chỉ thanh toán và địa chỉ nhận hàng không có giá trị phân tích trong data mart bán hàng.
- Giao dịch của khách chỉ quan tâm đến mức cửa hàng, thông tin máy POS không có giá trị phân tích. Tương tự việc phân tích cũng không quan tâm đến các giá trị % hoàn tiền trong một lần thanh toán mà chỉ quan tâm đến giao dịch của khách hàng: khách được hoàn bao nhiêu điểm, đã tiêu bao nhiêu điểm trong một phiên giao dịch.
- Việc phân tích số dư tài khoản chỉ quan tâm đến tổng số dư, các số liệu về thời gian phát sinh điểm và thời gian điểm hết hạn không được sử dụng trong data mart này.

### 3) Xác định bảng fact:

Mỗi bảng fact đại diện cho 1 tiến trình nghiệp vụ hoặc một phép đo nào đó. Trong trường hợp 1 tiến trình nghiệp vụ gồm nhiều tiến trình con, có thể tách bảng fact ra làm nhiều bảng fact con. Khi xác định bảng fact cần đặc biệt chú ý trường hợp một bảng dữ liệu nghiệp vụ lưu nhiều tiến trình khác nhau, khi đó nên cân nhắc tạo bảng fact riêng cho mỗi tiến trình. Đôi khi một sự kiện được lưu trữ trên nhiều bảng dữ liệu hoặc hệ thống khác nhau, khi đó cũng nên cân nhắc sử dụng chung bảng fact cho tất cả các thông tin đó.

Đôi khi một sự kiện có thể được chia nhỏ lưu trữ trên nhiều bảng fact khác nhau, mỗi bảng fact đại diện cho một sự kiện con cấu thành sự kiện ban đầu. Việc chia nhỏ bảng fact căn cứ vào 2 câu hỏi: (1) các sự kiện con có thời gian xảy ra giống nhau hay không; và (2) các sự kiện con có cùng độ chi tiết dữ liệu hay không. Các sự kiện xảy ra đồng thời và có cùng độ chi tiết dữ liệu nên được lưu trữ trong một bảng fact. Các sự kiện có thời gian xảy ra khác nhau hoặc có độ chi tiết dữ liệu khác nhau được lưu trữ trong các bảng fact khác nhau. Một giao dịch thương mại điện tử có thể được lưu trữ trên các bảng fact đại diện cho sự kiện khách đặt hàng, sự kiện khách thanh toán, sự kiện xuất hàng ra khỏi kho và sự kiện giao hàng đến khách, vì mỗi sự kiện xảy ra ở một thời điểm khác nhau. Một giao dịch ở siêu thị có thể được lưu trữ trên 2 bảng fact đại diện cho đơn hàng và thanh toán, vì mỗi sự kiện có độ chi tiết khác nhau.

Qua phân tích mô hình cơ sở dữ liệu và khảo sát nhu cầu người dùng, tìm được các bảng fact của 2 data mart Quản lý bán hàng và Khách hàng trung thành như sau:

Data Mart	Khu vực	STT	Bảng fact	Mô tả
QLBH	Base Area	1	Bảng fact đơn hàng	Lưu trữ thông tin chi tiết các đơn hàng của khách ở siêu thị / cửa hàng tiện lợi / trang web thương mại điện tử
		2	Bảng fact dòng đơn hàng	Lưu trữ thông tin các dòng đơn hàng khách mua ở siêu thị / cửa hàng tiện lợi / trang web thương mại điện tử
		3	Bảng fact trả hàng	Lưu các mặt hàng do khách đổi trả
		4	Bảng fact thanh toán	Lưu trữ thông tin các lần thanh toán của khách ở siêu thị / cửa hàng tiện lợi / trang web thương mại điện tử
	Mart Area	5	360° khách hàng	Bảng fact tổng hợp các chỉ tiêu, số liệu liên quan đến khách hàng trong kỳ báo cáo và lũy kế trên hệ thống quản lý bán hàng
		6	360° cửa hàng	Bảng fact tổng hợp các chỉ tiêu, số liệu liên quan đến cửa hàng trong kỳ báo cáo và lũy kế trên hệ thống quản lý bán hàng
KHTT	Base Area	7	Bảng fact giao dịch điểm	Lưu trữ thông tin các giao dịch của khách được ghi nhận trên hệ thống Khách hàng trung thành
		8	Bảng fact hoàn điểm do hệ thống	Lưu trữ thông tin các giao dịch thay đổi điểm của khách do hệ thống thực hiện
		9	Bảng fact số dư điểm	Lưu trữ thông tin số dư điểm của khách
		10	Bảng fact đăng ký thành viên	Lưu trữ thông tin hồ sơ đăng ký của khách
	Mart Area	11	360° khách hàng	Bảng fact tổng hợp các chỉ tiêu, số liệu liên quan đến khách hàng trong kỳ báo cáo và lũy kế trên hệ thống Khách hàng trung thành
		12	360° cửa hàng	Bảng fact tổng hợp các chỉ tiêu, số liệu liên quan đến cửa hàng trong kỳ báo cáo và lũy kế trên hệ thống Khách hàng trung thành
Unified	Mart Area	13	360° khách hàng	Lưu trữ thông tin các giao dịch của khách được ghi nhận trên hệ thống DF Corp
		14	360° cửa hàng	Lưu trữ thông tin các giao dịch của khách được ghi nhận trên hệ thống DF Corp

**Bảng 2.4:** Danh sách bảng fact kho dữ liệu DF Corp

Theo danh sách trên, ta thấy có 3 bảng fact đều có tên “360° khách hàng” (bảng số 5, số 11 và số 13), 3 bảng fact có tên “360° cửa hàng” (bảng số 6, số 12 và số 14), mỗi nhóm bảng đó đều tổng hợp các chỉ tiêu, số liệu liên quan đến cùng một đối tượng. Tuy nhiên ta thấy bảng số 5, số 6 thuộc data mart Quản lý bán hàng, bảng số 11, 12 thuộc data mart Khách hàng trung thành. Mỗi data mart có luồng xử lý dữ liệu độc lập nhau, chạy vào những khoảng thời gian có thể không giống nhau, được phân tích bởi các phòng ban có quyền hạn truy cập dữ liệu khác nhau nên cần chia ra thành nhiều bảng như vậy. Tùy nhu cầu sử dụng có thể kết hợp bảng tổng hợp có chung ý nghĩa giữa các data mart về một bảng chung. Theo thiết kế trên, bảng 5 kết hợp với bảng 11 tạo ra bảng 13, bảng 6 kết hợp với bảng 12 ra bảng 14.

Ngoài ra ta có thể thấy có sự liên quan giữa 2 bảng fact số 1 và số 7: cả 2 bảng đều lưu trữ thông tin đơn hàng của khách trên các cửa hàng của DF Corp. Theo logic tách – nhập bảng fact đã trình bày, 2 bảng fact đó cùng lưu thông tin của 1 sự kiện nên chỉ được lưu trữ trong 1 bảng dữ liệu duy nhất. Không may trong trường hợp của chúng ta, các bảng dữ liệu giao dịch của 2 hệ thống này được thiết kế không tốt nên không liên kết được với nhau (vì mã đơn hàng được lưu trữ trong trường description bảng TRANSACTION dưới dạng text). Đây là điều rất đáng tiếc làm ảnh hưởng đến chất lượng của kho dữ liệu, cần được chú ý theo dõi để xây dựng tiến trình liên kết ngay khi có thể. Nếu liên kết được, có thể xây dựng thêm một bảng “Fact giao dịch” dùng chung giữa 2 data mart Quản lý bán hàng và Khách hàng trung thành.

Trong một số trường hợp, dữ liệu bảng fact được lấy trực tiếp từ một bảng danh mục nào đó. Ở kho dữ liệu DF Corp, thông tin ngày đăng ký có thể lấy từ bảng danh sách khách hàng, tuy nhiên vẫn cần 1 bảng fact lưu lại lượt đăng ký (bảng 10). Nguyên nhân vì dữ liệu trên các bảng danh mục được cập nhật/xoá thường xuyên dẫn đến các kết quả khác nhau khi thực hiện cùng lệnh query tại các thời điểm khác nhau.

Đôi khi dữ liệu 2 bảng fact khác nhau được lưu trong một bảng ở hệ thống nghiệp vụ, giống trường hợp bảng fact số 2 và số 3 trên hệ thống DF Corp. Khi đó hệ thống nghiệp vụ bổ sung trường *status* vào bảng Dòng đơn hàng. Theo nguyên tắc tách bảng fact, 2 sự kiện mua hàng – trả hàng phải được chia tách thành 2 bảng fact khác nhau cả về mặt logic lẫn vật lý. Theo kinh nghiệm của tác giả, việc nhập bảng vật lý không thân thiện với người phân tích, khiến người phân tích phải ghi nhớ thêm logic, tăng khả năng mắc sai lầm, nên tách thành 2 bảng số 2 và số 3 như thiết kế. Tuy nhiên tùy theo nhu cầu người dùng có thể xây dựng một bảng dữ liệu thứ ba gộp các dòng dữ liệu liên quan đến đơn hàng để tiện query. Logic này áp dụng tương tự cho bảng số 7, số 8 (giao dịch và hoàn điểm trên hệ thống Khách hàng trung thành).

#### **4) Xác định bảng dimension:**

Sau khi có danh sách bảng fact, ta phải xác định các dimension tương ứng với từng bảng fact nói trên. Không như quy hoạch bảng fact, việc thiết kế danh sách dimension này tương đối khó khăn vì các dimension được dùng chung giữa các data mart với nhau, ở thời điểm thiết kế không thể xác định chính xác độ chi tiết dữ liệu và danh sách dimension cho các data mart được xây dựng sau này. Để giải quyết vấn đề này cần một phương pháp thiết kế cho phép mở rộng kho dữ liệu dần dần.

Tác giả hay dùng một phương án thiết kế gọi là Data Warehouse Bus Matrix, do Ralph Kimball đề xuất. Theo đó, người thiết kế xây dựng một ma trận 2 chiều với mỗi hàng là 1 tiến trình nghiệp vụ (1 bảng fact), mỗi cột là 1 dimension. Mỗi ô trong ma trận thể hiện bảng fact có tương tác với dimension hay không. Ở mỗi giai đoạn xây dựng data mart sau này, người thiết kế bổ sung bảng fact và dimension tương ứng vào ma trận.

Ma trận kiến trúc hệ thống kho dữ liệu được thể hiện trên bảng 2.5.

		BẢNG DIMENSION							
		Dimension Thẻ thành viên	Dimension Sản phẩm	Dimension Phương pháp thanh toán	Dimension Máy POS	Dimension Cửa hàng	Dimension Khách hàng	Dimension Khung giờ	Dimension Ngày tháng
BẢNG FACT									
QLBH	Bảng fact đơn hàng	X	X		X				
	Bảng fact dòng đơn hàng	X	X	X	X			X	
	Bảng fact trả hàng	X	X	X	X			X	
	Bảng fact thanh toán	X	X	X	X	X	X		
	360° khách hàng	X			X				
	360° cửa hàng	X							
KHTT	Bảng fact giao dịch điểm	X	X		X				X
	Bảng fact hoàn điểm do hệ thống	X	X		X				X
	Bảng fact số dư điểm	X			X				
	Bảng fact đăng ký thành viên	X	X	X	X				X
	360° khách hàng	X			X				
	360° cửa hàng	X							
Unified	360° khách hàng	X			X				
	360° cửa hàng	X							

**Bảng 2.5:** Ma trận kiến trúc kho dữ liệu DF Corp

Thực tế xây dựng kho dữ liệu, số lượng dimension ở ma trận nói trên thường không đủ, sẽ phát sinh thêm một số trong giai đoạn thiết kế bảng fact. Khi đó tùy tính chất bảng dimension có thể bổ sung các bảng bị thiếu.

## Phần 3: Xây dựng bảng fact

Như đã trình bày trong mục 1.2, bảng fact dùng để lưu các tiêu chí, chỉ tiêu về hoạt động sản xuất kinh doanh của tổ chức, doanh nghiệp. Một **chỉ tiêu (measurement, indicator)** được định nghĩa là một lượng quan sát được theo một đơn vị đo lường thống nhất. Mỗi bảng fact có 3 loại cột dữ liệu khác nhau: (1) nhóm cột lưu các chỉ tiêu, số liệu; (2) nhóm cột lưu các khoá phụ liên kết với bảng dimension; và (3) nhóm cột lưu các thông tin bổ sung dùng hỗ trợ hệ thống nghiệp vụ. Bản thân nhóm cột lưu chỉ tiêu lại được chia ra làm nhiều loại khác nhau nữa:

- Các cột số liệu **bán cộng dồn (semiadditive measure)** là các cột chỉ có thể được cộng theo một số dimension nhất định, ví dụ như các cột về tỉ lệ %
- Các cột số liệu **không cộng dồn được (nonadditive measure)** là các cột chỉ có ý nghĩa khi dùng các phép thống kê (tính trung bình chiều cao, cân nặng...)
- Các cột số liệu **thoái hoá (degenerate measure)** là các thông tin dimension tồn tại trong bảng fact nhưng không được ghi nhận ở bảng dimension riêng.

Mô hình dữ liệu đa chiều của kho dữ liệu được xây dựng xung quanh các tiêu chí này. Bảng số liệu chứa tiêu chí, còn bảng dimension chứa ngữ cảnh của các chỉ tiêu đó. Mỗi quan hệ tưởng chừng đơn giản này cung cấp cho người dùng cuối góc nhìn rất trực quan, đầy đủ và dễ sử dụng về số liệu trong kho.

Phần 2 của tài liệu đã trình bày phương pháp thiết kế tổng quan một kho dữ liệu. Phần 3 này sẽ tập trung thiết kế chi tiết các bảng fact của kho.

### 3.1 Nguyên tắc chung

Như đã trình bày ở mục 1.2.1, bảng fact là bảng dữ liệu lưu các thông tin đo đếm được liên quan đến một sự kiện hoặc một phép đo đặc toán học nào đó. Có 3 loại bảng fact: bảng fact giao dịch (transactional fact), bảng fact snapshot (snapshot fact) và bảng fact tổng hợp (aggregated fact).

Việc thiết kế một bảng fact cần tối thiểu các thông tin sau:

- Mô tả ý nghĩa bảng fact
- Mô tả nguồn dữ liệu của bảng fact: Có bảng fact lấy dữ liệu từ nhiều nguồn khác nhau. Có trường hợp việc extract dữ liệu fact phải lọc theo những điều kiện nhất định.
- Mô tả nguồn dữ liệu của bảng fact:
  - Lấy dữ liệu từ những nguồn nào chi tiết đến mức bảng dữ liệu và điều kiện query trên mỗi bảng
  - Cần cập nhật file thiết kế trong trường hợp có thay đổi từ hệ thống nghiệp vụ: thêm nguồn dữ liệu, sửa logic bảng dữ liệu...
- Có sơ đồ thể hiện quan hệ của bảng fact với các dimension có liên quan.
- Liệt kê danh sách và mô tả chi tiết các cột dữ liệu của bảng fact. Chú ý mô tả kỹ các cột dữ liệu có công thức tính toán phức tạp, các cột lấy dữ liệu từ nhiều nguồn khác nhau
- Danh sách index và partition (nếu có) sử dụng trong bảng fact: mô tả công thức index, partition
- Các mô tả, ghi chú khác về bảng fact (nếu có)

Ví dụ bảng fact thanh toán đơn hàng trên hệ thống DF Corp được mô tả trên bảng 3.1

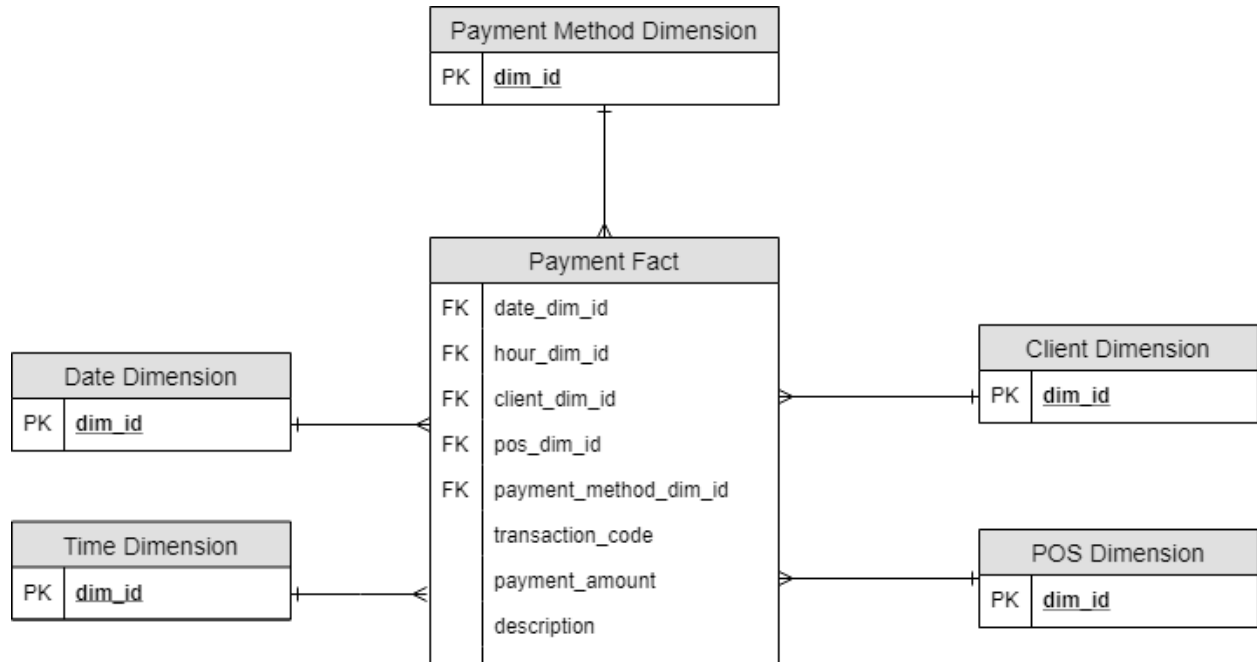
MÔ TẢ BẢNG FACT			
Subject Area	Quản lý bán hàng		
Tên bảng fact	Bảng fact thanh toán đơn hàng		
Loại bảng fact	Bảng fact giao dịch		
Tên bảng fact trong kho dữ liệu	F_TRANSACTION_PAYMENTS		
Tần suất cập nhật	Hàng ngày		
Phương pháp cập nhật	Insert		
Nguồn dữ liệu	Bảng tblPayments hệ thống bán hàng, điều kiện status = 'FINISH'		
Mô tả	Bảng fact lưu thông tin thanh toán của khách hàng		
STT	Tên cột	Kiểu dữ liệu	Mô tả
1	DATE_DIM_ID	NUMBER	Ngày khách hàng thanh toán. Khoá phụ liên kết đến dimension D_DATE
2	HOURL_DIM_ID	STRING	Giờ khách hàng thanh toán. Khoá phụ liên kết đến dimension D_HOUR
3	POS_DIM_ID	STRING	Máy POS thực hiện thanh toán. Khoá phụ liên kết đến dimension D_POS
4	CLIENT_DIM_ID	STRING	Khách hàng thực hiện thanh toán. Khoá phụ liên kết đến dimension D_CLIENT
5	PAYMENT_METHOD_DIM_ID	STRING	Phương pháp thanh toán. Khoá phụ liên kết đến dimension D_PAYMENT_METHOD
6	TRANSACTION_CODE	STRING	Mã hoá đơn trên hệ thống nghiệp vụ quản lý bán hàng
7	PAYMENT_AMOUNT	NUMBER	Số tiền khách thanh toán
8	DESCRIPTION	STRING	Mô tả khác từ hệ thống nghiệp vụ hoặc do thu ngân nhập
Index và Partition			
Đánh partition trường DATE_DIM_ID <ul style="list-style-type: none"> <li>- Kiểu partition: Range</li> <li>- Giá trị ban đầu: từ 20100101 đến 20300101 (từ 01/01/2010 đến 01/01/2030)</li> </ul> Các index: <ul style="list-style-type: none"> <li>- CLIENT_DIM_ID</li> </ul>			



- PAYMENT\_METHOD\_DIM\_ID
- PAYMENT\_STATUS\_DIM\_ID
- TRANSACTION\_CODE

**Bảng 3.1:** Mô tả bảng fact đơn hàng hệ thống DF Corp

Tương tự, sơ đồ quan hệ bảng fact thanh toán được thể hiện trên hình 3.1



**Hình 3.1:** Sơ đồ quan hệ bảng fact Thanh toán

Để sơ đồ có tính thẩm mỹ, bảng fact không cần thiết phải liệt kê tất cả các cột dữ liệu mà chỉ vẽ các cột được cho là quan trọng nhất (tại thời điểm thiết kế). Lúc này bảng dimension chưa được thiết kế, không có thông tin về cột dữ liệu nên cũng không cần thể hiện trên sơ đồ.

Trong quá trình xây dựng bảng fact cần chú ý trường hợp cột dữ liệu liên kết đến bảng dimension bị rỗng (nhận giá trị null). Do tính chất kiểu dữ liệu null hơi khác biệt (không thể so sánh 2 giá trị null với nhau) nên cần gán giá trị cho cột null để dễ thao tác. Thông thường tác giả hay thêm 1 bản ghi dimension với giá trị dim\_id = -1, name = 'Không có thông tin' vào bảng dimension và liên kết bản ghi fact null đến bản ghi dimension -1 vừa tạo.

Trong vòng đời kho dữ liệu, sẽ có lúc hệ thống nghiệp vụ đầu vào của kho được thay đổi/cập nhật logic. Khi chuyện đó xảy ra nhân viên phát triển kho dữ liệu cần sửa đổi tài liệu thiết kế, bổ sung logic mới vào các bảng fact có liên quan. Theo kinh nghiệm của tác giả, các nội dung có liên quan đến logic cũ/hệ thống cũ vẫn nên được ghi nhận trên tài liệu nhằm mục đích tham khảo nếu cần.

## 3.2 Bảng fact giao dịch

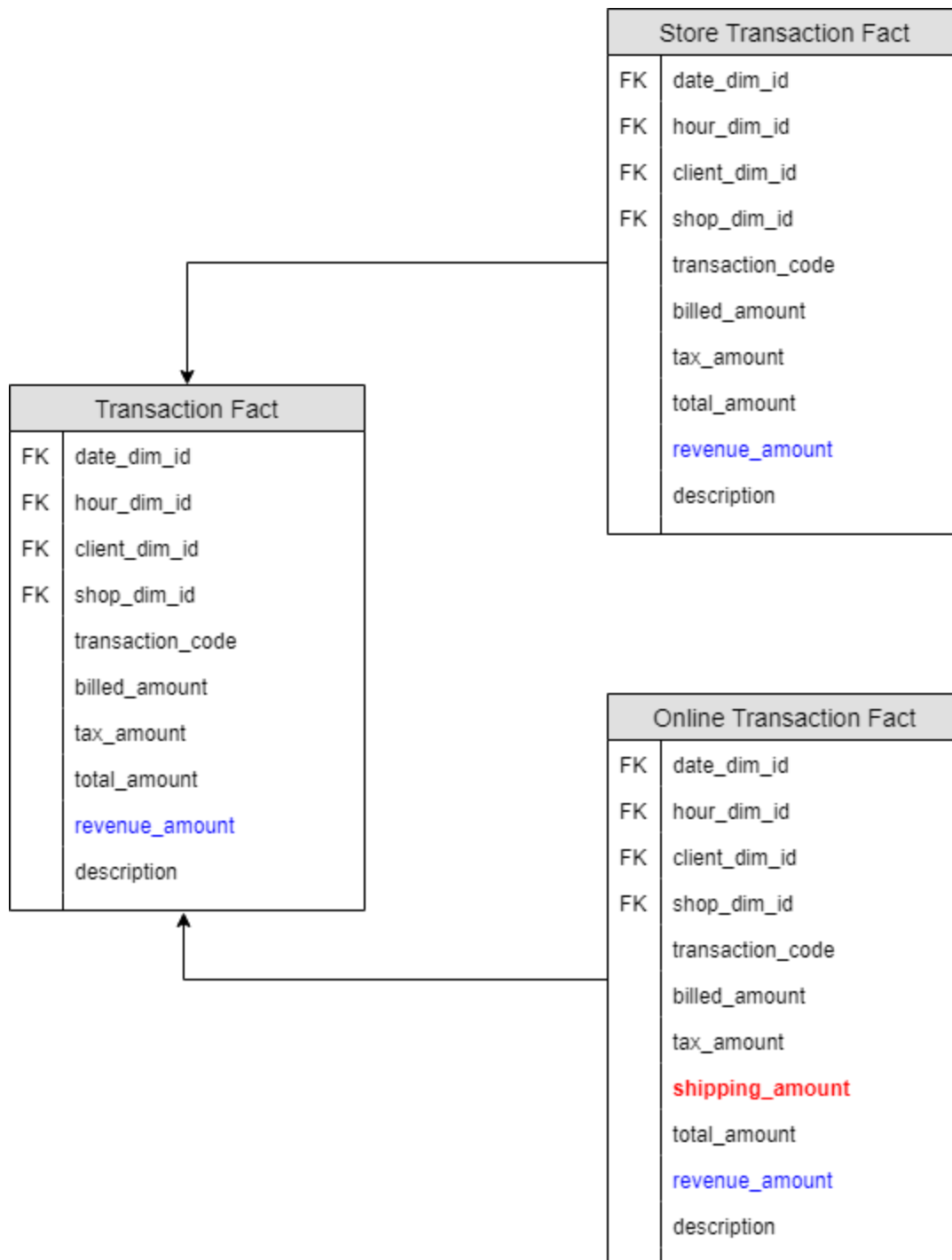
Bảng fact giao dịch mô tả bằng dữ liệu một sự kiện nào đó. Sự kiện này có thể rất rõ ràng, có số liệu đi kèm như sự kiện khách mua hàng, sự kiện khách trả hàng. Hoặc sự kiện có thể không rõ ràng, không có số liệu đi kèm như việc khách đăng ký thẻ thành viên.

Khi thiết kế bảng fact giao dịch này, một trong những việc quan trọng nhất là xác định danh sách các sự kiện được mô hình hoá, cụ thể là thiết kế một sự kiện có thể / có nên chia thành nhiều bảng fact hay không. Nếu nhập quá nhiều sự kiện vào một bảng fact, người phân tích sẽ phải ghi nhớ logic khi query dữ liệu, làm tăng rủi ro phân tích sai. Việc chia tách bảng fact quá nhiều lại làm kho dữ liệu phình to về mặt số lượng, khó nắm bắt được bức tranh tổng thể.

Một sự kiện lại có thể được lưu trữ phân tán trên nhiều hệ thống nghiệp vụ khác nhau, mỗi hệ thống lại có thể có cấu trúc dữ liệu khác nhau. Khi thiết kế chỉ có thể nắm rõ cấu trúc và logic dữ liệu của các hệ thống có sẵn, rất khó dự đoán cho tương lai. Thông thường một sự kiện được lưu trữ trên nhiều hệ thống khác nhau khi:

- Sự kiện xảy ra ở một hệ thống A nhưng một phần công việc thực hiện ở hệ thống B. Khi đó dữ liệu ở hệ thống B bổ sung thông tin cho sự kiện xảy ra ở A. Ở hệ thống DF Corp, giao dịch hoàn/tiêu điểm bổ sung thông tin cho giao dịch mua hàng của khách
- Hệ thống thông tin thiết kế theo mô hình phân tán: mỗi chi nhánh có hệ thống phần mềm quản lý riêng, hay gặp ở các công ty đa quốc gia
- Nhiều sự kiện có tính chất hơi khác nhau, được quản lý bởi các hệ thống phần mềm khác nhau. Trong ví dụ của chúng ta, hệ thống bán hàng DF Corp dùng chung cho cả siêu thị/cửa hàng tiện lợi/trang thương mại điện tử, tuy nhiên các công ty hay có hệ thống phần mềm riêng cho mỗi loại cửa hàng.

Ở trường hợp thứ hai và thứ ba kể trên, ngoài việc tập trung dữ liệu vào một bảng fact để tiện truy vấn toàn hệ thống, một nhu cầu thường gặp là chia nhỏ bảng fact ra thành nhiều mảnh khác nhau để người phân tích có thể tập trung vào một mảnh, không cần ghi nhớ điều kiện lấy dữ liệu, hạn chế nhầm lẫn. Rõ ràng thuê bao điện thoại ở Châu Phi có hành vi tiêu dùng rất khác thuê bao điện thoại Đông Nam Á; hoặc khách của siêu thị có nhu cầu tương đối khác biệt so với khách trang thương mại điện tử. Trong hai trường hợp kể trên, mỗi loại sự kiện có thể có các thuộc tính đặc thù cần phân tích. Ví dụ việc chia nhỏ bảng fact giao dịch trên hệ thống DF Corp được thể hiện trên hình 3.2.



**Hình 3.2:** Sơ đồ quan hệ các bảng fact Giao dịch

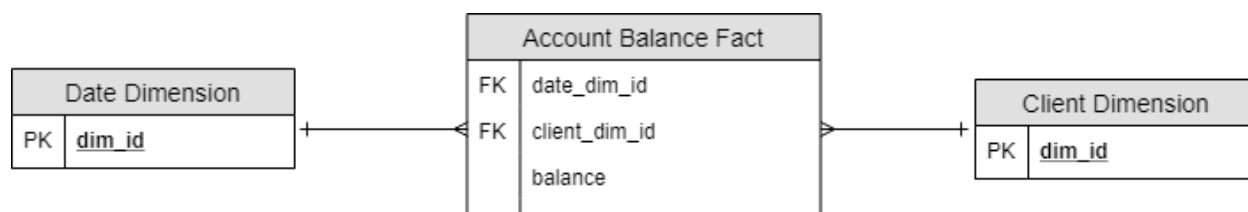
Ở đây giao dịch mua hàng online có thêm trường `shipping_amount` là thông tin fact cần phân tích. Hình vẽ trên chỉ là ví dụ minh họa đơn giản cho lý thuyết chia tách bảng fact, trên thực tế số lượng trường dữ liệu đặc thù của mỗi loại sự kiện thường lớn hơn rất nhiều.

Việc xác định danh sách chỉ tiêu trong bảng fact không nhất thiết phải bám theo các chỉ tiêu có sẵn trên nguồn dữ liệu mà có thể lấy ý kiến người phân tích, bổ sung thêm cột dữ liệu theo nhu cầu. Theo đó, có thể bổ sung cột  $\text{revenue\_amount} = \text{billed\_amount} - \text{tax\_amount}$  trên bảng fact Giao dịch ở hình 3.2. Công thức cột này cũng có thể thay đổi tùy theo tính chất bảng fact, ví dụ như với fact Giao dịch online, công thức trên trở thành  $\text{revenue\_amount} = \text{billed\_amount} - \text{shipping\_amount} - \text{tax\_amount}$ .

### 3.3 Bảng fact snapshot

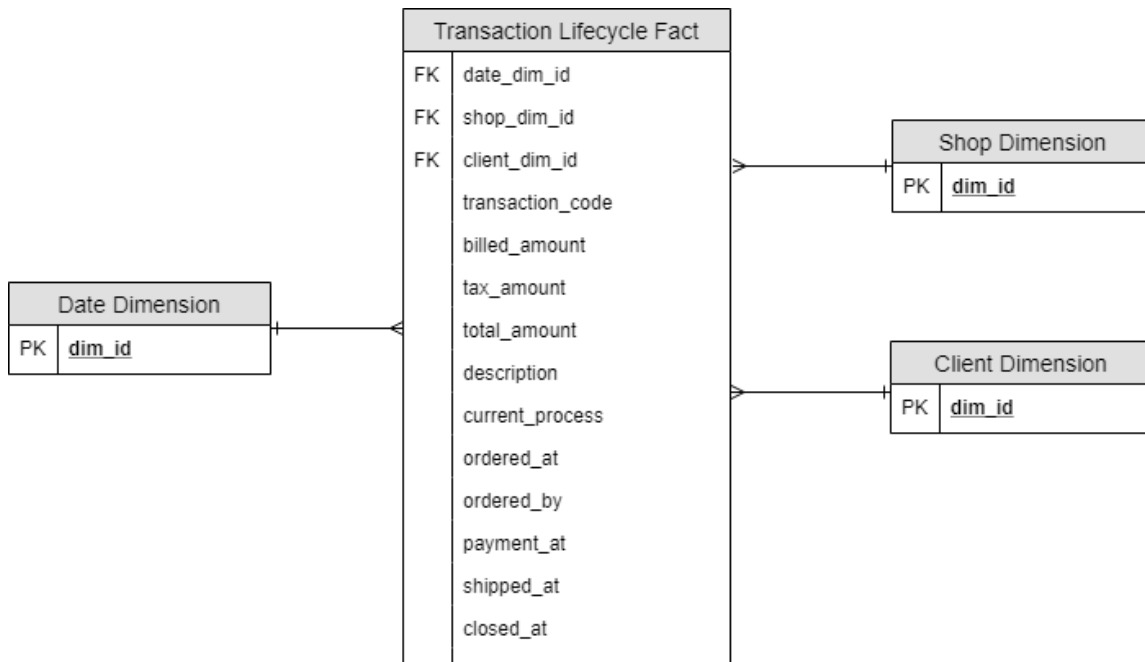
Bảng fact snapshot định kỳ lưu trữ thông tin của một phép đo hoặc thuộc tính của đối tượng nào đó, dùng để phân tích sự biến thiên dữ liệu theo thời gian. Có 2 loại bảng snapshot:

**1) Bảng fact snapshot (bình thường):** lưu thông tin phép đo hoặc thuộc tính của đối tượng quan tâm tại thời điểm đo, không quan tâm đến vòng đời của đối tượng. Hệ thống DF Corp sử dụng một bảng fact loại này để lưu thông tin số dư điểm trong tài khoản của khách, thể hiện trên hình 3.3.



Hình 3.3: Sơ đồ quan hệ bảng fact Số dư

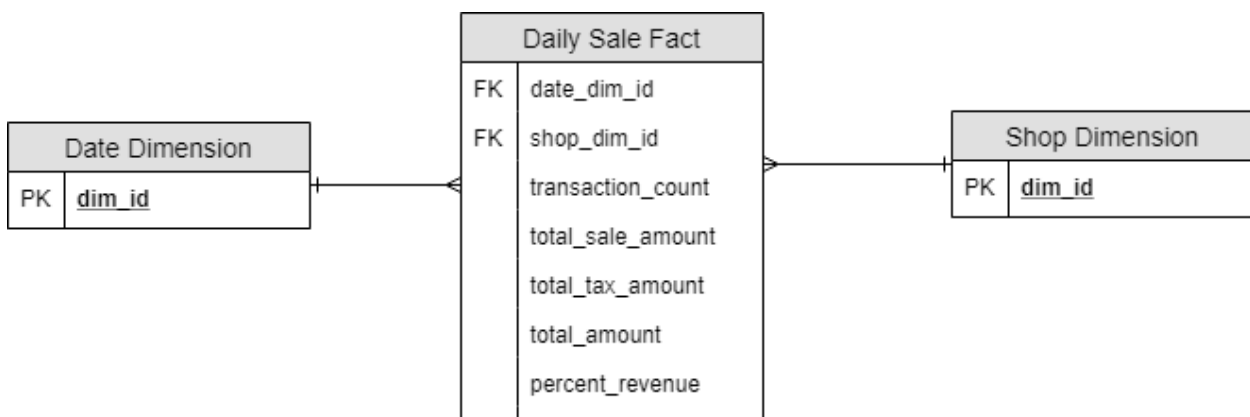
**2) Bảng accumulating snapshot:** lưu thông tin vòng đời đến thời điểm đo của một quy trình nghiệp vụ nào đó. Sử dụng bảng fact accumulating, người phân tích có thể biết chính xác trạng thái của một sự kiện nào đó hay phân tích các sự kiện đang diễn ra trên hệ thống theo trạng thái của chúng. Để có ý nghĩa, quy trình được mô hình hoá phải có vòng đời tương đối dài (so với kỳ đo dữ liệu). Quy trình được mô hình hoá cũng phải có số bước xử lý cố định, tuần tự (hoặc có thể nội suy ra một quy trình cố định, tuần tự). Hệ thống DF Corp không có loại bảng này nhưng có thể thiết kế cho quy trình bán hàng online, khi quy trình xử lý đơn hàng có số bước cố định (đặt hàng – thanh toán – ship hàng) và mất đến vài ngày để hoàn thành.



Hình 3.4: Sơ đồ quan hệ bảng fact Vòng đời đơn hàng

### 3.4 Bảng fact tổng hợp

Bảng **fact tổng hợp (aggregated fact)** tính toán sẵn các chỉ tiêu có liên quan đến một hoặc một nhóm dimension nào đó, giúp thuận lợi hơn cho quá trình phân tích và hiển thị báo cáo. Không như bảng fact giao dịch và fact snapshot, fact tổng hợp được xây dựng theo nhu cầu của người phân tích. Về mặt thể hiện, bảng fact tổng hợp khá giống fact snapshot, cùng định kỳ ghi nhận giá trị đo được về một sự vật nào đó. Điểm khác biệt là bảng fact tổng hợp tập trung số liệu từ nhiều phép đo khác nhau, các phép đo không chỉ thể hiện số liệu của thời điểm đo mà là phép tổng hợp từ chu kỳ trước đến chu kỳ này.



Hình 3.5: Sơ đồ quan hệ bảng fact tổng hợp Đơn hàng theo ngày

Bảng 3.5 mô tả một ví dụ bảng fact tổng hợp trên hệ thống DF Corp để theo dõi số lượng đơn hàng, thành tiền và lãi của mỗi cửa hàng trên hệ thống theo từng ngày. Trên hình ảnh ví dụ chỉ

thể hiện số liệu phát sinh trong kỳ báo cáo, tuy nhiên có thể thiết kế thêm các chỉ tiêu lũy kế như số lũy kế tháng, lũy kế năm, vòng đời...

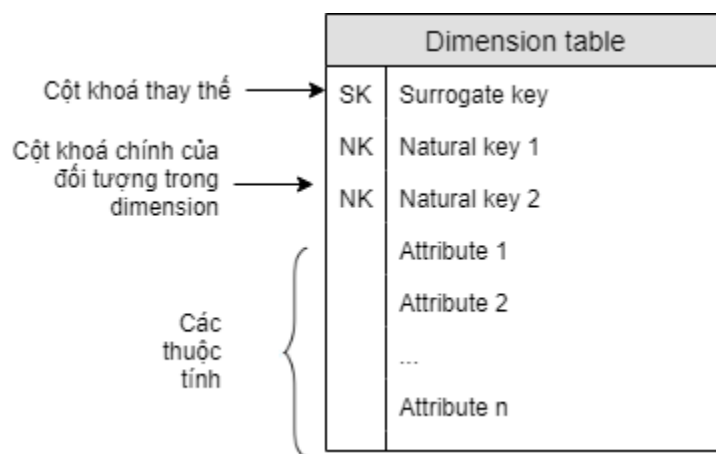
Trên hệ thống DF Corp, bảng fact 360° cửa hàng và 360° khách hàng chính là 2 bảng fact tổng hợp. Để đơn giản phù hợp với nội dung tài liệu này, tác giả chỉ giới thiệu bảng ở mức khách hàng/cửa hàng cho kỳ hạn ngày. Khi thiết kế hệ thống để sử dụng thực tế, có thể bổ sung các mức cao hơn theo nhu cầu như mức cửa hàng/khách hàng cho chu kỳ tháng, mức chi nhánh chu kỳ ngày, mức chi nhánh chu kỳ tháng...

## Phần 4: Xây dựng bảng dimension

**Bảng dimension** cung cấp các thông tin, ngữ cảnh cho bảng số liệu. Dù có quy mô nhỏ hơn bảng số liệu rất nhiều lần, các bảng cốt lõi chính là da thịt, quần áo của kho dữ liệu vì chỉ có thể hiểu được ý nghĩa của số liệu trong kho thông qua các bảng dimension. Phần 4 của tài liệu sẽ trình bày chi tiết hơn về bảng dimension.

### 4.1 Cấu trúc bảng dimension

Các cột trong bảng dimension được chia làm 3 nhóm khác nhau: (1) cột khoá thay thế (surrogate key); (2) cột khoá chính của đối tượng dimension (natural key); và (3) các thuộc tính mô tả (descriptive attributes).



**Hình 4.1:** Cấu trúc một bảng dimension

Theo chuẩn 3 thiết kế cơ sở dữ liệu quan hệ, mỗi bảng danh mục trong mô hình đều phải có khoá chính là một hoặc một nhóm cột dữ liệu nào đó. Bảng dimension dùng duy nhất 1 cột dữ liệu làm khoá chính gọi là cột **khóá thay thế (surrogate key)**. Cột khoá thay thế này do tiến trình xử lý dữ liệu quản lý, không phụ thuộc hệ thống nghiệp vụ. Một số phương án thiết kế bảng dimension (ví dụ SCD loại 2 sẽ trình bày ở phần sau) yêu cầu phải có khoá thay thế này. Cho dùng không cần thiết, bảng dimension cũng nên sử dụng khoá thay thế để đồng nhất cấu trúc giữa các bảng dimension và để tránh trường hợp phải sử dụng quá nhiều cột khoá chính. Để đồng nhất và tránh trùng lặp, tác giả thường đặt tên cột này theo định dạng [Tên dimension]\_DIM\_ID

Mỗi hệ quản trị cơ sở dữ liệu đều cung cấp nhiều lựa chọn khác nhau cho việc sinh dữ liệu khoá chính. Tối thiểu nhất, các hệ thống đó cung cấp kiểu số tự tăng giá trị (AUTO\_INCREMENT cho MySQL, IDENTITY cho MSSQL và sequence cho Oracle...). Theo quan điểm của tác giả, không nên sử dụng kiểu số tự tăng giá trị làm khoá chính cho dimension mà nên sử dụng các thuật toán sinh chuỗi ngẫu nhiên phân tán khác như thuật toán sinh UUID<sup>3</sup> (Universally Unique Identifier). Nguyên nhân vì việc sử dụng kiểu số tự tăng có thể gây xung đột hoặc chờ tài nguyên trên hệ

<sup>3</sup> [https://en.wikipedia.org/wiki/Universally\\_unique\\_identifier](https://en.wikipedia.org/wiki/Universally_unique_identifier)

thống kho dữ liệu thời gian thực (tiến trình này chờ tiến trình kia lấy ID xong mới được chạy tiếp). Tương tự kiểu dữ liệu số tự tăng, kiểu UUID đều được các hệ quản trị cơ sở dữ liệu viết thành API để sử dụng.

Loại cột dữ liệu thứ hai trong bảng dimension là các cột **khoá tự nhiên (natural keys)** của đối tượng dimension, có thể là một hoặc một nhóm cột nào đó. Các cột này chính là khoá chính của đối tượng được mô hình hoá trên bảng dimension.

Loại cột dữ liệu cuối cùng lưu thông tin về các **thuộc tính mô tả (descriptive attribute)**. Thông tin về ngữ cảnh của bảng dimension được lưu chính trong các cột dữ liệu này. Các thuộc tính mô tả có thể ở nhiều kiểu dữ liệu khác nhau như kiểu chữ, kiểu ngày tháng thậm chí kiểu số.

## 4.2 Phân cấp trong bảng dimension

Để đúc rút ra tri thức từ dữ liệu, người dùng cần phân tích số liệu bảng fact ở nhiều mức độ chi tiết khác nhau. Với bảng fact Giao dịch hệ thống quản lý bán hàng DF Corp, người phân tích thường theo dõi số liệu ở mức Chi nhánh hoặc Tỉnh thành, sau đó soi dần xuống mức Cửa hàng tùy theo nhu cầu. Tính chất **phân cấp (hierarchy)** của bảng dimension cho phép thực hiện công việc như vậy. Trên thực tế hệ thống kho dữ liệu có một loạt thuật ngữ dùng để gọi việc phân tích các mức dữ liệu khác nhau trên bảng fact:

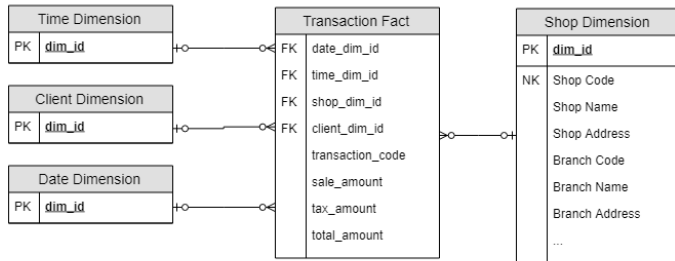
- Khả năng **nhìn lên (roll up)** hoặc **nhìn xuống (drill down)** cho phép hiển thị số liệu ở mức tổng hợp hơn (nhìn lên) hay chi tiết hơn (nhìn xuống)
- Khả năng **đảo chiều (pivot, rotate)** biến hàng thành cột, cột thành hàng
- Khả năng **cắt gọt (slice and dice)**: chỉ tập trung vào một số hàng hoặc cột nhất định trên bảng fact, ẩn các hàng hoặc cột còn lại.

Một bảng fact có thể phẳng hoàn toàn, không có phân cấp. Bảng fact cũng có thể có nhiều hơn một cây phân cấp, mỗi cây lại có số mức dữ liệu khác nhau. Bảng dimension Cửa hàng trên hệ thống DF Corp là một bảng có nhiều cây phân cấp như vậy. Trong bảng đó, ta thấy được dimension có thể phân cấp theo đơn vị quản lý (2 cấp: Cửa hàng => Chi nhánh) hoặc theo vị trí địa lý (3 cấp: Phường xã => Quận huyện => Tỉnh thành). Sử dụng kiểu phân cấp này, người dùng có thể phân tích, so sánh số liệu bán hàng theo từng chi nhánh hoặc từng tỉnh thành rất dễ dàng.

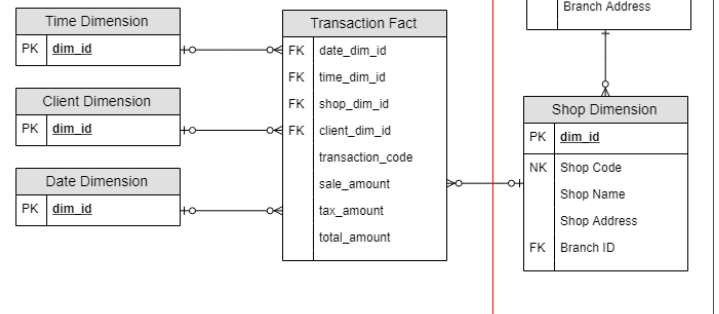
Mục 1.2.2 có mô tả thiết kế bảng dimension theo hướng hình sao (star schema) hoặc hình bông tuyết (snowflake schema) chính là nói đến phương án lưu trữ các phân cấp dimension trong bảng vật lý. Thiết kế theo hướng hình sao tức là lưu tất cả các cấp khác nhau của dimension vào một bảng duy nhất, ngược lại thiết kế theo hướng bông tuyết là chia nhỏ, mỗi bảng vật lý lưu thông tin một cấp của dimension. Tham khảo thêm hình 4.2 để nắm được mô hình quan hệ của 2 phương án thiết kế nói trên.



## Thiết kế hình sao



## Thiết kế hình bông tuyết



Hình 4.2: So sánh thiết kế dimension Shop theo hình sao và hình bông tuyết

## 4.3 Cập nhật dữ liệu bảng dimension

Dữ liệu trong dimension không cố định mà thường thay đổi theo thời gian (ví dụ nhân viên thuộc chi nhánh này, tháng sau chuyển sang chi nhánh khác). Để thỏa mãn tính nhất quán dữ liệu, đảm bảo số liệu trong quá khứ không bị thay đổi khi dữ liệu hiện tại được cập nhật, lý thuyết kho dữ liệu cung cấp 4 phương án cập nhật dữ liệu dimension gọi là Slowly Changing Dimension (SCD) như sau:

### 1) SCD loại 1: ghi đè thông tin lên bản ghi cũ

Với phương án này, khi hệ thống nghiệp vụ cập nhật thông tin, kho dữ liệu cập nhật thông tin mới vào bản ghi cũ. Phương án này thường dùng trong trường hợp người dùng trên hệ thống nghiệp vụ nhập liệu sai chính tả như sai tên, sai email, địa chỉ... và được nhân viên vận hành khắc phục bằng tay. Ngoài ra SCD loại 1 còn được sử dụng khi một số cột dữ liệu dimension thay đổi thông tin quá nhanh (trường hợp Mini dimension, sẽ trình bày chi tiết ở các mục sau).

ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận
1	A	1/4/2019	CV001	Lưu Đức Thắnggg	Phát triển
ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận
1	A	1/4/2019	CV001	Lưu Đức Thắng	Phát triển

Hình 4.3: Phương án cập nhật dữ liệu SCD loại 1

SCD loại 1 có ưu điểm là dễ xây dựng và sử dụng, cách thức vận hành tương tự hệ thống nghiệp vụ. Nhược điểm của nó là chỉ ghi nhận được thông tin mới nhất, làm mất mát thông tin về các lần cập nhật dữ liệu trước đó.

## 2) SCD loại 2: thêm bản ghi mới và vô hiệu hoá bản ghi cũ

Ở phương án 2 này, mỗi khi hệ thống nghiệp vụ cập nhật, kho dữ liệu vẫn giữ bản ghi cũ trong bảng dữ liệu nhưng vô hiệu hoá nó đi và thêm bản ghi mới. Phương án này thường được sử dụng khi người dùng muốn đảm bảo tính nhất quán của dữ liệu: việc thay đổi số liệu ở thời điểm hiện tại không làm ảnh hưởng đến số liệu trong quá khứ. Sử dụng phương án này, việc một nhân viên chuyển từ chi nhánh A sang chi nhánh B không làm thay đổi số liệu báo cáo doanh thu của 2 chi nhánh trước thời điểm điều chuyển.

ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận
1	A	1/4/2019	CV001	Lưu Đức Thắng	Phát triển
ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận
1	I	1/4/2019	CV001	Lưu Đức Thắng	Phát triển
2	A	1/4/2019	CV001	Lưu Đức Thắng	Kho dữ liệu

Hình 4.4: Phương án cập nhật dữ liệu SCD loại 2

Ưu điểm của phương án này là có thể theo dõi sự thay đổi thông tin trong cả vòng đời của đối tượng. Tuy nhiên việc phát triển, vận hành và sử dụng bảng dimension theo phương án này lại phức tạp hơn các phương án khác. Thêm nữa dung lượng bảng SCD 2 dễ bị phình to trong trường hợp dimension thay đổi thông tin liên tục.

Nếu thiết kế bảng dimension theo phương án này, tác giả thường thêm 4 cột dữ liệu vào bảng như sau:

- Cột DIM\_ID: khoá thay thế của bảng dimension (đã trình bày ở mục 4.1)
- Cột DIM\_STATUS: lưu trạng thái của bản ghi dimension, gồm 2 giá trị Active (đang hoạt động) và Inactive (vô hiệu hoá). Các bản ghi mới nhất ở trạng thái Active, các bản ghi cũ hơn ở trạng thái Inactive.
- Cột DIM\_START\_TIME: ghi nhận thời gian thêm bản ghi dimension vào bảng dữ liệu
- Cột DIM\_END\_TIME: ghi nhận thời gian bản ghi dimension bị vô hiệu hoá

Khi sử dụng bảng SCD 2, một đối tượng dimension được ghi nhận ở nhiều bản ghi khác nhau tuy nhiên tại mỗi thời điểm chỉ có 1 bản ghi ở trạng thái hoạt động. Khi sử dụng bảng này cần ghi nhớ nguyên tắc:

- Query dữ liệu dimension bằng DIM\_ID **không được** thêm điều kiện DIM\_STATUS

- Query dữ liệu dimension theo một cột thông tin nào đó **phải có** điều kiện DIM\_STATUS='A' (để lấy trạng thái mới nhất của đối tượng)

Dữ liệu cột DIM\_ID là chuỗi ký tự hoặc số vô nghĩa dùng làm khoá chính của bảng dimension (như đã trình bày ở mục 4.1) và thay đổi liên tục nên thông thường người dùng không query dữ liệu bằng cột DIM\_ID này mà chỉ dùng nó để join từ bảng fact.

Theo quy tắc trên, script SQL lấy thông tin tất cả nhân viên bộ phận Kho dữ liệu như sau:

```
select *
from D_EMPLOYEES
where 'Bộ phận' = 'Kho dữ liệu' and DIM_STATUS = 'A';
```

### 3) SCD loại 3: thêm trường dữ liệu mới ghi nhận thông tin cũ

Ở phương án 3 này, người thiết kế bổ sung một cột dữ liệu mới để ghi nhận giá trị gần nhất của thuộc tính. Phương án này tương đối dễ xây dựng, sử dụng và vận hành. Điểm yếu là chỉ có thể theo dõi thông tin gần nhất chứ không lấy được cả vòng đời.

ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận	Bộ phận (cũ)
1	A	1/4/2019	CV001	Lưu Đức Thắng	Phát triển	
ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận	Bộ phận (cũ)
1	A	1/4/2019	CV001	Lưu Đức Thắng	Kho dữ liệu	Phát triển

Hình 4.5: Phương án cập nhật dữ liệu SCD loại 3

Theo tác giả, SCD loại 3 này ít được sử dụng vì tại thời điểm thiết kế thường không xác định được chính xác danh sách cột thông tin cần theo dõi. Giải pháp an toàn theo dõi tất cả các cột thông tin lại làm bảng bị rộng ra rất nhiều khiến việc phát triển, sử dụng phức tạp hơn.

### 4) SCD loại 4: chuyển bản ghi cũ về bảng dimension quá khứ

Theo phương án này, cần xây dựng một bảng vật lý bổ sung cho mỗi dimension để lưu dữ liệu quá khứ, còn bảng dimension chính chỉ có bản ghi mới nhất.

ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận
1	A	1/4/2019	CV001	Lưu Đức Thắng	Phát triển
ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận
2	A	1/4/2019	CV001	Lưu Đức Thắng	Kho dữ liệu
ID	Trạng thái bản ghi	Ngày kích hoạt	Mã nhân viên	Tên nhân viên	Bộ phận
1	A	1/4/2019	CV001	Lưu Đức Thắng	Phát triển

**Hình 4.6:** Phương án cập nhật dữ liệu SCD loại 4

Phương án này khá giống SCD kiểu 2, điểm khác là các bản ghi cũ được chuyển sang bảng dữ liệu khác chứ không lưu ở bảng chính. Thay đổi này làm bảng dimension dễ sử dụng hơn vì query tại thời điểm hiện tại không cần thêm điều kiện DIM\_STATUS = 'A' như SCD 2.

## 5) Nhận xét

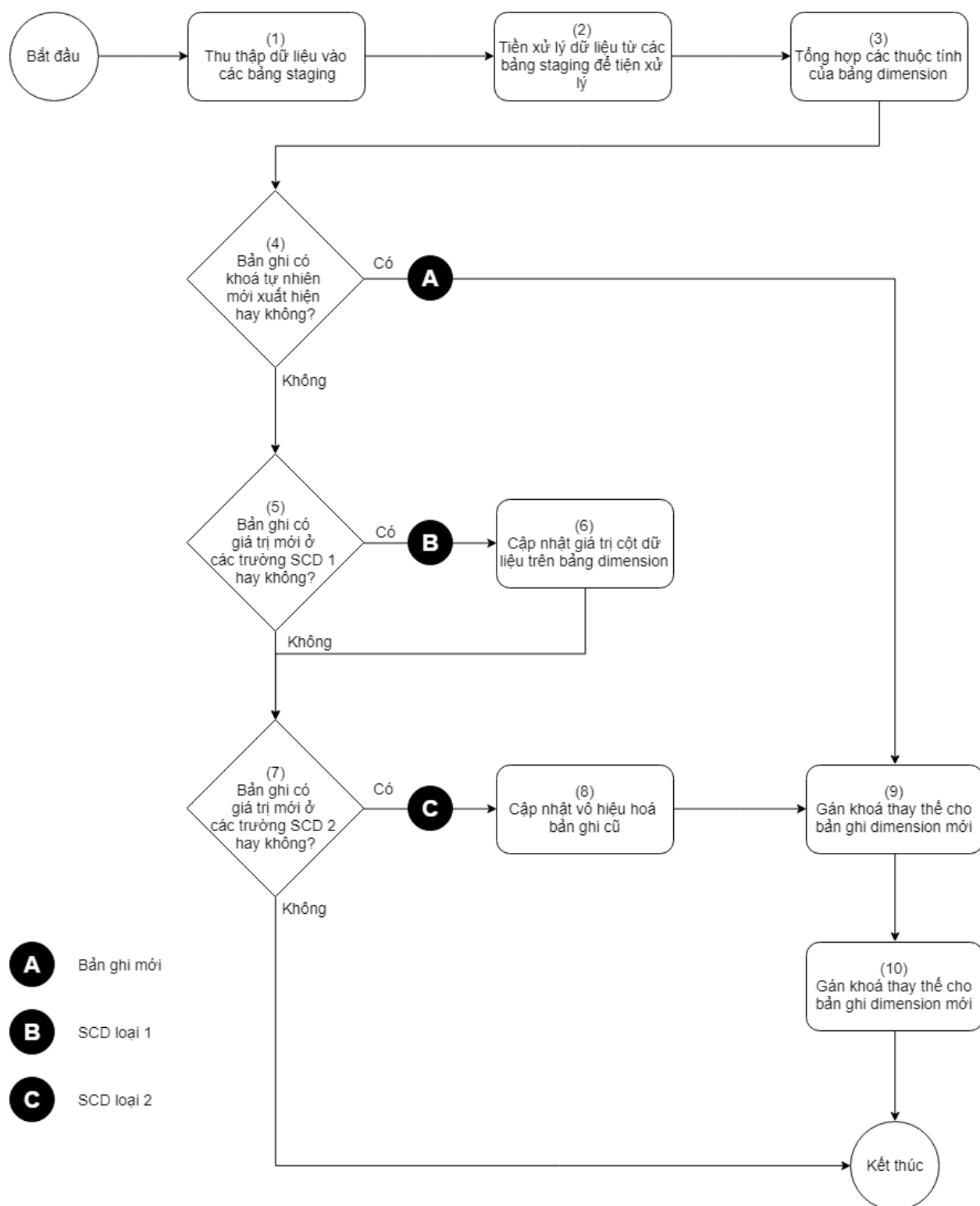
Theo quan điểm của tác giả, người sử dụng kho dữ liệu thường không dự đoán chính xác được nhu cầu sử dụng của bản thân. Do đó phương án SCD 1 và 3 thường không được thiết kế mà chỉ dùng dưới dạng ad-hoc, khi có sai sót từ hệ thống nghiệp vụ. Ngược lại phương án SCD 2 và 4 giúp theo dõi sự thay đổi thông tin trong vòng đời đối tượng dimension, lại không cần dự đoán trước nhu cầu nên hay được sử dụng.

Trong cả 4 phương án trên, SCD loại 2 đáp ứng được tất cả yêu cầu về theo dõi thay đổi của đối tượng chỉ trong một bảng dữ liệu, không cần merge nhiều bảng. Nhưng việc ghi nhớ nguyên tắc sử dụng bảng này cũng tương đối khó khăn, làm tăng nhiệt độ nổi loạn ếch lên đáng kể. Các bạn phân tích dữ liệu mức fresher hoặc junior thường gặp khó khăn khi dùng bảng này, làm sai sót quá trình lấy báo cáo.

Để giải quyết vấn đề trên, tác giả thường thiết kế các bảng dimension theo phương án SCD 2, tuy nhiên bổ sung thêm một bảng dữ liệu chỉ chứa các bản ghi được cập nhật nhất (bảng chính theo phương án SCD 4). Bảng dữ liệu bổ sung có thể là bảng vật lý, cũng có thể là view. Theo cách này, người dùng bình thường có thể tận dụng được sức mạnh của bảng SCD 2, trong khi các bạn ít kinh nghiệm hơn query trên bảng thứ hai để tránh trùng dữ liệu. Chú ý: đây không phải phương án hoàn hảo!

## 6) Cách xây dựng luồng ETL

Do có nhiều phương án cập nhật dữ liệu, việc xây dựng bảng dimension không còn đơn giản là chạy lệnh update nữa mà đòi hỏi rất nhiều công đoạn. Quy trình xử lý dữ liệu dimension tác giả thường sử dụng được thể hiện trong hình 4.7:



Hình 4.7: Quy trình xử lý dữ liệu dimension<sup>4</sup>

<sup>4</sup> Hình 17-2 sách "Star Schema: The Complete Reference" của tác giả Christopher Adamson

(1) Bước 1 của tiến trình xử lý dimension là đồng bộ dữ liệu từ hệ thống nghiệp vụ vào các bảng staging. Việc đồng bộ này có thể theo dạng incremental (chỉ lấy các bản ghi mới nhất rồi xử lý đồng bộ bằng ETL) hoặc đồng bộ full (lấy tất cả các bản ghi, bao gồm các bản ghi đã lấy từ lần đồng bộ trước nhưng chưa có cập nhật). Phương án đồng bộ incremental chỉ thực hiện được trên các bảng dữ liệu có khoá chính và có lưu trữ thời gian cập nhật dữ liệu cuối cùng của từng bản ghi, khó thực hiện và gây rủi ro mất đồng bộ. Phương án đồng bộ full lấy toàn bộ bản ghi trong bảng dữ liệu, có phương án xử lý rất đơn giản nhưng tạo gánh nặng cho hệ thống nghiệp vụ. Người phát triển ETL cần lựa chọn phương án hợp lý cân bằng giữa quá trình phát triển – vận hành kho dữ liệu và khả năng chịu tải hệ thống nghiệp vụ.

(2) Tiền xử lý dữ liệu hệ thống nghiệp vụ theo các quy tắc trên dimension như chuẩn hoá đơn vị đo, đơn vị tiền tệ, xử lý bản ghi trùng, phẳng hoá các thuộc tính (trường hợp bảng nguồn theo mô hình EAV).

(3) Tái tạo lại tất cả các đối tượng trong bảng dimension ở trạng thái mới nhất

(4) Kiểm tra bảng tạm ở bước (3) xem có đối tượng dimension nào mới hay không (bằng cách kiểm tra khoá tự nhiên). Nếu có đối tượng mới thì gán khoá thay thế cho các đối tượng đó (bước 9) rồi lưu bản ghi vào bảng dimension (bước 10).

(5) Với các đối tượng dimension cũ, kiểm tra xem cột dữ liệu SCD 1 có giá trị mới hay không, nếu có giá trị mới thì cập nhật cột đó vào bảng dimension (bước 6).

(6) Cũng trong luồng xử lý đối tượng dimension cũ, kiểm tra xem cột dữ liệu SCD 2 có giá trị mới hay không. Nếu không có giá trị mới thì ngừng luồng ETL. Nếu có giá trị mới thì cập nhật trạng thái các bản ghi cũ về “Vô hiệu hoá” (Inactive – ‘I’) (bước 8), gán khoá thay thế cho bản ghi mới (bước 9) rồi lưu bản ghi đó vào bảng dimension (bước 10).

Quy trình trên áp dụng cho SCD loại 1, loại 2 nhưng có thể điều chỉnh cho SCD loại 3, loại 4.

## 4.4 Nguyên tắc thiết kế

Tương tự bảng fact, việc thiết kế một bảng dimension cần tối thiểu các thông tin sau:

- Mô tả ý nghĩa bảng dimension
- Mô tả các phân cấp dữ liệu và phương án thiết kế bảng dimension (hình sao hay hình bông tuyết)
- Mô tả phương thức cập nhật dữ liệu (loại SCD)
- Mô tả nguồn dữ liệu của bảng dimension: Có dimension lấy dữ liệu từ nhiều nguồn khác nhau. Có trường hợp việc extract dữ liệu phải lọc theo những điều kiện nhất định.
- Mô tả nguồn dữ liệu của dimension:
  - Trong nhiều trường hợp bảng dimension có dữ liệu từ nhiều nguồn khác nhau, khi đó nên chỉ rõ nguồn nào là chính, nguồn nào bổ sung
  - Lấy dữ liệu từ những nguồn nào chi tiết đến mức bảng dữ liệu và điều kiện query trên mỗi bảng
  - Cần cập nhật file thiết kế trong trường hợp có thay đổi từ hệ thống nghiệp vụ: thêm nguồn dữ liệu, sửa logic bảng dữ liệu...

- Có sơ đồ thể hiện quan hệ của bảng dimension với các dimension có liên quan (nếu có)
- Liệt kê danh sách và mô tả chi tiết các cột dữ liệu của dimension. Chú ý mô tả kỹ các cột dữ liệu có công thức tính toán phức tạp, các cột lấy dữ liệu từ nhiều nguồn khác nhau, các cột áp dụng SCD loại 1 và loại 3.
- Danh sách index và partition sử dụng trong dimension: mô tả công thức index, partition
- Các mô tả, ghi chú khác về dimension (nếu có)

Ví dụ bảng dimension Cửa hàng trên hệ thống DF Corp được mô tả trên bảng 4.1

MÔ TẢ DIMENSION CỬA HÀNG			
Tên bảng dimension	Dimension cửa hàng		
Phương án thiết kế	Hình sao		
Các phân cấp trong bảng	<ul style="list-style-type: none"> <li>- Cửa hàng =&gt; Chi nhánh</li> <li>- Cửa hàng =&gt; Phường/xã =&gt; Quận/huyện =&gt; Tỉnh/thành phố</li> </ul>		
Tên bảng dimension trong kho dữ liệu	D_SHOPS		
Tần suất cập nhật	Hàng ngày		
Phương pháp cập nhật	SCD loại 2		
Nguồn dữ liệu	<ul style="list-style-type: none"> <li>- Dữ liệu cửa hàng lấy từ bảng tblShops hệ thống bán hàng</li> <li>- Dữ liệu chi nhánh lấy từ bảng tblBranches hệ thống bán hàng</li> <li>- Dữ liệu phường xã lấy từ bảng tblPrecints hệ thống bán hàng</li> <li>- Dữ liệu Quận huyện lấy từ bảng tblDistricts hệ thống bán hàng</li> <li>- Dữ liệu Tỉnh thành lấy từ bảng tblProvinces hệ thống bán hàng</li> </ul>		
Mô tả	Bảng dữ liệu lưu thông tin dimension Cửa hàng		
STT	Tên cột	Kiểu dữ liệu	Mô tả
1	DIM_ID	STRING	Khoá thay thế bảng dimension
2	DIM_STATUS	STRING	Trạng thái bản ghi dimension
3	DIM_START_TIME	DATETIME	Thời gian bản ghi dimension bắt đầu có hiệu lực
4	DIM_END_TIME	DATETIME	Thời gian bản ghi dimension hết hiệu lực
5	SHOP_ID	STRING	Mã cửa hàng
6	SHOP_NAME	STRING	Tên cửa hàng

7	SHOP_ADDRESS	STRING	Địa chỉ cửa hàng dạng free text
8	SHOP_PRECINT	STRING	Phường xã đặt cửa hàng
9	SHOP_DISTRICT	STRING	Quận huyện đặt cửa hàng
10	SHOP_PROVINCE	STRING	Tỉnh thành đặt cửa hàng
11	BRANCH_ID	STRING	Mã chi nhánh
12	BRANCH_NAME	STRING	Tên chi nhánh
13	DESCRIPTION	STRING	Ghi chú
<b>Index và Partition</b>			
Đánh partition trường DIM_STATUS <ul style="list-style-type: none"> <li>- Kiểu partition: List</li> <li>- Các giá trị: 'I', 'A', null</li> </ul> Các index: <ul style="list-style-type: none"> <li>- DIM_ID</li> <li>- SHOP_ID</li> <li>- BRANCH_ID</li> <li>- PRECINCT_ID</li> <li>- DISTRICT_ID</li> <li>- PROVINCE_ID</li> </ul>			
<b>Ghi chú</b>			
- Bổ sung bảng D_SHOPS_LATEST chỉ chứa các bản ghi cửa hàng mới nhất			

**Bảng 4.1:** Mô tả bảng dimension Cửa hàng hệ thống DF Corp

## 4.5 Một số bảng dimension đặc biệt

### 4.5.1 Dimension Ngày tháng

Khi thiết kế kho dữ liệu, có một dimension được chia sẻ chung cho tất cả các bảng fact, chính là **dimension Ngày tháng**. Các phép đo đặc, các sự kiện đều cần ghi nhận thời gian xảy ra phép đo hoặc sự kiện đó.



Calendar Dimension	
SK	dim_id
	day
	day_in_month
	day_in_week
	month
	month_name
	quarter
	quarter_name
	year
	is_weekend
	is_holiday

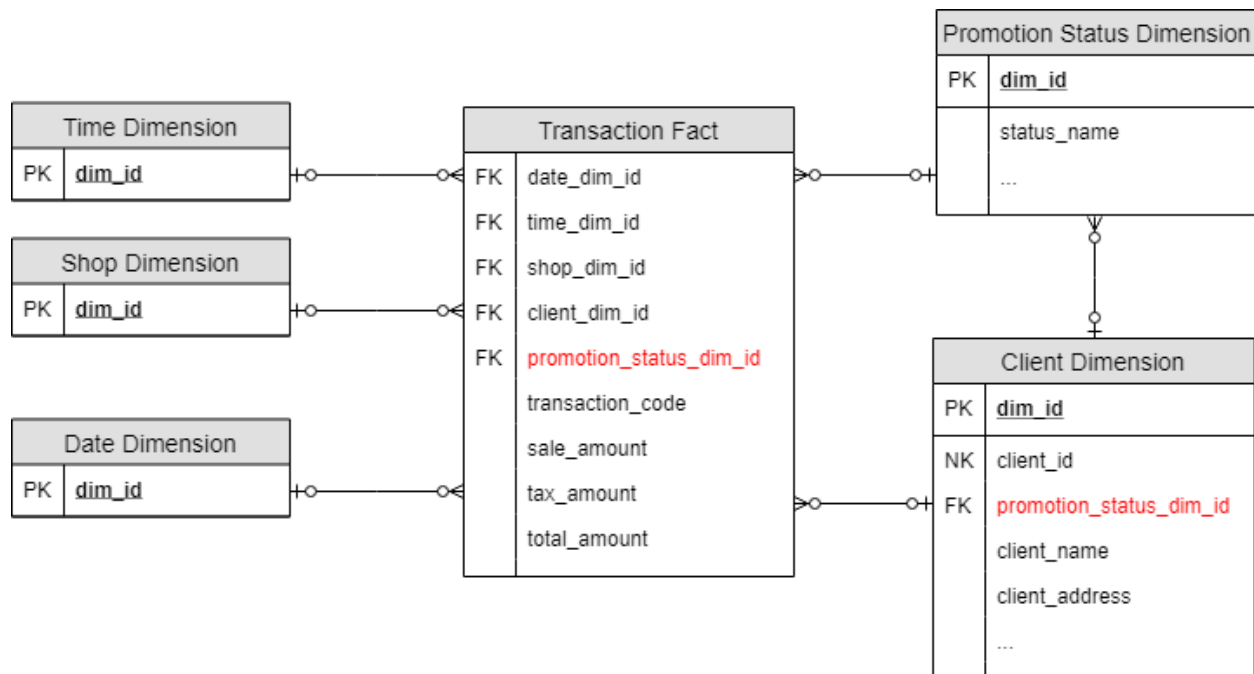
**Hình 4.8:** Dimension Ngày tháng

Dimension này không có nguồn từ hệ thống nghiệp vụ mà thường được sinh ra bằng cách viết code sinh dữ liệu (dùng SQL, Excel hoặc một ngôn ngữ lập trình nào đó). Bảng dimension này được tạo một lần vào đầu dự án rồi giữ nguyên trong suốt vòng đời xây dựng, vận hành, nâng cấp dự án kho dữ liệu. Việc thay đổi (nếu có) chỉ xảy ra định kỳ một năm một lần vì một số ngày nghỉ lễ ở Việt Nam tính theo lịch âm không có ngày dương cố định.

Không như các bảng dimension khác dùng mã thay thế, bảng dimension Ngày tháng thường được tác giả thiết kế có khoá thay thế có ý nghĩa, được mã hoá từ ngày của bản ghi theo format YYYYMMDD (ví dụ bản ghi của ngày 23 tháng 05 năm 2019 có giá trị DIM\_ID = 20190523. Thiết kế này phá vỡ nguyên tắc thiết kế khoá thay thế (đã trình bày ở mục 4.1), nhưng có lợi giúp cho việc partition và query theo partition trên bảng fact dễ dàng hơn.

#### 4.5.2 Dimension có thông tin thay đổi quá nhanh

Khi thiết kế dimension, đôi khi ta gặp trường hợp bảng dimension có một hoặc một vài cột thông tin nào đó có tần suất thay đổi với tần suất lớn, ví dụ như giá hàng hoá, khách hàng được khuyến mãi... Trong lý thuyết kho dữ liệu, các cột thông tin đó gọi là **Mini Dimension**. Sau khi khảo sát, nếu người dùng có nhu cầu phân tích sự biến thiên dữ liệu ta có thể thiết kế các cột đó thành một bảng fact snapshot. Ngược lại, nếu người dùng không có nhu cầu phân tích sự biến đổi thông tin thì thiết kế các cột dữ liệu đó theo kiểu SCD loại 1. Trong nhiều trường hợp, một số cột thông tin tự thân nó không có ý nghĩa nhưng lại trở nên quan trọng khi phân tích kết hợp với bảng fact. Khi đó, ta tạo một bảng dimension thứ ba (ứng với mỗi nhóm cột), bổ sung dimension đó vào bảng fact và thiết kế cột dữ liệu ở bảng dimension theo hướng SCD 1.



**Hình 4.9:** Phương án thiết kế Mini Dimension

Trong hệ thống DF Corp, Trạng thái khuyến mãi (PROMOTION\_STATUS) của bảng dimension Khách hàng chính là một cột như vậy. Theo mô tả, nếu tổng tiêu dùng tháng trước của khách đạt trên 5 triệu thì tháng này khách được hoàn 10% giá trị đơn hàng các ngày thứ sáu. Việc tính toán này được thực hiện hàng tháng, trạng thái của khách cũng thay đổi liên tục. Nếu thiết kế theo SCD 2, mỗi tháng sẽ phát sinh một số lượng lớn bản ghi ở bảng dimension Khách hàng vì việc chuyển trạng thái này. Theo phương án trên, ta xây dựng bảng dimension Promotion Status Dimension (D\_PROMOTION\_STATUS), liên kết dimension đó với bảng Transaction Fact và Client Dimension. Mỗi khi trạng thái khuyến mãi có thay đổi, cột thông tin promotion\_status\_dim\_id ở bảng Client Dimension được ghi đè giá trị mới theo phương pháp SCD 1 chứ không tạo thêm bản ghi như khi thay đổi thông tin các cột khác.

### 4.5.3 Dimension rác

Khi xây dựng dimension đôi khi ta gặp trường hợp một số cột dữ liệu trong bảng fact có thể quy thành dimension nhưng số lượng giá trị trong mỗi cột rất ít, đôi khi lại không có thông tin mô tả thêm và chỉ được sử dụng ở một bảng fact duy nhất. Các cột đó có thể gom nhóm thành một bảng dimension gọi là **dimension rác (junk dimension)**. Bảng dimension đó lưu tích Đều-các<sup>5</sup> tất cả các giá trị của từng cột dữ liệu.

Trên hệ thống DF Corp, bảng fact Thanh toán có thể sử dụng một bảng dimension rác như vậy. Khi thanh toán, khách hàng có thể lựa chọn nhiều phương thức trả tiền khác nhau như thẻ, ví điện tử, tiền mặt, điểm; nhiều loại thẻ khác nhau như thẻ tín dụng, thẻ tín chấp; nhiều nhà cung cấp khác nhau như Visa, Mastercard, JCB, NAPAS... Người thiết kế có thể xây dựng bảng dữ

<sup>5</sup> [https://en.wikipedia.org/wiki/Cartesian\\_product](https://en.wikipedia.org/wiki/Cartesian_product)

liệu cho mỗi lựa chọn nói trên, tuy nhiên cách làm đó làm tăng số lượng bảng dimension lên thêm 3 bảng mà các bảng đó có giá trị sử dụng rất hạn chế (có ích duy nhất với bảng fact Thanh toán). Thay vào đó, ta tạo một bảng dimension rác Phương thức thanh toán (D\_PAYMENT\_METHODS) với các giá trị sau:

Phương thức thanh toán	Kiểu thanh toán	Nhà cung cấp
Thẻ	Tín dụng	Visa
Thẻ	Tín chấp	NAPAS
Ví điện tử	Tín chấp	Momo
Tiền mặt	Tiền mặt	Tiền mặt
Điểm	Đổi điểm	DF Corp
...	...	...

**Bảng 4.2:** Các giá trị trong bảng dimension rác Phương thức thanh toán

#### 4.5.4 Dimension từ điển

Dimension từ điển là một bảng dimension đặc biệt, không liên kết với bảng fact mà chỉ có quan hệ với các bảng dimension khác. Mục đích của bảng này là lưu và đảm bảo thông tin một phân cấp dimension nào đó được nhất quán, giúp tránh được mất đồng bộ khi dữ liệu đến từ nhiều hệ thống khác nhau. Trên hệ thống DF Corp, một ví dụ cho bảng dimension từ điển là dimension Giới tính. Trên nhiều hệ thống nghiệp vụ, giới tính của khách hàng được đánh dấu bằng nhiều loại mã khác nhau như 'M', 'Male', 'male'... cho nam; 'F', 'Female', 'female'... cho nữ. Qua việc sử dụng dimension Giới tính, trường GENDER trong dimension Khách hàng sẽ có giá trị nhất quán hơn là 'Nam', 'Nữ'... Dimension từ điển cũng có thể được dùng để bổ sung thông tin cho một phân cấp nào đó như dân số, loại địa hình, mức thu nhập... cho thông tin vị trí địa lý

Việc thiết kế dimension từ điển y hệt như một dimension thông thường. Điểm khác duy nhất là lúc này bảng dữ liệu được liên kết là một bảng dimension chứ không phải bảng fact.

#### 4.5.5 Dimension có nhiều bảng vật lý

Nhu cầu về độ chi tiết dữ liệu trên cùng một dimension của các bảng fact khác nhau khiến ta phải thiết kế các bảng dimension giống nhau nhưng có độ chi tiết khác nhau. Các bảng này thường xuất hiện chung trong tài liệu thiết kế và chung luồng xử lý dữ liệu ETL.

Hệ thống DF Corp có 2 dimension Cửa hàng và Máy POS có tính chất như vậy. Về mặt cây phân cấp, cửa hàng là cấp độ cao hơn của máy POS nên dimension Thanh toán có thể thông qua máy POS để lấy thông tin cửa hàng. Theo nguyên tắc tương tự, ta có thể xây dựng bảng dimension Chi nhánh để dùng cho bảng fact tổng hợp Chi nhánh.

POS Dimension	
PK	dim_id
NK	POS Code
	POS Name
	Shop Name
	Shop Address
	Branch Code
	Branch Name
	Branch Address
	---

Shop Dimension	
PK	dim_id
NK	Shop Code
	Shop Name
	Shop Address
	Branch Code
	Branch Name
	Branch Address
	---

Branch Dimension	
PK	dim_id
NK	Branch Code
	Branch Name
	Branch Address
	---

**Hình 4.10:** Phương án thiết kế 3 bảng dimension Máy POS – Cửa hàng – Chi nhánh

Gần giống nguyên tắc trên nhưng dimension Ngày tháng có thể không cần thêm bảng dữ liệu nào. Như mục 4.5.1 đã trình bày, bảng này có giá trị cột khoá chính DIM\_ID mã hoá từ ngày tháng nên có thể dùng ngày 1 làm bản ghi đại diện cho tháng mong muốn, ngày 1 tháng 1 làm bản ghi đại diện cho năm mong muốn.

Trước đây tác giả từng mắc sai lầm trong việc thiết kế loại bảng này khi cố nhồi nhét tất cả các cấp độ dữ liệu vào một bảng vật lý. Theo thiết kế (sai lầm) đó, tác giả chỉ xây dựng một bảng fact Máy POS để lưu các bản ghi của máy POS, của cửa hàng, của chi nhánh. Muốn query cấp độ nào, người phân tích thêm điều kiện LEVEL = 'Cấp độ' vào script. Cách này làm giảm số lượng bảng dữ liệu nhưng khiến thiết kế không tường minh, yêu cầu người dùng phải đọc kỹ hướng dẫn sử dụng để tránh query sai.

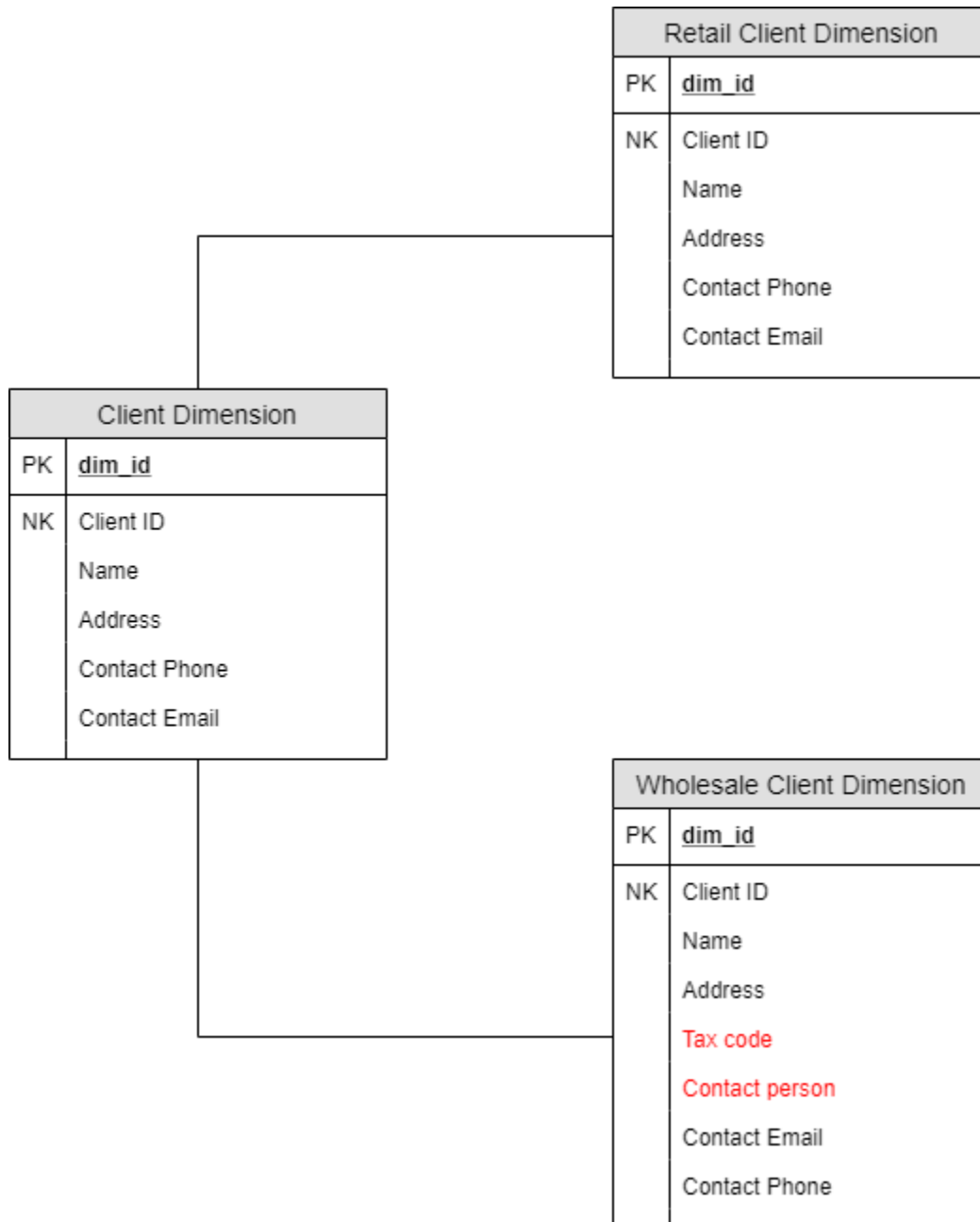
#### 4.5.6 Dimension có nhiều thuộc tính đặc thù

Cùng một dimension nhưng nhiều bản ghi lại có những thuộc tính đặc thù chỉ riêng bản ghi (hoặc nhóm bản ghi) đấy mới có, các bản ghi còn lại không có. Nếu người dùng cần phân tích các thuộc tính đặc thù đó, ta có 2 lựa chọn:

1. Bổ sung các thuộc tính đặc thù vào bảng dimension. Đối tượng nào không có thuộc tính thì cột đó nhận giá trị null. Phương án này được sử dụng khi số lượng cột dữ liệu đặc thù có hạn, có thể dự báo được.
2. Xây dựng một bảng dữ liệu chỉ chứa các thuộc tính chung và tất cả các bản ghi của dimension cần thiết kế. Sau đó xây dựng thêm các bảng dimension con, mỗi bảng chỉ chứa dữ liệu của một nhóm bản ghi nhất định nhưng bao gồm tất cả các thuộc tính đặc thù của nhóm bản ghi đó. Phương án này được sử dụng khi số lượng cột dữ liệu đặc thù tương đối lớn và không dự đoán được, khiến cho việc kết hợp bảng không khả thi.

Qua khảo sát hệ thống DF Corp, ta có thể xây dựng dimension Cửa hàng theo phương án 1, dimension Sản phẩm, Khách hàng theo phương án 2. Tuy cùng được thiết kế theo phương pháp

EAV, số lượng thuộc tính của các cửa hàng không nhiều, không cần thiết phải tạo bảng riêng cho 3 nhóm cửa hàng, nhất là khi nhóm cửa hàng Trang web thương mại điện tử chỉ có 1 bản ghi. Ngược lại, số lượng bản ghi của dimension Khách hàng và Sản phẩm đủ lớn, phù hợp hơn cho việc tạo bảng. Với dimension Sản phẩm, không cần thiết phải tạo mỗi bảng con cho từng nhóm sản phẩm mà chỉ cần tạo bảng cho những nhóm sản phẩm được quan tâm nhất.



**Hình 4.11:** Phương án thiết kế 3 bảng của dimension Khách hàng

Hình 4.10 thể hiện phương án thiết kế 3 bảng vật lý dimension Khách hàng gồm 1 bảng chính, 1 bảng lưu dữ liệu khách hàng lẻ (Retail Client Dimension), 1 bảng lưu dữ liệu khách hàng buôn

(Wholesale Client Dimension). Ta thấy bảng khách hàng buôn có thêm 2 cột dữ liệu Tax code và Contact person mà 2 bảng còn lại không có; còn bảng khách hàng lẻ có cấu trúc y hệt bảng chính. Chú ý bảng khách hàng lẻ chỉ có bản ghi khách hàng lẻ, bảng khách hàng buôn chỉ có bản ghi khách hàng buôn, bảng chính có bản ghi của tất cả các khách hàng.

#### 4.5.7 Dimension có phân cấp không giới hạn

Trong bảng dữ liệu quan hệ, đôi khi ta gặp trường hợp một danh mục nào đó có quan hệ phân cấp không giới hạn, ví dụ như các thư mục trong ổ đĩa. Trong hệ thống DF Corp, dữ liệu có thể phân cấp không giới hạn ở bảng danh mục Nhóm sản phẩm (có quan hệ self-join với chính nó), cụ thể là một chiếc điện thoại có thể thuộc nhóm “Điện thoại iPhone 8” bên trong nhóm “Điện thoại iPhone” bên trong nhóm “Điện thoại thông minh” bên trong nhóm “Điện thoại” bên trong nhóm “Điện máy”.

Có 2 phương án thiết kế bảng dimension có phân cấp không giới hạn như trên. Phương án thứ nhất là xây dựng bảng trung gian cho mỗi cấp, phương án này có thể giải quyết vấn đề một cách tổng thể, tuy nhiên không thân thiện với người dùng. Thay vào đó, tác giả thường sử dụng phương án thứ hai: phẳng hoá mối quan hệ giữa các bảng này.

Tên sản phẩm	Nhóm cấp 1	Nhóm cấp 2	Nhóm cấp 3	Nhóm cấp 4	Nhóm cấp 5
Điện thoại iPhone 8 Plus	Điện máy	Điện thoại	Điện thoại thông minh	iPhone	iPhone 8
Áo sơ mi Việt Tiến màu trắng	Thời trang	Thời trang nam	Áo nam	Áo sơ mi nam	Áo sơ mi nam
Chai nước Aquafina 0.5l	Thực phẩm	Thức uống	Nước khoáng, nước tinh khiết	Nước khoáng, nước tinh khiết	Nước khoáng, nước tinh khiết

**Bảng 4.3:** Ví dụ bảng dimension Sản phẩm có cây phân cấp không giới hạn

Với thiết kế phẳng hoá này, người dùng có thể thực hiện phân tích theo gói sản phẩm rất dễ dàng, trực tiếp câu query trên dimension Sản phẩm mà không cần liên kết bảng tạm. Cần chú ý theo thiết kế này một số sản phẩm dừng lại ở cấp 3, cấp 4 nhưng vẫn điền đủ thông tin cho các cấp bị thiếu, với giá trị các cấp bị thiếu ở cấp thấp nhất có thể.

## Phần 5: Kinh nghiệm xây dựng – vận hành kho dữ liệu

Trong các phần trước, ta đã tìm hiểu qua các bước xây dựng một kho dữ liệu, đi vào chi tiết các phương án thiết kế bằng fact, bằng dimension. Phần này giới thiệu một số vấn đề tác giả gặp phải khi xây dựng và vận hành một kho dữ liệu, cũng như phương án khắc phục các vấn đề nói trên. Các phương án này do tác giả rút từ kinh nghiệm bản thân, có thể không phải tối ưu, rất cần đóng góp của bạn đọc.

### 5.1 Kết hợp dữ liệu từ nhiều nguồn khác nhau

Dữ liệu trong kho không đến từ một mà được tổng hợp từ nhiều nguồn khác nhau. Các nguồn đó có thể là các hệ thống đang hoạt động song song, cũng có thể là các phiên bản khác nhau của một hệ thống. Dữ liệu trên các hệ thống đó không phải lúc nào cũng đồng nhất để tập trung vào một nguồn. Như trên hệ thống DF Corp giả tưởng, 2 bảng dữ liệu Đơn hàng trên 2 hệ thống không có cột dữ liệu chung, không thể liên kết 2 bản ghi vào làm 1 để tiện truy vấn được. Hoặc trên một hệ thống khác tác giả từng làm việc, bảng dữ liệu Cửa hàng của 2 hệ thống dùng 2 bộ mã khác nhau làm ID khiến không liên kết cửa hàng được.

Không có cách giải quyết chung nào cho việc kết hợp dữ liệu nói trên. Để kết nối dữ liệu, người thiết kế phải có kiến thức nền về nghiệp vụ và các hệ thống phần mềm cần kết nối. Đây là công việc cần kiến thức và sự phối hợp của người quản lý hệ thống và sự thoả hiệp của người phân tích trong trường hợp không có giải pháp hoàn hảo.

Với các bảng danh mục dimension có số lượng bản ghi “nhỏ”, người dùng có thể tự liên kết dữ liệu từ các hệ thống khác nhau một cách thủ công. Khi giải quyết bài toán mã cửa hàng nói trên, tác giả lấy danh sách khách hàng có định danh trên hệ thống khách hàng trung thành, tìm xem các khách hàng đó mua hàng ở cửa hàng nào trên hệ thống bán hàng rồi liên kết 2 bản ghi.

Có trường hợp dữ liệu từ nhiều nguồn không thể kết hợp với nhau được, như 2 bảng Giao dịch trên hệ thống DF Corp đã trình bày trong các phần trước. Đây là điều rất đáng tiếc. Khi đó người vận hành kho dữ liệu cần tác động ngược lại hệ thống nghiệp vụ, yêu cầu bổ sung thêm thông tin cho đến khi đáp ứng được.

### 5.2 Xử lý bản ghi dimension về trễ

Dữ liệu về trễ là vấn đề rất đau đầu trong xây dựng và vận hành hệ thống kho dữ liệu, nhất là trong các hệ thống phân tán hoặc theo thời gian thật. Nguyên nhân về muộn thường là do mất đồng bộ giữa kho dữ liệu và hệ thống nghiệp vụ, do lỗi của hệ thống nghiệp vụ không đẩy dữ liệu kịp thời cho kho dữ liệu... Lúc này, bản ghi fact đã có đầy đủ ID nhưng bản ghi dimension lại không có ID đó nên không join được. Phương án xử lý của tác giả như sau:

- Lấy danh sách các ID không liên kết được, insert ID vào bảng dimension để lấy khoá thay thế DIM\_ID. Lúc này bản ghi dimension chỉ có khoá tự nhiên và khoá thay thế, không có thông tin nào khác. Trường hợp bảng dimension có mini dimension, gán cho nó giá trị mặc định (“Không xác định” hoặc một giá trị xác định trước).
- Chạy tiến trình xử lý bản ghi fact vừa nhận được như bình thường



- Khi nhận được bản ghi dimension về trễ, xử lý các cột dữ liệu theo phương án SCD 1

### 5.3 Kiểm tra tính đúng đắn của dữ liệu

Kho dữ liệu là một cấu phần quan trọng trong hệ thống hỗ trợ ra quyết định. Để đảm bảo tính đúng đắn của số liệu trong kho, mỗi bước xử lý dữ liệu đều phải được kiểm tra, sửa chữa kịp thời. Trong quá trình vận hành, việc sai số liệu thường do 2 nguyên nhân: (1) tiến trình hoặc công cụ xử lý dữ liệu có bug; và (2) hệ thống nghiệp vụ thay đổi logic dữ liệu. Ngoài ra còn nguyên nhân thứ ba thường gặp trong các team nhỏ do bộ phận phát triển/vận hành kho dữ liệu không áp dụng các nguyên tắc phát triển phần mềm, deploy code vô tội vạ không có kế hoạch trước.

Khi xây dựng kho dữ liệu, tác giả thường xây dựng các tiến trình kiểm tra dữ liệu sau:

- (1) Kiểm tra tính nhất quán giữa dữ liệu trong hệ thống nghiệp vụ và dữ liệu ở vùng staging
- (2) Kiểm tra tính nhất quán giữa dữ liệu vùng staging và vùng base
- (3) Kiểm tra tính nhất quán giữa dữ liệu vùng base và dữ liệu vùng mart
- (4) Kiểm tra tính nhất quán giữa dữ liệu các bảng khác nhau ở vùng mart
- (5) Trong một số trường hợp, kiểm tra tính nhất quán giữa dữ liệu các bảng khác nhau vùng staging.

Tính nhất quán ở đây được hiểu là độ chính xác của số liệu các tiêu chí trình bày ở mục 2.2 trên các bảng dữ liệu/các hệ thống khác nhau. Thông thường các phép đo kiểm tra tiến trình số (1), (2), (3), (4) cần có độ chính xác cao xấp xỉ 100%, riêng phép đo số (5) có thể có độ chính xác thấp hơn tùy tính chất bảng dữ liệu. Các tiến trình kiểm tra cần được chạy định kỳ ngay sau khi tiến trình xử lý dữ liệu hoàn thành và có cảnh báo cho các bên liên quan nếu số liệu một phép kiểm tra nào đó lệch vượt ngưỡng.

Ngoài việc kiểm tra, có thể bổ sung thêm các tiến trình cảnh báo trong trường hợp dữ liệu đột biến, trở lên quá ít hoặc quá nhiều so với kỳ trước. Các tiến trình cảnh báo này có thể là dấu hiệu cho thấy hệ thống gặp vấn đề cần phối hợp khắc phục từ hệ thống nghiệp vụ.

### 5.4 Quản trị dữ liệu

Quản trị dữ liệu là nói đến tổng hợp các quy trình quản lý và công cụ hỗ trợ giúp tổ chức truy cập dữ liệu đúng đắn, tránh rủi ro. Quản trị dữ liệu cũng là việc xây dựng các tiêu chuẩn, vai trò để đảm bảo việc khai thác dữ liệu mang lại hiệu quả cao nhất mà vẫn quản lý được chi phí/rủi ro trong sử dụng dữ liệu. Việc quản trị này giúp đảm bảo tính nhất quán, tính thống nhất và tính kỷ luật trong khai thác dữ liệu.

Quản trị dữ liệu là một công việc rất quan trọng, khó khăn và tinh tế. Trong phạm vi tài liệu này, tác giả không đi sâu vào lý thuyết phần công việc đó, thay vào đó tác giả liệt kê các công việc mà bộ phận xây dựng – vận hành kho dữ liệu có thể thực hiện. Cụ thể bao gồm:

#### 1) Đảm bảo dữ liệu trong kho đầy đủ, chính xác:

- Đảm bảo việc tích hợp dữ liệu bằng fact, bảng dimension được từ nhiều nguồn khác nhau, tránh trường hợp một sự kiện/sự vật nào đó được ghi nhận trùng lặp.



- Theo dõi tiến trình kiểm tra tính đúng đắn của dữ liệu, giải quyết khi gặp vấn đề
- Tham gia vào quá trình phát triển hệ thống nghiệp vụ, xem xét cho ý kiến đề xuất đảm bảo hệ thống nghiệp vụ đáp ứng được yêu cầu của bộ phận phân tích.
- Đảm bảo tiến trình xử lý dữ liệu trong kho đồng bộ với logic và kế hoạch nâng cấp hệ thống nghiệp vụ.
- Xây dựng môi trường dev/test/production cho việc xây dựng và vận hành kho dữ liệu. Nên giả lập môi trường dev và test bằng các tiến trình sinh dữ liệu ngẫu nhiên.

## 2) Đảm bảo người sử dụng khai thác được dữ liệu

- Xây dựng tài liệu ở mức tổng quan và chi tiết, giúp người dùng có cái nhìn tổng thể về hệ thống và đóng vai trò từ điển khi người dùng tìm kiếm thông tin
- Lưu trữ tài liệu hợp lý, cho phép người dùng tìm kiếm khi có nhu cầu
- Bổ sung thông tin kịp thời cho tài liệu khi có thay đổi

=> Nên sử dụng bộ phần mềm viết tài liệu như Wikipedia, Atlassian Confluence, Microsoft Sharepoint...

## 3) Bảo mật dữ liệu

- Áp dụng các biện pháp kỹ thuật đảm bảo người dùng chỉ có thể truy cập các vùng dữ liệu được phân quyền
- Áp dụng các biện pháp kỹ thuật che dấu, xóa bỏ dữ liệu định danh và nặc-danh-hóa<sup>6</sup> dữ liệu cung cấp cho người phân tích.
- Áp dụng các biện pháp kỹ thuật hoặc hành chính lập ma trận thể hiện mối quan hệ giữa người dùng và các nhóm dữ liệu trong kho

Công việc quản trị dữ liệu không thể chỉ trông vào tính tự giác của các nhân viên vận hành – khai thác kho dữ liệu mà phải có quy trình kiểm tra, đánh giá. Khi đội ngũ khai thác dữ liệu đủ lớn, cần có bộ phận hoặc nhân sự chuyên trách cho việc quản trị này.

## 5.5 Một số kinh nghiệm khác

Ngoài các kinh nghiệm mang tính hệ thống trên, tác giả còn một số chú ý nhỏ lẻ khác:

- Hệ thống nghiệp vụ thường không ổn định, rất hay xảy ra trùng lặp dữ liệu, một bản ghi xuất hiện 2 lần. Khi dựng luồng ETL xử lý bảng dimension cần có phương án giải quyết việc trùng lặp này
- Một số hệ thống nghiệp vụ không xóa hẳn dữ liệu mà chỉ vô hiệu hóa bản ghi (tương tự SCD 2). Khi dựng luồng ETL cần xác định bản ghi cập nhật nhất, bỏ qua các bản ghi cũ.
- Người vận hành kho dữ liệu nên lập danh sách nguồn đầu vào và người vận hành các hệ thống đó để liên hệ khi cần. Danh sách này cần được chia sẻ với tất cả thành viên trong bộ phận.

<sup>6</sup> <https://en.wikipedia.org/wiki/De-identification>

## Phần 6: Tổng kết

Vậy là xong! Tài liệu này được viết ra nhằm giới thiệu cho người đọc tổng quan về hệ thống kho dữ liệu, các thuật ngữ dùng trong hệ thống kho dữ liệu (phần 1), quy trình thiết kế tổng thể một kho dữ liệu (phần 2), các thiết kế bảng fact (phần 3), bảng dimension (phần 4), và một số kinh nghiệm của tác giả trong quá trình xây dựng – vận hành một hệ thống kho dữ liệu (phần 5).

Kho dữ liệu là một ngành rộng, có nhiều hướng tiếp cận khác nhau. Cụ thể là tài liệu viết lại theo những lý thuyết và sách đã xuất bản của tiến sỹ Ralph Kimball, một tên tuổi lớn trong ngành. Hướng dẫn xây dựng và ví dụ trong tài liệu có thể không còn đúng khi xét trên một hướng tiếp cận khác (ví dụ hướng tiếp cận của tiến sỹ Bill Inmon, một tượng đài khác).

Sau cùng, việc xây dựng kho dữ liệu thiên về thuật ngữ “thiết kế”, tùy mỗi trường phái khác nhau, mỗi người khác nhau có thể tạo nên những cấu trúc khác nhau nhưng đều đáp ứng yêu cầu. Bỏ qua các chi tiết kỹ thuật lằng nhằng, cốt yếu nhất của xây dựng kho dữ liệu vẫn là phản ánh đúng hệ thống nghiệp vụ, và đưa ra được các thông tin quý giá nhất cho người dùng cuối. Hệ thống nào làm được, hệ thống đó thành công.

## Tài liệu tham khảo

Trong quá trình làm việc, tác giả thường sử dụng các sách tham khảo sau:

- 1/ "The Data Warehouse Toolkit, 3rd Edition" (2013) by Ralph Kimball et al
- 2/ "The Data Warehouse ETL Toolkit: Practical techniques for Extracting, Cleaning, Conforming and Delivering Data" (2004) by Ralph Kimball et al
- 3/ "Star Schema: The Complete Reference" (2010) by Christopher Adamson