Mục lục

[I. Lý do chọn đề tài 3](#_Toc137135179)

[II. Đối tượng và phạm vi 4](#_Toc137135180)

[III. Ý nghĩa khoa học và thực tiễn của đề tài 4](#_Toc137135181)

[CHƯƠNG I: TỔNG QUAN VỀ HỆ THỐNG KHO DỮ LIỆU VÀ THIẾT KẾ MÔ HÌNH 7](#_Toc137135182)

[Tổng quan về Steam 7](#_Toc137135183)

[Tổng quan kho dữ liệu 7](#_Toc137135184)

[Kiến trúc chung của Data Warehouse 9](#_Toc137135185)

[Mô hình logic của kho dữ liệu 10](#_Toc137135186)

[Mô hình dữ liệu đa chiều 13](#_Toc137135187)

[Tổng quan về dữ liệu 14](#_Toc137135188)

[1.1. Dữ liệu là gì? 14](#_Toc137135189)

[1.2. Cách thức lưu trữ: 14](#_Toc137135190)

[1.3. Kiểu dữ liệu: 15](#_Toc137135191)

[Tổng quan về mô hình Datavault 15](#_Toc137135192)

[Thu thập và phân tích các yêu cầu về dữ liệu 16](#_Toc137135193)

[Phân tích quy trình nghiệp vụ của nền tảng Steam 17](#_Toc137135194)

[Hoa hồng 17](#_Toc137135195)

[Bán phần cứng 17](#_Toc137135196)

[Phí nhà phát triển 17](#_Toc137135197)

[Marketing 18](#_Toc137135198)

[Xây dựng hệ thống Data Warehouse 18](#_Toc137135199)

[Thiết kế cơ sở dữ liệu lớp HUB 20](#_Toc137135200)

[Thuộc tính của lớp HUB: 21](#_Toc137135201)

[Quy tắc thiết kế lớp HUB: 21](#_Toc137135202)

[Xử lý dữ liệu của lớp HUB: 21](#_Toc137135203)

[Thiết kế cơ sở dữ liệu lớp LINK 21](#_Toc137135204)

[Quy tắc thiết kế cho lớp LINK gồm: 21](#_Toc137135205)

[Cách thiết kế lớp LINK thường bao gồm các bước sau: 21](#_Toc137135206)

[Cách xử lý dữ liệu trong lớp LINK bao gồm: 22](#_Toc137135207)

[Thiết kế cơ sở dữ liệu lớp SATELLITE 22](#_Toc137135208)

[Quy tắc thiết kế lớp SATELLITE có thể được mô tả như sau: 22](#_Toc137135209)

[Các quy tắc thiết kế lớp SATELLITE: 23](#_Toc137135210)

[Cách xử lý dữ liệu trong lớp SATELLITE bao gồm: 23](#_Toc137135211)

[Các quy trình ETL 23](#_Toc137135212)

[Phân tích dữ liệu 24](#_Toc137135213)

[Thiết kế database 24](#_Toc137135214)

[Các loại bảng SCD 26](#_Toc137135215)

[Áp dụng cho bài toán triển khai mô hình datavault cho ứng dụng phát hành game trực tuyến 29](#_Toc137135216)

[Mô tả chi tiết các thuộc tính trong Data vault 31](#_Toc137135217)

[Xây dựng các tầng Data mart 37](#_Toc137135218)

[Thiết kế ETL (Extract-Transform-Load) và các công cụ hỗ trợ 46](#_Toc137135219)

[Tóm tắt: 48](#_Toc137135220)

[CHƯƠNG II: XÂY DỰNG HỆ THỐNG DATA WAREHOUSE SÀN THƯƠNG MẠI ĐIỆN TỬ STEAM TÍCH HỢP ĐA NGUỒN 50](#_Toc137135221)

[1. Tổng quan về hệ thống kho dữ liệu Steam 50](#_Toc137135222)

[2. Mục tiêu của hệ thống dữ liệu đa chiều áp dụng trong sàn thương mại điện tử Steam 51](#_Toc137135223)

[3. Tổng quan về Amazon Web Service (AWS) 52](#_Toc137135224)

[4. Các dịch vụ lưu trữ và tính toán 54](#_Toc137135225)

[EC2 (Elastic Compute Cloud): 54](#_Toc137135226)

[S3 (Simple Storage Service): 54](#_Toc137135227)

[EBS (Elastic Block Store): 54](#_Toc137135228)

[Glacier: 54](#_Toc137135229)

[RDS (Relational Database Service) 55](#_Toc137135230)

[Lambda: 55](#_Toc137135231)

[Redshift: 55](#_Toc137135232)

[Glue: 55](#_Toc137135233)

[Autoscaling: 56](#_Toc137135234)

[5. Triển khai AWS cho mô hình datavault 56](#_Toc137135235)

[6. Công cụ hỗ trợ ETL – Dbeaver 68](#_Toc137135236)

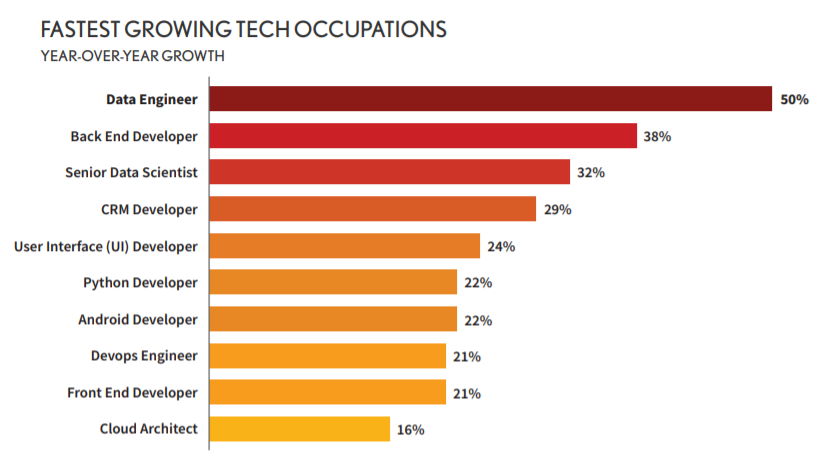
[7. Tổng quan về Tableau 71](#_Toc137135237)

[Tóm tắt 73](#_Toc137135238)

# Lý do chọn đề tài

Trong thời đại số hóa như hiện nay, việc quản lý dữ liệu là một trong những yếu tố quan trọng nhất đối với các công ty, đặc biệt là trong lĩnh vực thương mại điện tử. Các hệ thống xử lí thông tin và dữ liệu nên được tích hợp và quản lý một cách chặt chẽ để đảm bảo hiệu quả và độ chính xác của các thông tin báo cáo. Điều này càng trở nên quan trọng hơn đối với các hệ thống báo cáo tự động, vì hệ thống báo cáo này phải được tạo ra đúng thời điểm và đảm bảo tính chính xác về mặt dữ liệu. Do đó, việc xây dựng một hệ thống Data Warehouse có khả năng tích hợp và phân tích các nguồn dữ liệu phức tạp sẽ là một giải pháp tối ưu cho các công ty tài chính. Trong báo cáo này, em sẽ trình bày và đưa ra chi tiết về quá trình xây dựng hệ thống Data Warehouse và tích hợp các nguồn dữ liệu này để phục vụ cho hệ thống báo cáo tự động, cũng như nói về lợi ích và tác động to lớn của hệ thống này đối với hoạt động của công ty Valve.

Tại công ty Valve, việc thu thập và phân tích dữ liệu là một hoạt động cực kỳ quan trọng. Để đảm bảo được việc thu thập và phân tích này đạt được hiệu quả cao nhất, hệ thống data warehouse là một trong những yếu tố không thể thiếu. Data warehouse giúp tổng hợp, tích hợp và phân tích các nguồn dữ liệu khác nhau trên cùng một nền tảng, đồng thời cung cấp các báo cáo định kì và tự động cho người dùng. Để thực hiện mục tiêu đó, đề tài "Xây dựng hệ thống data warehouse tích hợp các nguồn dữ liệu phục vụ cho hệ thống báo cáo tự động" được đưa ra để tối ưu hoá quy trình thu thập, xử lý và phân tích dữ liệu, từ đó công ty có thể đưa ra được các quyết định một cách chính xác và nhanh chóng hơn.



*Hình 1: Sự bùng nổ của dữ liệu*

Trong quá trình xây dựng hệ thống báo cáo này, em đã ứng dụng một số công nghệ và phương pháp tiên tiến để tối ưu hóa hiệu suất cũng như độ chính xác của hệ thống. Ngoài ra, em cũng đưa ra một số đề xuất và phương án để đảm bảo tính bảo mật và bảo trì của hệ thống trong thời gian dài. Qua đó, em sẽ trình bày về quá trình xây dựng hệ thống Data Warehouse, các nguồn dữ liệu tích hợp vào hệ thống này, cũng như các công nghệ và phương pháp đã sử dụng để phân tích và cung cấp các báo cáo tự động cho người dùng. Do kiến thức còn nhiều hạn chế nên không thể tránh khỏi những thiếu sót trong quá trình thực hiện đề tài này. Vì vây, em rất mong nhận được những sự góp ý từ thầy cô để có thể hoàn thiện đề tài.

Em xin chân thành cảm ơn!

## Đối tượng và phạm vi

Trong quá trình thực hiện đề tài, em đã tập trung nghiên cứu, đánh giá và nhận xét các điểm hạn chế của kho dữ liệu của một công ty Valve, từ đó đề xuất các phương pháp giải quyết cho từng vấn đề cụ thể. Tập trung và xoay quanh hai chủ thể chính đó là:

* Hệ thống Data Warehouse (kho dữ liệu).
* Hệ thống báo cáo tự động tích hợp với các nguồn dữ liệu

Tuy em đã xác định được các đối tượng nghiên cứu, nhưng hai đối tượng này lại có quy mô vô cùng lớn. Để phù hợp với phạm vi và thời lượng của khoá luận tốt nghiệp, em sẽ tập trung nghiên cứu một nội dung có quy môn nhỏ hơn. Cụ thể, từ dữ liệu và yêu cầu của công ty Valve, em sẽ đề xuất và đưa ra một hệ thống cụ thể, chi tiết được bắt đầu từ việc xử lý dữ liệu nguồn và cuối cùng trích xuất báo cáo phân tích về hai nội dung này.

# Ý nghĩa khoa học và thực tiễn của đề tài

Nhận ra ý nghĩa to lớn, thực tiễn của đề tài. Trong xuyên suốt quá trình thực hiện chuyên đề thực tâp này, em luôn hướng đến một số mục tiêu và kết quả như sau:

1. Đánh giá tình trạng và giới hạn của hệ thống Data Warehouse hiện tại trong một công ty Valve.

2. Đề xuất một mô hình kiến trúc mới cho hệ thống kho dữ liệu để giải quyết các giới hạn và bất cập của các hệ thống cũ, cũng như thêm các thành phần mới như sau:

* Hạ tầng kho dữ liệu bao gồm: Một cơ sở dữ liệu được thiết kế đặc biệt để phục vụ cho kho dữ liệu, có kiến trúc cho phép mở rộng các công cụ trích xuất dữ liệu, công cụ phân tích và báo cáo.
* Mô hình kho dữ liệu: Nghiên cứu và áp dụng mô hình Data tiên tiến trên nền tảng các công cụ IBM.

3. Kế hoạch thiết kế thử nghiệm với hệ thống hệ thống báo cáo phân tích khách hàng, nhằm mục đích đánh giá hiệu suất xử lý của kho dữ liệu mới.

Lời cảm ơn

Đầu tiên, em xin chân thành cảm ơn các Thầy/Cô Học viện Ngân Hàng nói chung và khoa Hệ thống thông tin quản lý. Trong bốn năm qua, dưới sự chỉ dạy của các Thầy các Cô, em đã được tiếp nhận những kiến thức sâu rộng, bao quát và vô cùng quý giá. Những kiến thức và kinh nghiệm mà em đã học được từ các thầy cô là rất quý giá và sẽ giúp em phát triển trong tương lai.

Đặc biệt, em xin gửi một lời cảm ơn sâu sắc đến Thầy Đinh Trọng Hiếu -Giảng viên hướng dẫn của em. Em cảm ơn Thầy đã luôn tận tình giảng dạy và hướng dẫn em trong suốt thời gian thực hiện khoá luận tốt nghiệp, cũng như đã tạo điều kiện tốt nhất cho em để có thể hoàn thành Báo cáo khoá luận tốt nghiệp này. Em xin cảm ơn Thầy vì đã đặt kì vọng và luôn tin tưởng em!

# CHƯƠNG I: TỔNG QUAN VỀ HỆ THỐNG KHO DỮ LIỆU VÀ THIẾT KẾ MÔ HÌNH

## Tổng quan về Steam

**Steam** là một nền tảng phân phối nội dung số và cộng đồng trực tuyến dành cho các trò chơi điện tử. Nền tảng này được phát triển bởi công ty Valve Corporation và ra mắt lần đầu vào năm 2003. Steam đã trở thành một trong những nền tảng phân phối trò chơi hàng đầu trên thế giới, với hàng triệu người dùng trên toàn cầu.

Steam cho phép người dùng mua và tải xuống các trò chơi số từ các nhà phát triển độc lập và nhà phát hành lớn. Người dùng có thể duyệt qua cửa hàng trực tuyến của Steam, tìm kiếm các trò chơi theo thể loại, đánh giá và đề xuất, và chọn mua trò chơi mà họ quan tâm. Steam cũng cung cấp các dịch vụ bổ sung như cập nhật tự động, đồng bộ hóa lưu trữ trò chơi trên nhiều thiết bị và khả năng chơi trò chơi trực tuyến thông qua các máy chủ của Steam.

Một trong những yếu tố nổi bật của Steam là tính năng cộng đồng. Người dùng có thể kết bạn, tham gia vào nhóm, thảo luận với nhau, và chia sẻ thành tựu trong trò chơi. Steam cũng cung cấp tính năng trò chơi đa người chơi trực tuyến và cung cấp các công cụ cho các nhà phát triển để xây dựng và quản lý các cộng đồng trò chơi của riêng họ.

Ngoài ra, Steam cũng hỗ trợ một loạt các dịch vụ khác như Workshop - nơi người dùng có thể tạo, tải lên và chia sẻ nội dung tùy chỉnh cho các trò chơi, Steam Broadcasting - cho phép người dùng phát trực tiếp trò chơi của mình cho cộng đồng, và SteamVR - một nền tảng tương thích với thực tế ảo.

Steam đã trở thành một phần quan trọng của ngành công nghiệp trò chơi điện tử và đã đạt được sự phổ biến lớn. Nó không chỉ là một nền tảng phân phối trò chơi, mà còn là một cộng đồng năng động cho các game thủ trên toàn thế giới.

## Tổng quan kho dữ liệu

**Kho dữ liệu (Data Warehouse):** Kho dữ liệu là một tập hợp các dữ liệu được tổ chức và ổn định, thường được cập nhật thường xuyên để phục vụ cho mục đích phân tích và tạo báo cáo, giúp hỗ trợ trong quá trình ra quyết định.

Kho dữ liệu Steam là một thuật ngữ được sử dụng để chỉ tập hợp các thông tin liên quan đến trò chơi và người dùng trên nền tảng Steam của Valve Corporation. Steam là một nền tảng phân phối số cho các trò chơi điện tử trên máy tính, nơi người dùng có thể mua, tải xuống và chơi trò chơi trực tuyến. Kho dữ liệu Steam chứa các thông tin như danh sách trò chơi có sẵn trên Steam, thông tin về trò chơi, đánh giá của người dùng, cập nhật, thành tích và thống kê về số lượng người chơi trực tuyến. Ngoài ra, nó cũng chứa thông tin về tài khoản người dùng, danh sách bạn bè, thành tích cá nhân và hồ sơ trò chơi. Trong đó, Steam API cung cấp một cách để các nhà phát triển truy cập và sử dụng dữ liệu từ kho dữ liệu Steam để tạo ra các ứng dụng, công cụ và dịch vụ liên quan đến Steam. Việc sử dụng kho dữ liệu Steam có thể giúp cho việc phân tích thị trường trò chơi, tạo ra các công cụ quản lý cộng đồng, hoặc xây dựng các ứng dụng liên quan đến trò chơi trên Steam.

Kho Dữ liệu Thị trường trò chơi là một phần quan trọng của nền tảng Steam, trong đó các mô hình dữ liệu được kết nối với nhau và được điều khiển bởi các yêu cầu kinh doanh liên quan đến việc cung cấp dịch vụ quảng cáo. Trong ngành công nghiệp trò chơi, thông tin về thị trường và người dùng là vô cùng quan trọng. Kho Dữ liệu Thị trường trò chơi trong Steam ghi nhận và phân tích các dữ liệu về các trò chơi, người dùng, giao dịch và hành vi của cộng đồng. Các mô hình dữ liệu trong kho dữ liệu này được tổ chức và kết nối với nhau để tạo ra một cái nhìn toàn diện về thị trường trò chơi trên nền tảng Steam. Việc sử dụng dữ liệu trong kho Dữ liệu Thị trường trò chơi giúp Valve Corporation, công ty phát triển Steam, hiểu rõ hơn về xu hướng, sở thích và nhu cầu của người dùng. Các thông tin này được sử dụng để cung cấp dịch vụ quảng cáo hướng đúng đối tượng, từ đó tăng khả năng tiếp cận và tương tác của các nhà phát triển và nhà quảng cáo với người dùng Steam. Qua việc theo dõi và phân tích dữ liệu, kho Dữ liệu Thị trường trò chơi cung cấp thông tin về các yếu tố như số lượng người dùng, lượt tải xuống, đánh giá và phản hồi từ cộng đồng, doanh thu và mức độ phổ biến của các trò chơi. Điều này giúp Valve Corporation hiểu rõ thị trường và phân tích xu hướng để đưa ra quyết định kinh doanh thông minh, từ việc phát triển nội dung mới, xây dựng chiến lược quảng cáo đích đáng và nâng cấp nền tảng Steam. Quảng cáo được coi là trung tâm của yêu cầu kinh doanh trong kho Dữ liệu Thị trường trò chơi. Việc sử dụng dữ liệu từ kho Dữ liệu Thị trường giúp tăng cường khả năng quảng cáo chính xác và hiệu quả trên Steam. Các nhà phát triển và nhà quảng cáo có thể sử dụng thông tin từ kho Dữ liệu Thị trường để tìm hiểu và tiếp cận đúng đối tượng khách hàng, tối ưu hóa chiến dịch quảng cáo và đo lường hiệu quả.

Ví dụ, phân tích trải nghiệm khách hàng có thể được tiến hành quan điểm lấy khách hàng làm trung tâm vào nhận thức của họ về các khoản thanh toán sản phẩm và cấp độ dịch vụ như một phần của hoạt động kinh doanh tổng thể thực hiện với tổ chức phát hành trò chơi điện tử.

Thiết kế mô hình kho dữ liệu của Steam cũng hỗ trợ sự phân tách thông tin thành các mart dữ liệu / lược đồ sao cấu trúc, để giải quyết các chủ đề phân tích cụ thể. Điều này tạo điều kiện cho lưu trữ thông tin dựa trên chủ đề dễ tiếp cận hơn định dạng để truy vấn dữ liệu đồng thời cung cấp điểm truy cập để biết thông tin tổng hợp. Kho dữ liệu có thể triển khai trên nhiều Hệ thống quản lý cơ sở dữ liệu (DBMS) và không bị hạn chế trong môi trường phần cứng hoặc phần mềm cụ thể.

Các tính chất của mô hình kho dữ liệu:

* Hướng chủ đề (Subject – Oriented): Để có hiệu quả, dữ liệu trong kho dữ liệu phải được lưu trữ và tổ chức theo một định dạng tham chiếu và đo lường chung.
* Tính tích hợp (Intergrated): Để có hiệu quả, dữ liệu trong kho dữ liệu phải được tổ chức theo một định dạng tham chiếu và đo lường chung.
* Tính ổn định (Durable): Data Warehouse phải là một môi trường định hướng ra quyết định ổn định.
* Tính biến đổi theo thời gian (Time-oriented)

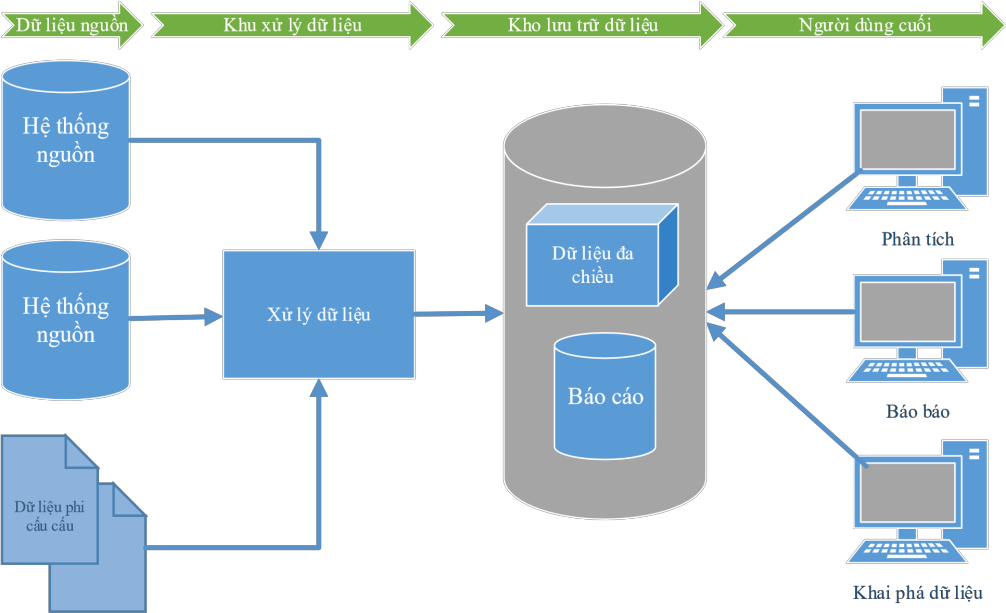
Lợi ích của Data Warehouse:

* Dễ dàng truy cập dữ liệu: Kho dữ liệu cho phép lưu trữ và quản lý dữ liệu một cách có tổ chức và đồng nhất, giúp người dùng dễ dàng truy cập và tìm kiếm dữ liệu theo nhu cầu.
* Phân tích và báo cáo hiệu quả: Kho dữ liệu cung cấp những công cụ phân tích và trực quan hóa dữ liệu mạnh mẽ để đánh giá hiệu quả của các hoạt động kinh doanh, từ đó giúp các nhà quản lý đưa ra quyết định đúng đắn và hiệu quả.
* Tăng tính nhất quán và đáng tin cậy của dữ liệu: Kho dữ liệu đảm bảo tính nhất quán và đáng tin cậy của dữ liệu bằng các quy trình và tiêu chuẩn chung để xử lý và kiểm tra dữ liệu.
* Tối ưu hóa hiệu suất hệ thống: Kho dữ liệu cho phép tối ưu hóa hiệu suất hệ thống bằng cách loại bỏ dữ liệu không cần thiết hoặc lặp lại, tăng tốc độ truy cập dữ liệu và giảm thời gian phản hồi của hệ thống.
* Tăng cường sự cạnh tranh: Kho dữ liệu có thể giúp doanh nghiệp nắm bắt các thông tin về thị trường, đối thủ cạnh tranh cũng như xu hướng ngành nghề, từ đó đưa ra các chiến lược kinh doanh hợp lí, đúng đắn và tăng cường khả năng cạnh tranh của doanh nghiệp trên thị trường.

## Kiến trúc chung của Data Warehouse

Mô hình kiến trúc cơ bản của Data Warehouse gồm 4 thành phần:

* Dữ liệu nguồn (Data Sources): Là các nguồn dữ liệu khác nhau được sử dụng để tạo nên kho dữ liệu, bao gồm các hệ thống OLTP, hệ thống CRM, các file Excel, các file log, và nhiều nguồn dữ liệu khác.
* Các phần mềm ETL (Extract, Transform, Load): Là các phần mềm đóng vai trò trích và xuất dữ liệu từ những nguồn dữ liệu khác nhau và chuyển đổi dữ liệu để đáp ứng yêu cầu của kho dữ liệu cũng như tải dữ liệu vào kho dữ liệu.
* Kho dữ liệu (Data Warehouse): Là nơi dữ liệu được lưu trữ sau khi được trích xuất, chuyển đổi và tải vào kho. Kho dữ liệu được tổ chức theo cấu trúc phù hợp với các yêu cầu của tổ chức và được sử dụng để phân tích dữ liệu.
* Các công cụ phân tích (Analytics Tools): Là các công cụ phân tích được sử dụng để truy xuất dữ liệu từ Data Warehouse, phân tích dữ liệu, và đưa ra các insights hữu ích để hỗ trợ quyết định. Những công cụ này gồm các bảng điều khiển, các câu truy vấn SQL, các báo cáo và các phần mềm phân tích dữ liệu.

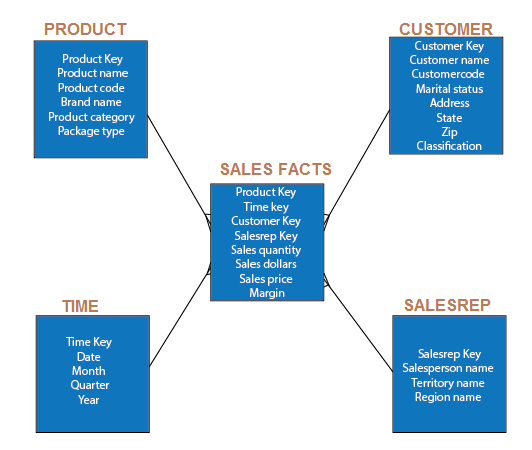


*Hình 1.1: Mô tả kiến trúc kho dữ liệu*

## Mô hình logic của kho dữ liệu

#### Sơ đồ hình sao (Star schema):

Là một tập các bảng chiều khác nhau kết nối với một bảng sự kiện ở trung tâm.



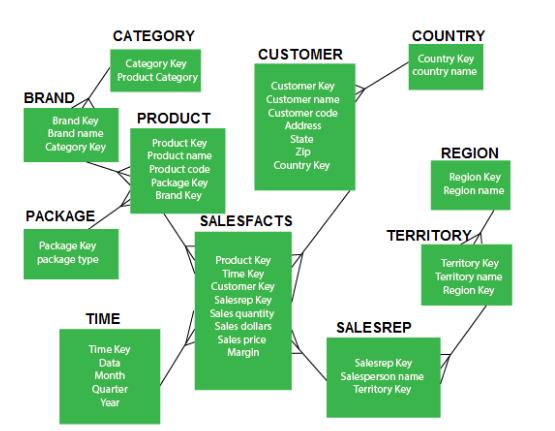
*Hình 1.2: Mô tả sơ đồ hình sao*

***Ưu điểm:*** Mô hình logic được mô tả đơn giản, dễ hiểu. Mô hình ngôi sao có nhiều ưu điểm. Mỗi bảng dimension có liên quan đến bảng fact bằng một quan hệ và không có bảng dimension nào khác để miêu tả dữ liệu. Dữ liệu được đơn giản hóa để truy vấn và giúp cho thời gian thực thi được nhanh hơn.

***Nhược điểm:*** Có tính dư thừa, mỗi một bảng Dimension lưu trữ thông tin tách rời chính vì đó dữ liệu không được chuẩn hóa. => Tiêu tốn nhiều tài nguyên bộ nhớ và có tỉ lệ rủi ro về tính toàn vẹn dữ liệu.

#### Sơ đồ bông tuyết (Snowflake schema)

Sơ đồ bông tuyết là một loại mô hình dữ liệu đa chiều được sử dụng cho kho dữ liệu. Nó được gọi là sơ đồ bông tuyết vì biểu đồ của nó giống như một bông tuyết với nhiều nhánh phân nhánh từ bảng dữ kiện ở trung tâm. Sơ đồ bông tuyết là sự mở rộng của sơ đồ sao trong đó các bảng dimension được chuẩn hóa để loại bỏ dữ liệu dư thừa.



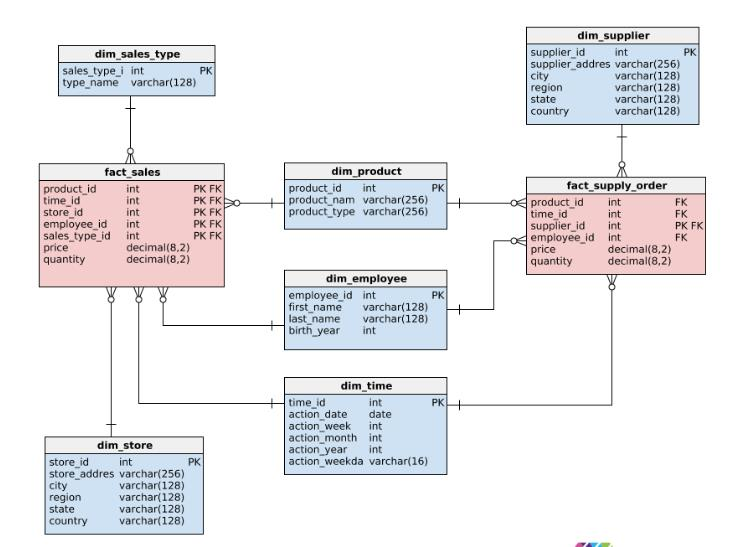
*Hình 1.3 Mô tả sơ đồ bông tuyết*

***Ưu điểm:*** Dữ liệu của bảng Dimension được chuẩn hóa, được cấu trúc rõ ràng và chia thành nhiều chiều: chính và phụ. Số chiều được phân cấp thể hiện định dạng chuẩn của bảng Dimension.

***Nhược điểm:*** Làm giảm hiệu suất truy vấn.

#### Sơ đồ chòm sao sự kiện (Fact constellations schema):

Sơ đồ chòm sao sự kiện là một loại sơ đồ dữ liệu đa chiều và là một sự kết hợp của nhiều sơ đồ sao có các bảng dimension chung được sử dụng cho kho dữ liệu.



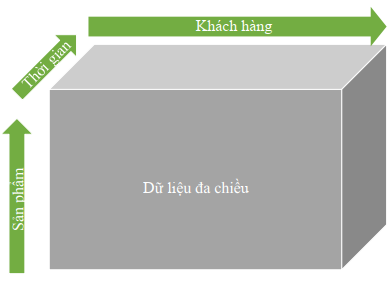
*Hình 1.4 Mô tả sơ đồ chòm sao*

***Ưu điểm:*** Giúp giảm kích thước của CSDL đặc biệt là khi sử dụng các bảng Dimension có nhiều giá trị.

***Nhược điểm:*** Phức tạp và rủi ro hơn nhiều so với mô hình ngôi sao.

## Mô hình dữ liệu đa chiều

Mô hình dữ liệu đa chiều là một loại mô hình dữ liệu được sử dụng cho các mô hình Data Warehouse và các hệ thống OLAP. Dữ liệu được thể hiện dưới dạng khối (cube) với nhiều chiều, mỗi chiều mô tả một đặc trưng của dữ liệu. Mỗi chiều có thể có nhiều mức trừu tượng được xác định bằng phân cấp khái niệm. Mô hình này cho phép người dùng xem dữ liệu một cách trực quan với nhiều mức độ khác nhau.



*Hình 1.5 Mô hình dữ liệu đa chiều*

## Tổng quan về dữ liệu

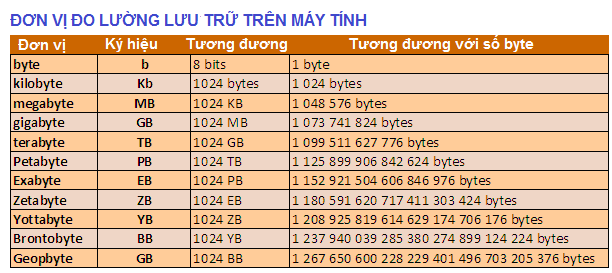
### Dữ liệu là gì?

Dữ liệu là tập hợp thông tin thu thập được từ các hoạt động quan sát, đo lường, nghiên cứu, khám phá hoặc phân tích. Dữ liệu bao gồm sự thật, con số, số liệu, tên và mô tả của sự vật. Nó có thể được tổ chức dưới dạng sơ đồ, biểu đồ hoặc bảng. Nói chung, dữ liệu là tập hợp các ký tự được thu thập cho một mục đích nhất định. Nếu không có ngữ cảnh cụ thể, dữ liệu không có giá trị đối với con người và máy tính.

Trong lĩnh vực tính toán, dữ liệu được xem là thông tin đã được chuyển đổi sang định dạng hữu ích để truyền hoặc xử lý được trên máy tính. Dữ liệu được chuyển đổi sang dạng kỹ thuật số nhị phân để sử dụng với các thiết bị và phương tiện truyền thông khác. Nếu dữ liệu được lưu trữ dưới dạng kỹ thuật số cơ bản thì được gọi là "dữ liệu thô".

* 1. Cách thức lưu trữ:

Máy tính dùng mẫu số gồm hai số 0 và 1 để biểu diễn dữ liệu, bao gồm hình ảnh, video, âm thanh và văn bản ở dạng nhị phân. Bit là đơn vị nhỏ nhất của dữ liệu, mỗi bit chỉ chứa một giá trị duy nhất. Tám bit tạo thành một byte. Kilobyte, Megabyte và Gigabyte là các đơn vị phổ biến để đo lượng bộ nhớ lưu trữ của máy tính. Trong khi đó, với sự phát triển của công nghệ thông tin, khối lượng dữ liệu được thu thập và lưu trữ tăng lên đáng kể, các đơn vị đo lường dữ liệu cũng được nâng cấp tương ứng. Ví dụ, đơn vị đo lường dữ liệu lớn nhất hiện tại là "Geopbyte" viết tắt là GB, vượt xa kích thước của Gigabyte.



*Hình 1.6. Các đơn vị dữ liệu sắp xếp từ nhỏ tới lớn*

### Kiểu dữ liệu:

Trong vòng 10 năm qua, sự phát triển của web và thiết bị điện tử thông minh (như điện thoại thông minh, TV, đồng hồ thông minh...) đã gây ra sự "phát triển vượt bậc" của dữ liệu kỹ thuật số, và được coi là "dữ liệu bùng nổ". Dữ liệu hiện nay bao gồm các ghi chép của các hoạt động trực tuyến, thông tin dưới dạng âm thanh, văn bản, hình ảnh và video. Phần lớn dữ liệu hiện nay không có cấu trúc cụ thể.

## Tổng quan về mô hình Datavault

Mô hình Data Vault là một mô hình thiết kế cơ sở dữ liệu linh hoạt và khả năng mở rộng, được sử dụng rộng rãi trong việc xây dựng các hệ thống kho dữ liệu (data warehouse) và phân tích dữ liệu. Mô hình này được phát triển bởi Dan Linstedt vào năm 2000 và còn được cải tiến và sử dụng đến ngày nay.

Mô hình Data Vault có cấu trúc phân tầng với 3 lớp chính bao gồm:

* Lớp nạp dữ liệu (loading layer): lớp này được sử dụng để lưu trữ dữ liệu nguyên gốc được thu thập từ các nguồn khác nhau, được gọi là dữ liệu định dạng (raw data). Dữ liệu định dạng này được chuyển đổi sang dạng tiêu chuẩn (standardized data) và được lưu trữ trong lớp tiêu chuẩn hóa (staging area).
* Lớp tiêu chuẩn hóa (staging layer): lớp này được sử dụng để tiêu chuẩn hóa dữ liệu định dạng được lưu trữ trong lớp nạp dữ liệu và chuyển đổi chúng thành dữ liệu tiêu chuẩn.
* Lớp phân tích (presentation layer): lớp này được sử dụng để lưu trữ dữ liệu đã được tiêu chuẩn hóa và được tổ chức thành các bảng dữ liệu (dimension tables và fact tables) để phục vụ cho việc phân tích và báo cáo.

Mô hình Data Vault sử dụng các đối tượng cơ bản bao gồm: Satellites (vệ tinh), Hubs (trung tâm) và Links (liên kết). Satellites chứa thông tin chi tiết về một thuộc tính của một đối tượng, Hubs chứa thông tin chính xác về đối tượng, và Links được sử dụng để liên kết các đối tượng với nhau.

Mô hình Data Vault có nhiều ưu điểm như khả năng linh hoạt, tự động, khả năng mở rộng dễ dàng và khả năng quản lý dữ liệu lớn. Tuy nhiên, việc thiết kế và triển khai mô hình này cần có kiến thức và kinh nghiệm chuyên sâu để đảm bảo tính ổn định và hiệu quả của hệ thống kho dữ liệu.

## Thu thập và phân tích các yêu cầu về dữ liệu

Trong quá trình này, cần phải đưa ra các câu hỏi để hiểu rõ nhu cầu của khách hàng và đội ngũ phát triển về dữ liệu, bao gồm:

* Loại dữ liệu nào được sử dụng trong hệ thống?

Để xác định loại dữ liệu được sử dụng trong hệ thống, cần tìm hiểu các ứng dụng, hệ thống và nguồn dữ liệu khác được sử dụng bởi khách hàng hoặc đội ngũ phát triển như: dữ liệu về khách hàng (thông tin cá nhân, lịch sử mua hàng, phản hồi khách hàng, v.v.) hoặc dữ liệu về sản phẩm (mã sản phẩm, mô tả sản phẩm, giá, v.v.)

* Dữ liệu được lưu trữ ở đâu?

Cần tìm hiểu về các nguồn dữ liệu được lưu trữ trong hệ thống khách hàng hoặc đội ngũ phát triển, bao gồm cả cơ sở dữ liệu, các tệp dữ liệu, ứng dụng và các nguồn dữ liệu bên ngoài như: các hệ thống quản lý khách hàng (CRM), hệ thống quản lý sản phẩm (PIM), hệ thống bán hàng (POS), hệ thống quảng cáo và tiếp thị, các bảng tính Excel và các hệ thống khác.

* Các thông tin nào cần được tích hợp vào Data Warehouse?

Các thông tin cần được tích hợp vào Data Warehouse phụ thuộc vào mục đích sử dụng của hệ thống, các chỉ tiêu cần đạt được, và các yêu cầu phân tích dữ liệu của khách hàng. Tùy vào mục đích sử dụng, có thể tích hợp các thông tin về khách hàng, sản phẩm, bán hàng, tài chính và các nguồn dữ liệu khác.

* Những chỉ tiêu nào cần đạt được thông qua việc xử lý dữ liệu?

Các chỉ tiêu này còn phụ thuộc vào mục đích sử dụng của hệ thống, nhưng thường bao gồm các chỉ tiêu liên quan đến doanh số bán hàng, lợi nhuận, chi phí, sản phẩm, khách hàng, ngoài ra còn các chỉ tiêu tài chính khác.

* Những câu hỏi hay những tình huống phân tích nào cần được hỗ trợ bởi hệ thống Data Warehouse?

Các câu hỏi và tình huống phân tích có thể khác nhau tùy thuộc vào mục đích sử dụng của hệ thống, nhưng thường bao gồm các câu hỏi về xu hướng, dự đoán, biến động, so sánh, phân tích đối tượng, phân tích đa chiều và phân tích thống kê.

Sau khi thu thập được các yêu cầu về dữ liệu từ khách hàng và đội ngũ phát triển, ta sẽ tiến hành phân tích và đánh giá tính khả thi của chúng. Việc này sẽ giúp đưa ra quyết định về việc lựa chọn nguồn dữ liệu và phát triển hệ thống Data Warehouse phù hợp nhất.

Đánh giá tính khả thi các yêu cầu về dữ liệu được phân tích dựa trên các yếu tố như khả năng thu thập, tích hợp và xử lý dữ liệu từ các nguồn dữ liệu khác nhau, để đáp ứng các chiến lược kinh doanh của tổ chức, và chi phí và thời gian thực hiện. Đánh giá tính khả thi có thể bao gồm cả khả năng quản lý và bảo mật dữ liệu, khả năng thực hiện các phân tích và báo cáo đáp ứng các yêu cầu của khách hàng và tổ chức, cũng như khả năng phát triển và nâng cấp cho hệ thống Data Warehouse theo thời gian. Quá trình đánh giá tính khả thi sẽ giúp xác định được các yêu cầu có khả năng thực hiện và đóng góp tích cực cho tổ chức, từ đó tối ưu hóa việc đầu tư và tiết kiệm tài nguyên.

## Phân tích quy trình nghiệp vụ của nền tảng Steam

### Hoa hồng

Steam kiếm tiền từ mọi trò chơi được bán thông qua nền tảng của nó. Công ty tính phí cắt giảm 30% cho mỗi giao dịch. Trong khi người dùng trả tiền cho trò chơi, nhà phát triển chịu số tiền đã trả cho Steam. Tuy nhiên, Steam cũng cung cấp mức giá tốt hơn cho các nhà phát triển khác. Ví dụ: họ chỉ tính phí 25% nếu họ bán các trò chơi trị giá từ 10 triệu đến 50 triệu đô la. Đối với các nhà phát triển kiếm được trên số tiền đó, hoa hồng là 20%.

### Bán phần cứng

Steam cũng mạo hiểm tham gia vào ngành công nghiệp phần cứng thông qua các sản phẩm của mình. Ban đầu, công ty tin tưởng vào uy quyền kỹ thuật số. Tuy nhiên, nó cũng nhận ra sự cần thiết của các sản phẩm khác. Steam đã phát triển và tiếp thị Steam Link và bộ điều khiển trò chơi. Những công cụ này cho phép công ty kiếm tiền từ việc bán phần cứng của mình. Công ty cũng đã ra mắt Steam Deck vào năm 2022, mang lại nhiều thu nhập hơn từ những sản phẩm này.

### Phí nhà phát triển

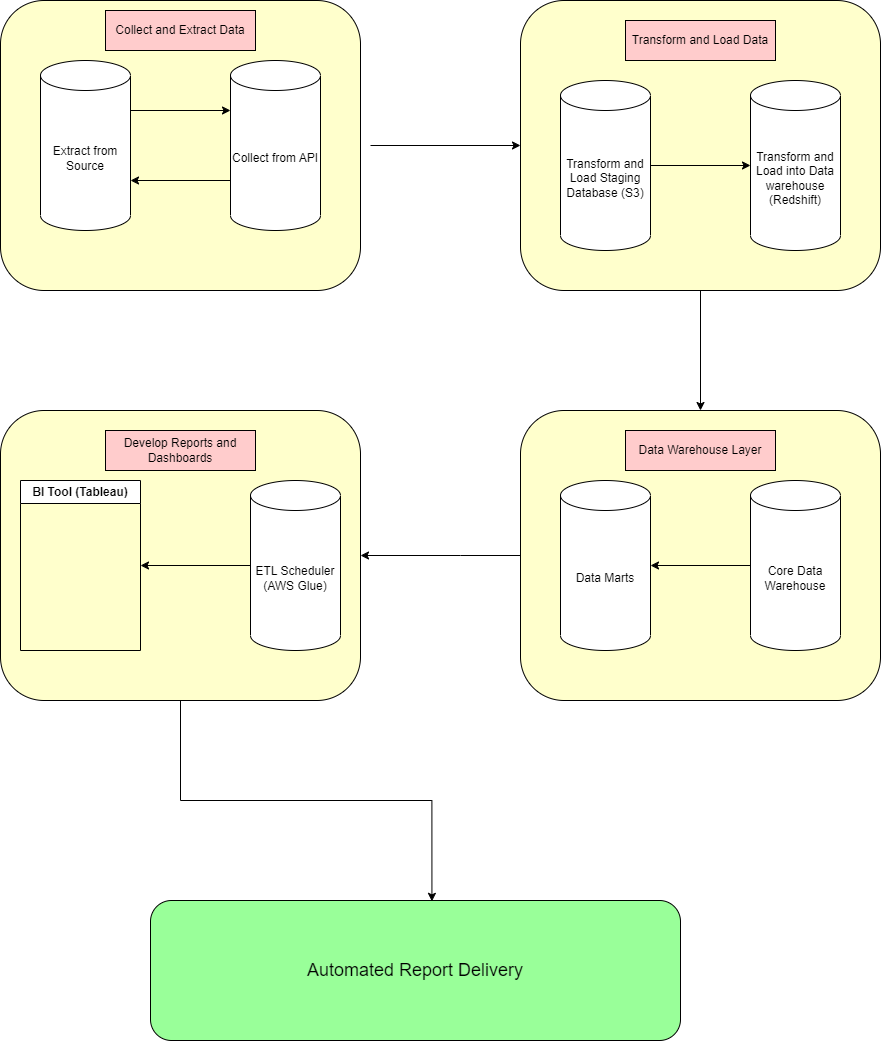
Steam cung cấp cho các nhà phát triển nhiều công cụ và dịch vụ khác nhau. Những công cụ này thuộc thương hiệu Steamworks. Sử dụng chúng, các nhà phát triển có thể xây dựng và phân phối trò chơi thông qua nền tảng Steam. Điều này cho phép các nhà xuất bản bán các gói và DLC. Đổi lại, Steam tính phí nhà phát triển cho họ. Tuy nhiên, những công cụ này có sẵn miễn phí cho các nhà phát triển hiện có.

### Marketing

Steam Greelight là một nền tảng cho phép người dùng Steam đang hoạt động bỏ phiếu cho các tựa game độc lập mà họ muốn xem được giới thiệu trên trang web của họ. Với một khoản phí gửi nhỏ, bất kỳ nhà phát triển độc lập nào cũng có thể tải trò chơi của họ lên dịch vụ. Sau khi gửi, hàng nghìn người dùng sẽ xem và có thể bỏ phiếu hoặc nhận xét về trò chơi. Video trò chơi đại diện cho sản phẩm cuối cùng hầu hết các trò chơi được đưa lên Greenlight đều còn vài tháng hoặc thậm chí một năm nữa mới được phát hành. Để loại bỏ các nội dung gửi spam và trò đùa ra khỏi hệ thống, có một khoản phí gửi một lần sẽ cho phép tài khoản steam của bạn gửi các trò chơi lên Steam Greenlight. Khi bạn đã trả phí, bạn có thể gửi bao nhiêu trò chơi mà bạn muốn. Valve khuyến khích các nhà phát triển cập nhật sự hiện diện của họ trong steam thường xuyên nếu họ muốn để giữ chân người hâm mộ và khách hàng tiềm năng tham gia vào quá trình phát triển

## Xây dựng hệ thống Data Warehouse

Sau khi đã đánh giá tính khả thi của các yêu cầu về dữ liệu, ta có thể tiến hành xây dựng hệ thống Data Warehouse. Các bước chính trong quá trình này bao gồm:



*Hình 1.7. Các bước xây dựng DWH và báo cáo tự động*

Thu thập nguồn dữ liệu: Thu thập và chuẩn hóa dữ liệu từ các nguồn khác nhau, bao gồm cả dữ liệu cấu trúc và phi cấu trúc. Sử dụng các công cụ AWS như AWS Glue và AWS Data Pipeline để thực hiện ETL (Extract, Transform, Load) quá trình này.

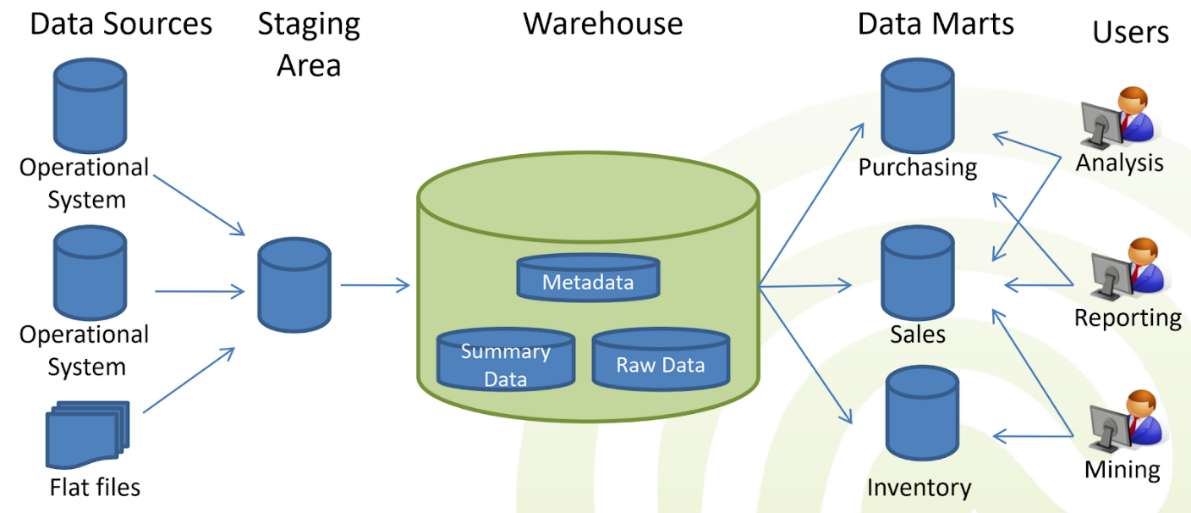
Lưu trữ dữ liệu: Lưu trữ dữ liệu đã được chuẩn hóa vào Redshift data warehouse. Redshift là một data warehouse có khả năng xử lý dữ liệu lớn, được cung cấp bởi AWS.

Xây dựng Data Warehouse: Thiết kế các mô hình dữ liệu cho hệ thống data warehouse, bao gồm cả các bảng kích thước (dimension tables) và các bảng sự kiện (fact tables). Sử dụng mô hình dữ liệu Data Vault để đảm bảo tính nhất quán và dễ dàng mở rộng.

Sử dụng các công cụ ETL để cập nhật dữ liệu mới vào Redshift và đồng bộ hóa với các bảng kích thước và các bảng sự kiện. Các công cụ ETL có thể được tích hợp trong AWS Glue hoặc được tạo bởi các công cụ của bên thứ ba.

Xây dựng các Dashboard và Reports: Sử dụng các công cụ BI như Tableau để truy vấn và trực quan hóa dữ liệu trong Redshift data warehouse. Các bảng kích thước và các bảng sự kiện có thể được kết nối với Tableau để tạo ra các báo cáo và dashboard.

Đồng bộ dữ liệu: Cập nhật và đồng bộ hóa các báo cáo và dashboard với dữ liệu mới nhất trong Redshift, sử dụng các công cụ ETL và lên lịch tự động để thực hiện quá trình này.



*Hình 1.8. Quy trình phân tích dữ liệu*

Khi đã xác định được các yêu cầu về dữ liệu cần tích hợp và đạt được các chỉ tiêu phân tích, ta sẽ tiến hành thiết kế cơ sở dữ liệu lớp HUB để tích hợp các nguồn dữ liệu vào hệ thống Data Warehouse.

## Thiết kế cơ sở dữ liệu lớp HUB

Lớp HUB là một phần quan trọng trong mô hình Datavault và là nơi chứa các thông tin đầu vào từ các nguồn dữ liệu khác nhau. Thiết kế cơ sở dữ liệu lớp HUB được đặc trưng bởi các thuộc tính và quy tắc thiết kế sau đây:

### Thuộc tính của lớp HUB:

* HUB Key: đây là trường dữ liệu duy nhất trong lớp HUB. Nó được sử dụng để liên kết với các lớp khác trong mô hình Datavault.
* Business Key: đây là thuộc tính thể hiện giá trị thực tế trong các hệ thống nguồn. Ví dụ: mã khách hàng, mã sản phẩm…
* Load Date: đây là thuộc tính cho biết ngày giờ dữ liệu được tải vào lớp HUB.
* Source Key: đây là thuộc tính để theo dõi thông tin nguồn dữ liệu. Nó cho phép Ta biết nơi dữ liệu được lấy từ.

### Quy tắc thiết kế lớp HUB:

* Lớp HUB được thiết kế để giữ nguyên toàn bộ thông tin đầu vào từ các nguồn dữ liệu.
* Mỗi lớp HUB chỉ chứa một loại dữ liệu.
* Lớp HUB không bị phụ thuộc vào bất kỳ lớp nào khác trong mô hình Datavault.

### Xử lý dữ liệu của lớp HUB:

* Dữ liệu đầu vào được tải vào lớp HUB mà không cần xử lý.
* Các bản ghi mới được chèn vào lớp HUB, còn các bản ghi đã tồn tại được bỏ qua hoặc được ghi đè.
* Lớp HUB được sử dụng để đối chiếu dữ liệu từ các nguồn khác nhau. Dữ liệu được cập nhật trong lớp HUB và các lớp khác sẽ được cập nhật sau đó.

## Thiết kế cơ sở dữ liệu lớp LINK

Lớp LINK là một phần quan trọng trong thiết kế cơ sở dữ liệu Data Vault. Nó kết nối giữa các lớp HUB và cung cấp thông tin về mối quan hệ giữa chúng. Đây là nơi chứa các khóa của các lớp HUB và các thuộc tính của mối quan hệ.

### Quy tắc thiết kế cho lớp LINK gồm:

* Lớp LINK kết nối giữa ít nhất hai lớp HUB.
* Mỗi lớp LINK có thể kết nối tới nhiều lớp HUB khác nhau.
* Lớp LINK không có dữ liệu bổ sung ngoài khóa của các lớp HUB.
* Mỗi lớp LINK phải có một khóa tổng hợp duy nhất, bao gồm tất cả các khóa của các lớp HUB liên quan...

### Cách thiết kế lớp LINK thường bao gồm các bước sau:

* Xác định các mối quan hệ giữa các lớp HUB.
* Xác định khóa chính của mỗi lớp HUB.
* Xác định các thuộc tính cần thiết cho mỗi mối quan hệ.
* Tạo các lớp LINK để kết nối giữa các lớp HUB.
* Mỗi lớp LINK phải có một khóa tổng hợp duy nhất, bao gồm tất cả các khóa của các lớp HUB liên quan.

### Cách xử lý dữ liệu trong lớp LINK bao gồm:

* Dữ liệu được chia thành các bản ghi tương ứng với các khóa tổng hợp của mỗi lớp LINK.
* Các bản ghi được thêm vào hoặc cập nhật trong lớp LINK bằng cách sử dụng các khóa của các lớp HUB liên quan.
* Dữ liệu trong lớp LINK được duy trì một cách độc lập với các lớp HUB và có thể được sử dụng để tạo ra các báo cáo và phân tích liên quan đến mối quan hệ giữa các lớp HUB.

## Thiết kế cơ sở dữ liệu lớp SATELLITE

Lớp Satellite là một trong ba lớp trong mô hình Data Vault, chịu trách nhiệm lưu trữ các thông tin chi tiết và tương tác với các khóa ngoại của lớp Hub hoặc Link. Lớp Satellite là nơi lưu trữ dữ liệu cụ thể và đầy đủ về các sự kiện trong quá trình kinh doanh. Dữ liệu lưu trữ trong lớp Satellite thường không thể tái sử dụng và không phải là thông tin quan trọng như lớp Hub và Link.

Các thuộc tính của lớp Satellite bao gồm các thuộc tính cơ bản và các thuộc tính chuyên dụng. Các thuộc tính cơ bản bao gồm ngày hiệu lực, ngày hết hạn, ngày khởi tạo, người tạo, người cập nhật, mã số phiên bản. Các thuộc tính chuyên dụng được thiết kế dựa trên nghiệp vụ kinh doanh của tổ chức, chẳng hạn như địa điểm, giá cả, số lượng và số tiền giao dịch.

### Quy tắc thiết kế lớp SATELLITE có thể được mô tả như sau:

* Lớp Satellite được thiết kế để lưu trữ thông tin chi tiết về các sự kiện, hoạt động, hoặc hành động diễn ra trong thực tế.
* Các bảng trong lớp Satellite thường được thiết kế để lưu trữ các thông tin chi tiết như thời gian, địa điểm, tình trạng, hành động, và kết quả của các sự kiện.
* Các thuộc tính của lớp Satellite thường bao gồm khoá chính (PK), khoá ngoại (FK) đến lớp HUB và lớp LINK, và các thuộc tính mô tả thông tin chi tiết của sự kiện.
* Các quan hệ giữa các bảng trong lớp Satellite có thể là one-to-many hoặc many-to-many, phụ thuộc vào kiểu dữ liệu của sự kiện được lưu trữ.

### Các quy tắc thiết kế lớp SATELLITE:

* Mỗi bảng trong lớp Satellite nên được thiết kế để lưu trữ thông tin chi tiết của một sự kiện duy nhất.
* Khoá chính của mỗi bảng trong lớp Satellite nên được thiết kế để đảm bảo tính duy nhất và cập nhật dễ dàng.
* Các bảng trong lớp Satellite nên được thiết kế để lưu trữ các thông tin chi tiết về các sự kiện mà không bị ràng buộc bởi các quan hệ giữa các bảng.
* Các thuộc tính trong lớp Satellite nên được thiết kế để đảm bảo tính linh hoạt và dễ dàng thay đổi.

### Cách xử lý dữ liệu trong lớp SATELLITE bao gồm:

* Thu thập dữ liệu: Các thông tin về vệ tinh được thu thập từ các nguồn khác nhau, bao gồm các trang web, tài liệu về vệ tinh, hoặc từ các hệ thống khác.
* Kiểm tra và xử lý dữ liệu: Sau khi thu thập được dữ liệu, chúng ta cần kiểm tra và xử lý dữ liệu để đảm bảo tính chính xác và đầy đủ của dữ liệu.
* Lưu trữ dữ liệu: Các thông tin về vệ tinh được lưu trữ trong cơ sở dữ liệu, bao gồm các bảng và mối quan hệ giữa chúng.
* Truy xuất dữ liệu: Các thông tin về vệ tinh được truy xuất từ cơ sở dữ liệu bằng cách sử dụng các truy vấn SQL. Các truy vấn này cho phép lấy thông tin về các vệ tinh theo các tiêu chí khác nhau, ví dụ như theo tên, theo loại quỹ đạo hoặc theo ngày phóng.

Ngoài ra, để đảm bảo tính toàn vẹn và hiệu quả trong việc lưu trữ và xử lý dữ liệu, có thể sử dụng các công cụ quản lý cơ sở dữ liệu như MySQL, Oracle hay MongoDB.

## Các quy trình ETL

Quy trình ETL (Extract-Transform-Load) là một quy trình quan trọng trong xây dựng hệ thống DataVault. Quy trình này bao gồm các bước trích xuất (Extract), chuyển đổi (Transform) và tải dữ liệu (Load) từ các nguồn dữ liệu khác nhau vào DataVault.

Bước trích xuất (Extract) là quá trình thu thập dữ liệu từ các nguồn khác nhau như hệ thống giao dịch, hệ thống CRM hay các file dữ liệu khác. Các công cụ trích xuất dữ liệu thường được sử dụng như là các ứng dụng ETL hoặc các kịch bản Python.

Bước chuyển đổi (Transform) là quá trình biến đổi dữ liệu trước khi nó được tải vào hệ thống DataVault. Trong bước này, các định dạng dữ liệu khác nhau được chuyển đổi và các bước tiền xử lý được thực hiện như là loại bỏ các giá trị trùng lặp hay các giá trị thiếu.

Bước tải dữ liệu (Load) là quá trình tải dữ liệu đã được trích xuất và chuyển đổi vào hệ thống DataVault. Trong quá trình này, các dữ liệu được lưu trữ trong các lớp HUB, LINK và SATELLITE. Các công cụ tải dữ liệu thường được sử dụng như là các ứng dụng ETL hoặc các kịch bản Python.

Các quy trình ETL trong hệ thống DataVault cũng thường bao gồm các công cụ kiểm tra chất lượng dữ liệu (Data Quality Checks) để đảm bảo rằng dữ liệu được tải vào hệ thống là chính xác và đầy đủ. Ngoài ra, các công cụ này cũng được sử dụng để xác định các lỗi cấu trúc hoặc các giá trị không hợp lệ trong dữ liệu, giúp người dùng có thể sửa chữa và bổ sung dữ liệu một cách hiệu quả.

Tóm lại, quy trình ETL là một phần quan trọng trong xây dựng hệ thống DataVault, giúp trích xuất, chuyển đổi và tải dữ liệu vào hệ thống một cách nhanh chóng và chính xác.

## Phân tích dữ liệu

Phân tích dữ liệu là quá trình quan trọng trong việc thiết kế một hệ thống Data Vault. Nó đặc biệt quan trọng trong việc xác định các lớp HUB, LINK và SATELLITE phù hợp với mô hình datavault.

Đầu tiên, để phân tích dữ liệu, các nhà thiết kế hệ thống sẽ phải hiểu rõ yêu cầu và mục tiêu của hệ thống. Điều này đòi hỏi các nhà thiết kế phải tìm hiểu và nghiên cứu các nguồn dữ liệu đang được sử dụng trong tổ chức, bao gồm cả dữ liệu tổng hợp và chi tiết.

Sau đó, các nhà thiết kế sẽ phân tích cấu trúc dữ liệu và xác định các yếu tố quan trọng như các trường dữ liệu, các quan hệ giữa các bảng, các phương pháp thống kê và xử lý dữ liệu.

Dựa trên phân tích của họ, các nhà thiết kế sẽ xác định các lớp HUB, LINK và SATELLITE phù hợp với mô hình datavault. Các lớp này được xác định bằng cách phân tích quan hệ giữa các bảng và cách dữ liệu được sử dụng trong tổ chức.

Khi xác định các lớp HUB, LINK và SATELLITE, các nhà thiết kế sẽ tiến hành tạo các bản thiết kế chi tiết cho từng lớp, bao gồm cả các thuộc tính và quy tắc thiết kế.

Quá trình phân tích dữ liệu là bước quan trọng để đảm bảo rằng hệ thống datavault được thiết kế một cách chính xác và đáp ứng được các yêu cầu và mục tiêu của tổ chức.

## Thiết kế database

Mô hình Data Vault là một phương pháp thiết kế cơ sở dữ liệu được sử dụng để lưu trữ các dữ liệu lịch sử, phù hợp với các hệ thống phân tán và có quy mô lớn. Mô hình này tập trung vào việc đảm bảo tính toàn vẹn dữ liệu và dễ dàng thực hiện các truy vấn phức tạp trong các hệ thống dữ liệu phân tích (Data Warehouse) và các mô hình dữ liệu phức tạp.

Mô hình Data Vault bao gồm ba loại bảng: Hub, Link và Satellite.

* Hub: Hub là bảng chứa thông tin về các thực thể trong hệ thống. Hub chỉ chứa một cột duy nhất là khóa tự nhiên (Natural Key). Mỗi khóa tự nhiên đại diện cho một thực thể trong hệ thống. Ví dụ, trong hệ thống bán lẻ, Hub có thể đại diện cho khách hàng, sản phẩm hoặc nhà cung cấp.
* Link: Link là bảng chứa thông tin về các quan hệ giữa các thực thể trong hệ thống. Mỗi bảng Link có thể chứa nhiều cột khóa tự nhiên của các bảng Hub khác nhau. Ví dụ, bảng Link trong hệ thống bán lẻ có thể chứa khóa tự nhiên của khách hàng và sản phẩm để đại diện cho việc mua hàng của khách hàng.
* Satellite: Satellite là bảng chứa các thuộc tính chi tiết về các thực thể trong hệ thống. Mỗi bảng Satellite có thể chứa nhiều cột thuộc tính chi tiết của một thực thể trong hệ thống. Mỗi bảng Satellite cũng chứa các cột biểu thị cho ngày bắt đầu và ngày kết thúc của sự kiện, cho phép theo dõi lịch sử của các thực thể.

Các bảng trong mô hình Data Vault được thiết kế để đảm bảo tính toàn vẹn dữ liệu, sử dụng khóa tự nhiên để đại diện cho các thực thể và quan hệ giữa các thực thể. Khóa tự nhiên cũng được sử dụng để thiết lập các khóa ngoại giữa các bảng Hub và các bảng Link.

Một số quy tắc thiết kế cơ sở dữ liệu trong mô hình Data Vault bao gồm:

* Sử dụng khóa tự nhiên làm khóa chính cho các bảng Hub và Satellite.
* Sử dụng khóa tự nhiên của các bảng Hub trong các bảng Link.

Các quan hệ giữa các bảng trong mô hình Data Vault được thiết kế dựa trên nguyên tắc quan hệ giữa các thực thể. Các bảng Hub đại diện cho các thực thể, các bảng Link đại diện cho các quan hệ giữa các thực thể và các bảng Satellite chứa các thuộc tính chi tiết của các thực thể.

Quan hệ giữa các bảng được thiết lập thông qua các khóa ngoại. Một khóa ngoại trong Data Vault sẽ luôn bao gồm ít nhất một cột khóa tự nhiên của một bảng Hub hoặc Link và một cột thời gian (từ ngày đến ngày) của bảng Satellite.

Ví dụ, để thiết kế một cơ sở dữ liệu Data Vault cho hệ thống bán lẻ, ta có thể thiết kế các bảng như sau:

* Bảng Hub Customer: chứa các thông tin về khách hàng bao gồm khóa tự nhiên CustomerID và các thuộc tính khác như tên khách hàng, địa chỉ, điện thoại, email, v.v.
* Bảng Hub Product: chứa các thông tin về sản phẩm bao gồm khóa tự nhiên ProductID và các thuộc tính khác như tên sản phẩm, giá, mô tả, v.v.
* Bảng Link Sales: chứa thông tin về các giao dịch mua bán bao gồm khóa tự nhiên SaleID, khóa ngoại đến bảng Customer và khóa ngoại đến bảng Product.
* Bảng Satellite SalesDetails: chứa các thông tin chi tiết về giao dịch bao gồm số lượng sản phẩm, giá tiền, ngày mua hàng, v.v. Bảng này có khóa ngoại đến bảng Sales.

Trong mô hình Data Vault, các bảng Satellite có thể được thiết kế để lưu trữ các phiên bản lịch sử của các thuộc tính của một thực thể. Các phiên bản lịch sử này có thể được sử dụng để phân tích các xu hướng lịch sử của dữ liệu. Các bảng Link cũng có thể được thiết kế để lưu trữ các quan hệ lịch sử giữa các thực thể, cung cấp thông tin về các sự kiện liên quan đến các quan hệ đó.

Với cách thiết kế này, mô hình Data Vault có thể đảm bảo tính toàn vẹn dữ liệu và dễ dàng thực hiện các truy vấn phức tạp trong các hệ thống dữ liệu phân tích và các mô hình dữ liệu phức tạp.

## Các loại bảng SCD

Để quản lí dữ liệu một cách logic và truy xuất nhanh chóng, các bảng SCD được tạo ra và đóng vai trò quan trọng trong việc này, đặc biệt trong việc tạo ra datamart. Bảng SCD cung cấp các cơ chế để theo dõi sự thay đổi của dữ liệu trong thời gian, giúp cho việc phân tích và báo cáo trên các Datamart trở nên chính xác và đáng tin cậy hơn.

Khi xây dựng Datamart, chúng ta thường tập trung vào một loại thông tin cụ thể, như doanh số bán hàng, lợi nhuận, chi phí hoặc thông tin về khách hàng. Với mỗi loại thông tin này, ta cần có các bảng SCD tương ứng để quản lý sự thay đổi của các thuộc tính liên quan đến loại thông tin đó. Ví dụ, để quản lý thông tin về khách hàng, ta cần có các bảng SCD để quản lý thông tin về tên, địa chỉ, số điện thoại, email, v.v. của khách hàng trong thời gian.

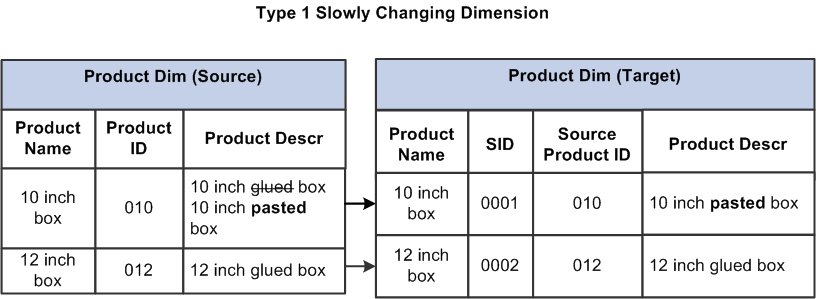
Các bảng SCD trong Datamart thường được thiết kế để giữ lại các bản ghi lịch sử của các đối tượng kinh doanh khi chúng thay đổi. Các bản ghi này bao gồm thông tin về các thuộc tính của đối tượng tại một thời điểm nhất định trong quá khứ.

Các bảng SCD thường có các cột sau:

* Surrogate Key: Đây là khóa chính được sử dụng để liên kết với các bảng khác trong Datamart. Thông thường, đây là một khóa tự tăng được sử dụng để thay thế khóa chính của đối tượng kinh doanh.
* Natural Key: Đây là một khóa được tạo bởi các thuộc tính của đối tượng kinh doanh, nhưng có thể không là duy nhất trong các bản ghi. Natural key được sử dụng để tìm kiếm các bản ghi liên quan đến đối tượng kinh doanh.
* Effective Date: Đây là ngày bắt đầu có hiệu lực của bản ghi. Khi bản ghi được cập nhật, ngày này sẽ được thay đổi để đánh dấu sự thay đổi.
* Expiry Date: Đây là ngày kết thúc hiệu lực của bản ghi. Khi bản ghi được cập nhật, ngày này của bản ghi trước đó sẽ được cập nhật để đánh dấu ngày kết thúc của bản ghi trước đó.
* Current Flag: Đây là một cờ được sử dụng để đánh dấu bản ghi hiện tại. Cờ này sẽ được đặt thành true cho bản ghi hiện tại và false cho các bản ghi lịch sử.

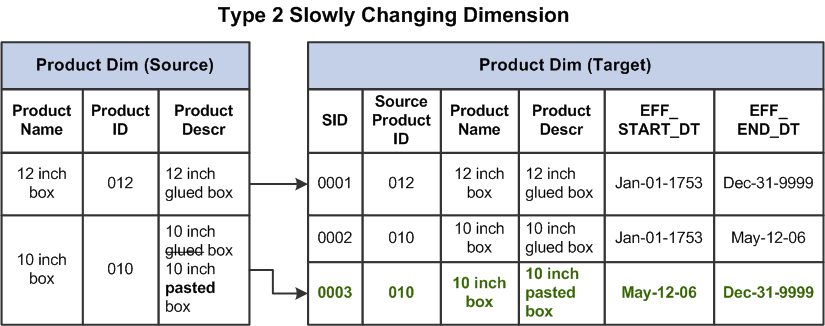
Các bảng SCD được sử dụng trong Datamart có thể được phân loại thành ba loại chính:

* SCD type 1: Thông tin chỉ được cập nhật, không lưu trữ lịch sử thay đổi. Trong trường hợp thông tin thay đổi, giá trị mới sẽ được ghi đè lên giá trị cũ. Loại bảng này được sử dụng khi chỉ cần lưu trữ thông tin mới nhất của thực thể và không quan tâm đến lịch sử thay đổi của thông tin. *Ví dụ: bảng lưu trữ tên khách hàng*.



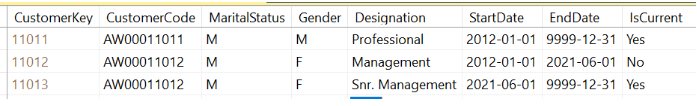
*Hình 1.9. Tính chất bảng SCD type 1*

* SCD type 2: Lưu trữ lịch sử thay đổi thông tin của thực thể bằng cách tạo mới bản ghi trong bảng khi thông tin thay đổi. Các bản ghi mới sẽ được đánh số thứ tự theo thời gian, và chỉ có bản ghi mới nhất được coi là có hiệu lực. Loại bảng này được sử dụng khi cần lưu trữ thông tin kinh doanh lịch sử và có thể phân tích các thay đổi của thông tin đó trong quá khứ. *Ví dụ: bảng lưu trữ thông tin khách hàng như tên, địa chỉ, số điện thoại, vv.*



*Hình 1.10. Tính chất bảng SCD type 2*

* SCD type 3: Lưu trữ thông tin thay đổi cùng với giá trị cũ trong cùng một bản ghi. Loại bảng này được sử dụng khi cần lưu trữ thông tin thay đổi, nhưng chỉ quan tâm đến thông tin thay đổi gần đây nhất và không cần phân tích lịch sử thay đổi. Ví dụ: bảng lưu trữ thông tin số điện thoại của khách hàng.



*Hình 1.11. Tính chất bảng SCD type 3*

Việc xử lý dữ liệu trong Datamart đòi hỏi sự chính xác và hiệu quả, bởi vì các dữ liệu này sẽ được sử dụng để phân tích và ra quyết định. Vì vậy, khi thiết kế các bảng SCD trong Datamart, cần cân nhắc các yếu tố sau:

* Xác định các thuộc tính quan trọng của bảng dữ liệu: Trước khi thiết kế bảng SCD, Ta cần xác định các thuộc tính quan trọng của bảng dữ liệu. Các thuộc tính này có thể bao gồm các thông tin về khách hàng, sản phẩm, doanh thu, chi phí, v.v.
* Xác định loại bảng SCD phù hợp: Sau khi xác định các thuộc tính quan trọng, Ta cần xác định loại bảng SCD phù hợp để lưu trữ các dữ liệu này. Có thể sử dụng các bảng SCD Type 1, Type 2 hoặc Type 3 để lưu trữ các dữ liệu này.
* Thiết kế bảng SCD: Sau khi xác định loại bảng SCD phù hợp, Ta cần thiết kế bảng SCD cho các dữ liệu. Các bảng SCD cần có các cột để lưu trữ thông tin về trạng thái hiện tại và trạng thái trước đó của các bản ghi, cùng với thời gian bắt đầu và kết thúc của mỗi trạng thái.
* Cập nhật dữ liệu: Khi có dữ liệu mới được thêm vào hoặc cập nhật, Ta cần xác định liệu dữ liệu này có liên quan đến bảng SCD nào và thực hiện các bước cập nhật tương ứng cho bảng đó. Các bước cập nhật bao gồm thêm mới bản ghi, cập nhật bản ghi hiện tại và kết thúc bản ghi trước đó.
* Đảm bảo tính toàn vẹn dữ liệu: Cuối cùng, khi xử lý dữ liệu trong Datamart, Ta cần đảm bảo tính toàn vẹn dữ liệu. Điều này bao gồm kiểm tra tính đúng đắn và đầy đủ của các bản ghi, đảm bảo tính nhất quán của các bảng dữ liệu.

Các bảng SCD có thể được cập nhật theo cách thủ công hoặc tự động thông qua quá trình ETL. Việc sử dụng bảng SCD trong datamart giúp cho việc phân tích và theo dõi dữ liệu theo thời gian trở nên dễ dàng và chính xác hơn.

## Áp dụng cho bài toán triển khai mô hình datavault cho ứng dụng phát hành game trực tuyến

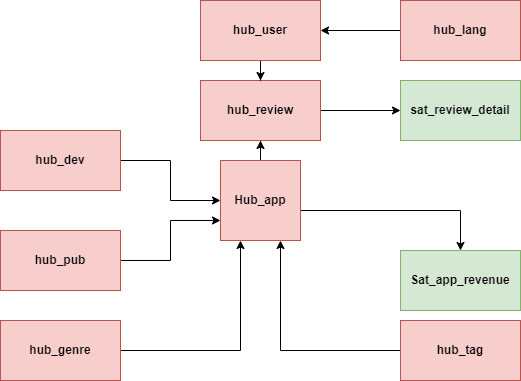
Với mô hình Datavault, nền tảng Steam có thể tận dụng các lợi ích và tiềm năng của việc xây dựng một hệ thống lưu trữ dữ liệu linh hoạt và mở rộng. Mô hình Datavault là một phương pháp thiết kế cơ sở dữ liệu phân cấp và tổ chức dữ liệu theo cách mà nó có thể dễ dàng được thêm vào và mở rộng mà không làm ảnh hưởng đến cấu trúc tổng thể. Điều này rất hữu ích trong trường hợp của Steam, vì nền tảng này phải xử lý một lượng lớn dữ liệu về trò chơi, người dùng, giao dịch và các hoạt động trong cộng đồng.

Nền tảng Steam có thể tổ chức dữ liệu thành các bộ sưu tập (hubs), các thực thể chính (hubs satellite) và các liên kết (links). Bộ sưu tập đại diện cho các lĩnh vực dữ liệu khác nhau, chẳng hạn như thông tin về trò chơi, người dùng và giao dịch. Các thực thể chính chứa thông tin chi tiết về từng đối tượng, ví dụ: một trò chơi cụ thể, một người dùng cụ thể hoặc một giao dịch cụ thể. Các liên kết kết nối các thực thể chính và tạo ra mối quan hệ giữa chúng.

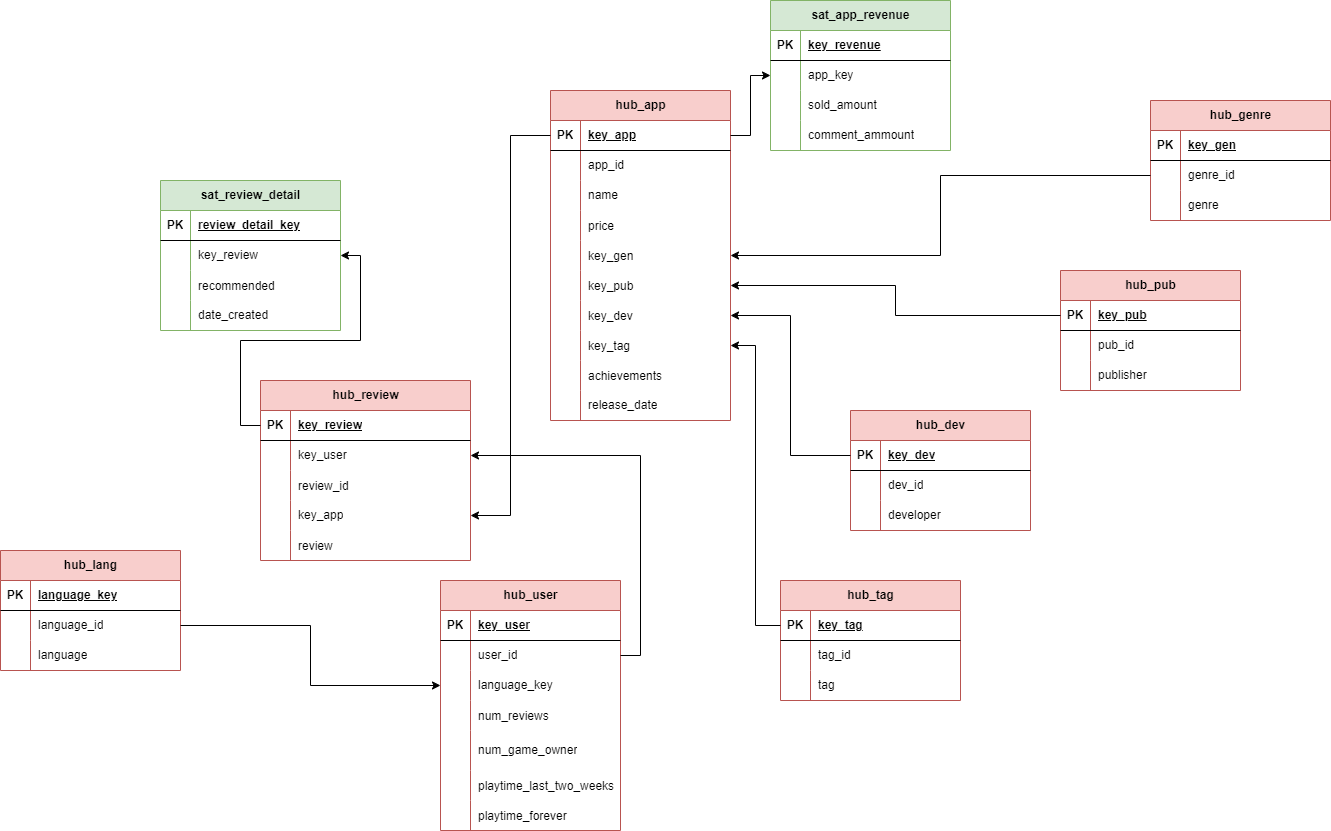
Mô hình Datavault cung cấp tính linh hoạt khi thêm mới dữ liệu vào hệ thống. Với việc Steam ngày càng phát triển và có thêm nhiều tính năng mới, việc có một hệ thống lưu trữ dữ liệu mở rộng giúp nền tảng này dễ dàng thích ứng với sự mở rộng và phát triển. Ngoài ra, mô hình Datavault cũng có khả năng tách biệt dữ liệu và cho phép truy cập nhanh chóng và hiệu quả vào các phần dữ liệu cụ thể. Điều này rất hữu ích khi Steam cần xử lý các truy vấn phức tạp và cung cấp dữ liệu cho các ứng dụng và dịch vụ khác nhau trên nền tảng.

Tổng quan, việc áp dụng mô hình Datavault trong nền tảng Steam mang lại nhiều lợi ích, bao gồm khả năng mở rộng, linh hoạt và hiệu quả trong việc quản lý và truy xuất dữ liệu. Điều này giúp cung cấp trải nghiệm tốt hơn cho người dùng và hỗ trợ sự phát triển và mở rộng của Steam trong tương lai. Với sự phát triển của công nghệ và sự cạnh tranh ngày càng gay gắt, mô hình Datavault đang trở thành lựa chọn hàng đầu cho doanh nghiệp, áp dụng để nâng cao chất lượng dịch vụ và tối ưu hoá quy trình làm việc.

Đối với bài toán cụ thể này, hiện nay em có đề xuất một mô hình datavault có thể áp dụng thực tiễn đối với một số các nền tảng phát hành trò chơi điện tử nói chung và Steam nói riêng:



*Hình 1.12. Mô hình data vault của steam*



*Hình 1.13. Mô hình data vault của steam với đầy đủ chi tiết*

### Mô tả chi tiết các thuộc tính trong Data vault

Bảng hub\_lang trong mô hình Data Vault được sử dụng để lưu trữ thông tin về ngôn ngữ của người dùng:

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| language\_key | String | Khóa của bảng hub\_lang | KEYLANG01 |
| language\_id | String | ID định danh của ngôn ngữ | LANG01 |
| language | String | Ngôn ngữ | schinese |

*Bảng 1.1. Mô tả chi tiết bảng hub\_lang*

Bảng hub\_user trong mô hình Data Vault được sử dụng để lưu trữ thông tin về người dùng. Bảng này giữ các bản ghi đại diện cho mỗi người dùng trong nền tảng Steam. Bảng hub "user" cho phép theo dõi và phân tích hoạt động của người dùng trong hệ thống. Thông qua việc lưu trữ thông tin như ngày tham gia, hoạt động, … em tiến hành xây dựng các bảng satellites nhằm phân tích các trải nghiệm người dùng thông qua ứng dụng.

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| key\_user | String | Khóa chính của bảng hub\_user | KEY76561198035975321 |
| user\_id | String | ID của người dùng | 76561198035975321 |
| language\_key | String | Khoá ngoại của bảng, để nối với bảng hub\_lang | KEYLANG01 |
| num\_reviews | INT | Số lượng bình luận mà người dùng đã bình luận | 12 |
| num\_game\_owner | INT | Số lượng trò chơi mà người dùng sở hữu | 23 |
| playtime\_last\_two\_weeks | INT | Tổng thời gian cuối cùng người dùng sử dụng ứng dụng trong vòng 2 tuần | 2456 |
| playtime\_forever | INT | Tổng thời gian người dùng sử dụng ứng dụng | 768951 |

*Bảng 1.2. Mô tả chi tiết bảng hub\_user*

Bảng hub\_dev trong mô hình data vault để lưu trữ thông tin về nhà phát triển game/sản phẩm. Bảng này chứa các bản ghi đại diện cho mỗi nhà phát triển hoặc nhà sản xuất trong ngữ cảnh của nền tảng phát hành trò chơi.

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| key\_dev | String | Khoá chính của bảng hub\_dev | KEYDEV001 |
| dev\_id | String | ID của nhà phát triển game | DEV001 |
| developer | String | Tên nhà phát triển game | Valve |

*Bảng 1.3. Mô tả chi tiết bảng hub\_dev*

Bảng hub\_genre trong mô hình data vault để lưu trữ thông tin về thể loại sản phẩm/game. Nó cho phép lưu trữ thông tin về các thể loại sản phẩm một cách hiệu quả và linh hoạt

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| key\_gen | String | Khoá chính của bảng hub\_gen | KEYGEN01 |
| gen\_id | String | ID của thể loại | GEN01 |
| genre | String | tên thể loại | Action |

*Bảng 1.4. Mô tả chi tiết bảng hub\_genre*

Bảng hub\_pub trong mô hình data vault để lưu trữ thông tin nhà phát hành sản phẩm/game. Bảng này chứa các bản ghi đại diện cho mỗi nhà phát triển hoặc nhà sản xuất trong ngữ cảnh của nền tảng phát hành trò chơi.

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| key\_pub | String | Khoá chính của bảng hub\_pub | KEYPUB001 |
| pub\_id | String | ID của nhà phát hành game | PUB001 |
| publisher | String | Tên nhà phát hành game | Valve |

*Bảng 1.5. Mô tả chi tiết bảng hub\_pub*

Bảng hub\_tag trong mô hình data vault để lưu trữ thông tin về các nhãn (tags), được sử dụng để phân loại hoặc đánh dấu các sản phẩm, game hoặc nội dung trong hệ thống. Nó cho phép lưu trữ thông tin về các nhãn một cách hiệu quả và linh hoạt.

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| key\_tag | String | Khoá chính của bảng hub\_tag | KEYTAG001 |
| tag\_id | String | ID của tag | TAG001 |
| tag | String | Tên tag name | Action |

*Bảng 1.6. Mô tả chi tiết bảng hub\_tag*

Bảng hub\_app trong mô hình data vault để lưu trữ thông tin về sản phẩm/game. Bảng này chứa các bản ghi đại diện cho mỗi sản phẩm hoặc game có sẵn trong nền tảng. Bảng này được nối với các bảng: Hub\_genre, hub\_dev, hub\_pub, hub\_tag. Em tiến hành xây dựng các bảng satellites nhằm phân tích các doanh thu, độ tương tác với người dùng thông qua nhà phát hành, thể loại, nhãn, nhà phát triển.

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| key\_app | String | Khoá chính của bảng hub\_app | KEYAPP01 |
| app\_id | int | ID của sản phẩm | 10 |
| name | String | tên sản phẩm | Counter-Strike |
| price | Float | giá sản phẩm | 7.19 |
| release\_date | datetime | ngày phát hành | 11/1/2000 |
| achievements | int | số lượng giải thưởng/thành tựu mà sản phẩm đã đạt được | 1 |
| key\_dev | String | Khoá ngoại của bảng hub\_app, dùng để nối với bảng hub\_dev | KEYDEV001 |
| key\_gen | String | Khoá ngoại của bảng hub\_app, dùng để nối với bảng hub\_gen | KEYGEN01 |
| key\_pub | String | Khoá ngoại của bảng hub\_app, dùng để nối với bảng hub\_pub | KEYPUB00001 |
| key\_tag | String | Khoá ngoại của bảng hub\_app, dùng để nối với bảng hub\_tag | KEYTAG001 |

*Bảng 1.7. Mô tả chi tiết bảng hub\_app*

Bảng hub\_review trong mô hình data vault để lưu trữ nhận xét của người dùng về sản phẩm/game. Thông qua việc lưu trữ thông tin như điểm đánh giá, ngày đăng nhận xét và các thuộc tính khác, em có thể xây dựng các bảng satellites để lưu trữ thông tin chi tiết về nhận xét, phân tích sự phân bổ điểm đánh giá, xác định nhận xét tích cực hay tiêu cực, và thực hiện các phân tích về quan điểm người dùng. Bảng này được nối với các bảng hub\_user để chiết xuất thông tin người dùng và hub\_app để phản ánh ý kiến của người dùng về sản phẩm/game đó.

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| key\_review | String | Khóa của bảng hub\_review | KEYREV00000001 |
| key\_user | String | Khoá ngoại của bảng hub\_review, dùng để nối với bảng hub\_user | KEY76561198035975321 |
| review\_id | String | ID của lời nhận xét đó | 10677921 |
| key\_app | String | Khoá ngoại của bảng hub\_review, dùng để nối với bảng hub\_app | KEYAPP9840 |
| review | String | Lời nhận xét của người dùng về sản phẩm đó | 傻逼游戏玩你妈 把老子显卡都烧了 |

*Bảng 1.8. Mô tả chi tiết bảng hub\_review*

Bảng sat\_review\_detail thể hiện chi tiết các phản ánh của người dùng về sản phẩm. Bảng này được xây dựng nhằm mục đích phân tích trải nghiệm của họ đối với sản phẩm. sat\_review\_detail được nối với bảng hub\_review thông qua các khoá ngoại

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| review\_detail\_key | String | Khóa của bảng sat\_review\_detail | KEYREV00000001DETAIL |
| key\_review | String | Khóa của bảng hub\_review | KEYREV00000001 |
| recommended | Boolean | Người chơi có đề xuất game đó hay không, giá trị của nó là null (trung lập), true (có gợi ý), false (không gợi ý) | True |
| date\_created | datetime | Thời gian comment đó được tạo | 17/09/2001 |

*Bảng 1.9. Mô tả chi tiết bảng sat\_review\_detail*

Bảng sat\_app\_revenue thể hiện chi tiết doanh thu của sản phẩm/game thông qua thể loại, nhà phát hành, nhà phát triển, … từ đó đưa ra các quyết định kinh doanh đúng đắn. Bảng sat\_app\_revenue được nối với bảng hub\_app thông qua các khoá ngoại

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| key\_revenue | String | Khóa của bảng sat\_app \_detail | KEYREV00000001 |
| app\_key | String | Khóa của bảng hub\_review | KEYAPP11 |
| sold\_amount | int | Tổng số lượt mua | 1846434290 |
| comment\_amount | int | Tổng số bình luận được tạo | 184643429 |

*Bảng 1.10. Mô tả chi tiết bảng sat\_app\_revenue*

### Xây dựng các tầng Data mart

Tiếp đến là tầng Datamart, tầng Datamart sẽ lấy dữ liệu từ tầng DWH được xử lý và cung cấp dữ liệu cho các người dùng cuối, chẳng hạn như báo cáo kinh doanh, phân tích dữ liệu. Tầng Datamart là nơi tập trung các báo cáo và phân tích dữ liệu cụ thể cho các phòng ban hoặc đơn vị trong. Với tầng Datamart, doanh nghiệp có thể tập trung vào việc phân tích dữ liệu theo một cách cụ thể hơn, từ đó giúp các bộ phận trong doanh nghiệp có thể nắm bắt được tình hình kinh doanh và đưa ra các quyết định đúng đắn hơn.

Trong tầng Datamart của mô hình Data Vault, các bảng Dim và Fact được sử dụng để tạo ra các báo cáo phân tích dữ liệu.

Các bảng Dim (Dimension) là những bảng chứa các thông tin mô tả về các đối tượng, thực thể trong hệ thống. Ví dụ như bảng Dim\_Customer có thể chứa thông tin về khách hàng như tên, địa chỉ, độ tuổi, v.v. Những thông tin này được sử dụng để phân loại, phân tích dữ liệu theo khách hàng.

Các bảng Fact (Fact Table) là những bảng chứa các thông tin số liệu thống kê, số liệu đo lường trong hệ thống. Ví dụ như bảng Fact\_Transaction chứa các thông tin về số tiền giao dịch, ngày thực hiện, loại giao dịch, v.v. Những thông tin này được sử dụng để tính toán, phân tích dữ liệu theo các chỉ tiêu, số liệu thống kê.

Các bảng Dim và Fact thường được kết nối với nhau thông qua các khóa ngoại (foreign key) để tạo ra các mối quan hệ giữa chúng và giúp dễ dàng phân tích dữ liệu theo các quan hệ, phân loại khách hàng, sản phẩm, v.v.

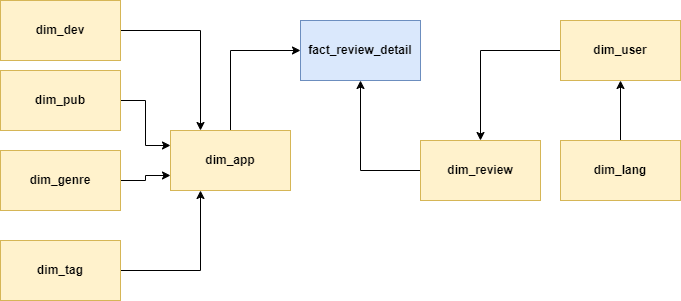
So sánh giữa bảng Dim và bảng Fact:

|  |  |  |
| --- | --- | --- |
| Đặc điểm | Bảng Dim | Bảng Fact |
| Định nghĩa | Bảng thông tin các đối tượng có liên quan tới bảng Fact. | Bao gồm các chỉ số đo lường các hoạt động trong tổ chức. |
| Thuộc tính | Nằm xung quanh bảng Fact trong Star schema và Snowflake Schema. | Trung tâm trong Star schema và Snowflake schema, xung quanh là các bảng Dim. |
| Thiết kế | Phải bao quát các thông tin của một đối tượng. Tránh nhét vào nhiều thông tin của nhiều đối tượng cùng một nơi. | Thiết kế đúng với trọng tâm của vấn đề hay nhu cầu. |
| Phục vụ | Thu thập các thông tin liên quan tới một đối tượng nào đó trong tổ chức (như thông tin cửa hàng chẳng hạn). | Một bảng Fact tập hợp các chỉ số đo lượng có liên quan tới các bảng Dim, dùng cho báo cáo, thống kê. |
| Kiểu dữ liệu | Mỗi bảng Dim chứa các thông tin về một đối tượng. Ví dụ bảng Product gồm Product Id, Category, SubCategory… | Bảng Fact chứa các cột dữ liệu số liên quan tới các cột đối tượng, khoá ngoại của bảng Dim. |
| Khóa chính | Mỗi bảng Dim đều có một khoá chính để xác định rõ tính độc nhất của mỗi dòng trong bảng. | Khoá chính của bảng Fact là tập hợp không lặp lại các khoá ngoại từ các bảng Dim. |
| Lưu trữ | Lưu trữ dữ liệu các đối tượng một cách chi tiết. | Được dùng để lưu trữ chỉ số, đồng thời để lọc kết quả trên nó dựa theo thông tin từ các bảng Dim. |
| Sự phân cấp | Có sự phân cấp: Ví dụ đối tượng Địa Điểm có thể chứa Quốc Gia, Khu vực, Quận/Huyện, Mã Zip… | Không có. |

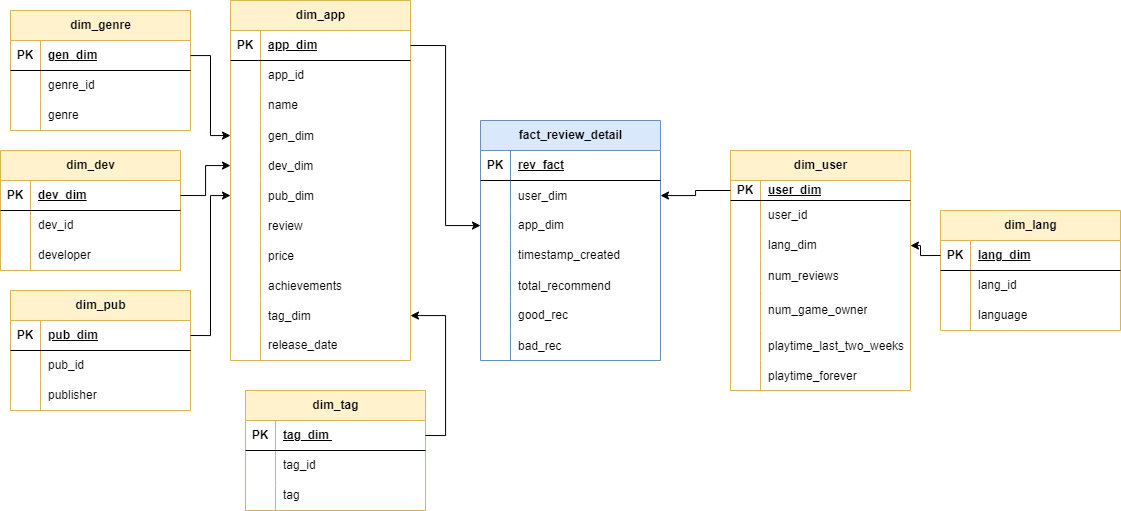
*Bảng 1.8. So sánh giữa dim và fact*

#### Mô hình Datamart đối với ý kiến người dùng

Đây là tầng datamart lưu trữ, phân tích các trải nghiệm người dùng thông qua ứng dụng.



*Hình 1.14. Mô hình data mart schema bông tuyết*



*Hình 1.15. Mô hình data mart đầy đủ các bảng*

Bảng gen\_dim lưu trữ các thể loại game.

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| dim\_gen | String | Khoá chính của bảng gen\_dim | DIMGEN01 |
| gen\_id | String | ID của thể loại | GEN01 |
| genre | String | tên thể loại | Action |

*Bảng 1.9. Mô tả chi tiết bảng genre\_dim*

Bảng dev\_dim lưu trữ thông tin các nhà phát triển

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| dim\_dev | String | Khoá chính của bảng dev\_dim | DIMDEV001 |
| dev\_id | String | ID của nhà phát triển game | DEV001 |
| developer | String | Tên nhà phát triển game | Valve |

*Bảng 1.10. Mô tả chi tiết bảng dev\_dim*

Bảng tag\_dim lưu trữ các thông tin về nhãn game

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| dim\_tag | String | Khoá chính của bảng tag\_dim | DIMTAG001 |
| tag\_id | String | ID của tag | TAG001 |
| tag | String | Tên tag name | Action |

*Bảng 1.11. Mô tả chi tiết bảng tag\_dim*

Bảng pub\_dim lưu trữ thông tin về nhà phát hành

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| dim\_pub | String | Khoá chính của bảng pub\_dim | DIMPUBB001 |
| pub\_id | String | ID của nhà phát hành game | PUB001 |
| publisher | String | Tên nhà phát hành game | Valve |

*Bảng 1.12. Mô tả chi tiết pub\_dim*

Bảng app\_dim lưu trữ thông tin về ứng dụng, liên kết với các bảng tag\_dim, dev\_dim, pub\_dim và genre\_dim để trích xuất các thông tin

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| dim\_app | String | Khoá chính của bảng app\_dim | DIMAPP01 |
| app\_id | int | ID của sản phẩm | 10 |
| name | String | tên sản phẩm | Counter-Strike |
| price | Float | giá sản phẩm | 7.19 |
| release\_date | datetime | ngày phát hành | 11/1/2000 |
| achievements | int | số lượng giải thưởng/thành tựu mà sản phẩm đã đạt được | 1 |
| dim\_dev | String | Khoá ngoại của bảng hub\_app, dùng để nối với bảng dev\_dim | DIMDEV001 |
| dim\_gen | String | Khoá ngoại của bảng app\_dim, dùng để nối với bảng hub\_gen | DIMGEN01 |
| dim\_pub | String | Khoá ngoại của bảng hub\_app, dùng để nối với bảng pub\_dim | DIMPUB00001 |
| dim\_tag | String | Khoá ngoại của bảng hub\_app, dùng để nối với bảng tag\_dim | DIMTAG001 |

*Bảng 1.13 Mô tả chi tiết bảng app\_dim*

Bảng lang\_dim dùng để lưu trữ thông tin về ngôn ngữ

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| dim\_lang | String | Khóa của bảng lang\_dim | DIMLANG01 |
| language\_id | String | ID định danh của ngôn ngữ | LANG01 |
| language | String | Ngôn ngữ | schinese |

*Bảng 1.14. Mô tả chi tiết bảng lang\_dim*

Bảng user\_dim dùng để lưu trữ thông tin về người dùng. Bảng này kết nối với bảng lang\_dim để trích xuất thông tin

|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| dim\_user | String | Khóa chính của bảng user\_dim | DIM76561198035975321 |
| user\_id | String | ID của người dùng | 76561198035975321 |
| language\_key | String | Khoá ngoại của bảng, để nối với bảng lang\_dim | DIMLANG01 |
| num\_reviews | INT | Số lượng bình luận mà người dùng đã bình luận | 12 |
| num\_game\_owner | INT | Số lượng trò chơi mà người dùng sở hữu | 23 |
| playtime\_last\_two\_weeks | INT | Tổng thời gian cuối cùng người dùng sử dụng ứng dụng trong vòng 2 tuần | 2456 |
| playtime\_forever | INT | Tổng thời gian người dùng sử dụng ứng dụng | 768951 |

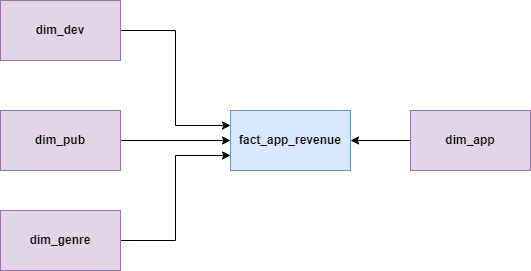
*Bảng 1.15. Mô tả chi tiết bảng user\_dim*

Bảng fact\_review\_detail là một bảng dữ liệu được sử dụng để lưu trữ các thông tin chi tiết về các đánh giá và phản hồi của người dùng đối với một ứng dụng cụ thể. Các bảng dim\_user, dim\_app được kết nối với bảng này thông qua khoá ngoại.

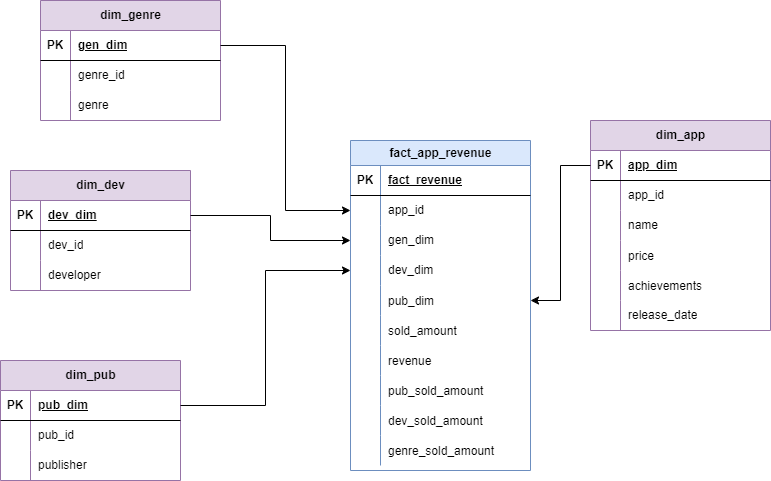
|  |  |  |  |
| --- | --- | --- | --- |
| Tên trường dữ liệu | Kiểu dữ liệu | Mô tả | Ví dụ dữ liệu |
| rev\_fact | String | Khoá chính của bảng fact\_review\_detail | FACT123REVIEW |
| dim\_user | String | Khoá ngoại của bảng fact\_review\_detail, dùng để nối bảng user\_dim | DIM76561198035975321 |
| dim\_app | String | Khoá ngoại của bảng fact\_review\_detail, dùng để nối bảng app\_dim | DIMAPP01 |
| time\_created | datetime | Ngày tạo comment đó | 6/6/1999 |
| total\_comment | INT | tổng số đề xuất của game đó | 32654942 |
| good\_rec | INT | tổng số đề xuất nhãn “có đề xuất” | 2864 |
| bad\_rec | INT | tổng số đề xuất nhãn “không đề xuất” | 98237846 |

*Bảng 1.16. Mô tả chi tiết đối với bảng fact\_review\_detail*

#### Mô hình Data mart đối với doanh thu



*Hình: Mô hình Datamart schema chòm sao*



*Hình: Mô hình Data mart với đầy đủ các bảng*

Data mart bao gồm các bảng dev\_dim, pub\_dim, genre\_dim, app\_dim có mô tả giống tầng Data mart trên.

Bảng fact\_app\_revenue được sử dụng để thu thập và tổng hợp thông tin về doanh thu mà một ứng dụng đã tạo ra. Bảng này bao gồm các thông tin như nguồn doanh thu, số lượng người dùng trả phí, mức bán.

## Thiết kế ETL (Extract-Transform-Load) và các công cụ hỗ trợ

Quy trình ETL (Extract, Transform, Load) là một phần quan trọng trong việc xây dựng hệ thống Data Warehouse. Nó giúp chuyển đổi dữ liệu từ các nguồn khác nhau thành một định dạng chung và tối ưu hóa để lưu trữ trong hệ thống Data Warehouse.

Cụ thể, các bước cơ bản của quy trình ETL gồm:

Bước trích xuất (Extract): Dữ liệu được trích xuất từ các nguồn dữ liệu khác nhau, bao gồm các hệ thống OLTP (Online Transaction Processing) và các tệp dữ liệu khác.

Bước chuyển đổi (Transform): Dữ liệu được chuẩn hóa và chuyển đổi để phù hợp với mô hình dữ liệu của Data Warehouse và các yêu cầu báo cáo. Bước này có thể bao gồm các hoạt động như lọc dữ liệu, định dạng lại các giá trị, tính toán các chỉ số mới và phân tích dữ liệu.

Bước tải lên (Load): Dữ liệu được tải lên vào các bảng trong Data Warehouse theo mô hình kiến trúc được thiết kế trước đó. Bước này có thể được thực hiện bằng cách sử dụng các công cụ như AWS Glue, AWS Data Pipeline, hay các công cụ ETL khác.

Có nhiều công cụ và kỹ thuật khác nhau được sử dụng trong quy trình ETL, tùy thuộc vào yêu cầu cụ thể của dự án. Dưới đây là một số công cụ và kỹ thuật phổ biến được sử dụng trong quy trình ETL:

Công cụ Extract:

* SQL Server Integration Services (SSIS): cho phép truy cập và trích xuất dữ liệu từ nhiều nguồn khác nhau, bao gồm cả các cơ sở dữ liệu quan hệ và các tệp dữ liệu định dạng khác nhau.
* Apache NiFi: cung cấp các tính năng trích xuất dữ liệu và chuyển đổi dữ liệu cho các mạng máy tính phân tán.

Công cụ Transform:

* Apache Spark: cho phép chuyển đổi dữ liệu và tính toán song song trên một cụm máy tính phân tán.
* Truy vấn SQL: cung cấp các hàm và cú pháp để chuyển đổi dữ liệu trong cơ sở dữ liệu quan hệ.
* Python và R: cung cấp các thư viện và công cụ để chuyển đổi dữ liệu, tính toán và xử lý dữ liệu.

Công cụ Load:

* SQL Server Integration Services (SSIS): cho phép lưu trữ dữ liệu đã chuyển đổi vào các bảng trong hệ thống Data Warehouse.
* Apache NiFi: cung cấp tính năng ghi dữ liệu vào nhiều hệ thống lưu trữ khác nhau, bao gồm cả các cơ sở dữ liệu quan hệ và các hệ thống lưu trữ NoSQL.

Khi quy trình ETL hoàn tất, dữ liệu sẽ được đưa vào Data Warehouse. Tại đây, các chuyên gia dữ liệu sẽ tiến hành xây dựng các báo cáo và dashboard dựa trên nhu cầu của người dùng. Hệ thống báo cáo tự động sẽ được tích hợp với Data Warehouse, cho phép người dùng truy cập vào các báo cáo và dashboard để phân tích dữ liệu một cách thuận tiện và nhanh chóng.

Để triển khai hệ thống báo cáo tự động, có nhiều công cụ phần mềm có thể được sử dụng. Các công cụ này cho phép người dùng thiết kế các báo cáo và dashboard, kết nối đến Data Warehouse và hiển thị dữ liệu trực quan trong các đồ thị và biểu đồ. Một số công cụ phổ biến được sử dụng trong hệ thống báo cáo tự động bao gồm:

* Tableau: Là một trong những công cụ phân tích dữ liệu và tạo báo cáo hàng đầu hiện nay. Tableau cho phép người dùng kết nối đến các nguồn dữ liệu khác nhau và thiết kế các báo cáo và dashboard một cách dễ dàng và trực quan.
* Power BI: Là một công cụ tạo báo cáo của Microsoft, cho phép người dùng tạo các báo cáo và dashboard bằng cách kết nối đến các nguồn dữ liệu khác nhau. Power BI cũng cung cấp nhiều tính năng phân tích dữ liệu mạnh mẽ để giúp người dùng hiểu rõ hơn về dữ liệu.

Cả hai công cụ này đều hỗ trợ kết nối trực tiếp đến hệ thống Data Warehouse của Ta, giúp cho việc tạo báo cáo và dashboard trở nên dễ dàng và nhanh chóng hơn. Bên cạnh đó, các công cụ này cũng cung cấp các tính năng bảo mật để đảm bảo rằng dữ liệu của Ta được bảo vệ một cách an toàn.

Trong kiến trúc chung của hệ thống, quy trình ETL và hệ thống báo cáo tự động được tích hợp với nhau để đảm bảo rằng dữ liệu được cập nhật và truy cập một cách nhanh chóng và hiệu quả. Khi quy trình ETL được thực hiện, dữ liệu mới được lưu trữ trong Data Warehouse và sẵn sàng để được truy cập và sử dụng bởi hệ thống báo cáo tự động.

Tất cả các công cụ này đều có khả năng kết nối đến Data Warehouse và cho phép người dùng tạo các báo cáo và dashboard để phân tích dữ liệu. Sau khi hoàn thành, các báo cáo và dashboard này có thể được xuất ra dưới dạng file PDF hoặc Excel để chia sẻ cho các thành viên khác trong tổ chức hoặc được truy cập trực tiếp từ trang web của hệ thống báo cáo tự động.

## Tóm tắt:

Chương I bắt đầu bằng một tổng quan về hệ thống kho dữ liệu và mô hình thiết kế. Đầu tiên, nó cung cấp một cái nhìn tổng quan về Steam, một nền tảng phân phối game trực tuyến. Sau đó, em đi vào khái niệm và khái quát về kho dữ liệu và kiến trúc chung của Data Warehouse. Tiếp theo, em giới thiệu mô hình logic của kho dữ liệu và mô hình dữ liệu đa chiều. Nó cung cấp một cái nhìn tổng quan về dữ liệu, bao gồm định nghĩa dữ liệu, cách thức lưu trữ dữ liệu và kiểu dữ liệu.

Chương tiếp tục với một tổng quan về mô hình Datavault, một phương pháp thiết kế kho dữ liệu linh hoạt và bền vững. Sau đó, nó trình bày quá trình thu thập và phân tích các yêu cầu về dữ liệu, cũng như phân tích quy trình nghiệp vụ của nền tảng Steam, bao gồm các khía cạnh như hoa hồng, bán phần cứng, phí nhà phát triển và marketing.

Chương I tiếp tục với việc hướng dẫn xây dựng hệ thống Data Warehouse và mô tả chi tiết về thiết kế cơ sở dữ liệu các lớp HUB, LINK và SATELLITE. Nó cung cấp các quy tắc và quy trình thiết kế cho mỗi lớp, cũng như cách xử lý dữ liệu trong từng lớp.

Chương cũng giới thiệu các quy trình ETL (Extract-Transform-Load) và phân tích dữ liệu trong hệ thống kho dữ liệu. Em đề cập đến việc thiết kế cơ sở dữ liệu và các loại bảng SCD (Slowly Changing Dimensions).

Cuối cùng, em đưa ra áp dụng của mô hình Datavault cho bài toán triển khai ứng dụng phát hành game trực tuyến và mô tả chi tiết về các thuộc tính trong Data vault, trình bày việc xây dựng các tầng Data mart. Cuối cùng, chương đề cập đến thiết kế ETL và các công cụ hỗ trợ.

# CHƯƠNG II: XÂY DỰNG HỆ THỐNG DATA WAREHOUSE SÀN THƯƠNG MẠI ĐIỆN TỬ STEAM TÍCH HỢP ĐA NGUỒN

## Tổng quan về hệ thống kho dữ liệu Steam

Hiện nay, một số doanh nghiệp đã chuyển đổi sang mô hình dữ liệu đa chiều (multidimensional model) để quản lý các thông tin kinh doanh như doanh thu, chi phí, khách hàng và sản phẩm. Các thông tin này được lưu trữ trong các khối dữ liệu và được sử dụng để tạo ra các báo cáo và phân tích kinh doanh, giúp quản lý và đưa ra các quyết định kinh doanh chính xác và hiệu quả.

Mô hình dữ liệu đa chiều là một phương pháp thiết kế cơ sở dữ liệu dùng để phân tích và báo cáo các thông tin kinh doanh. Nó sử dụng các khối (cubes) để lưu trữ dữ liệu, với mỗi khối tương ứng với một loại thông tin cụ thể (ví dụ: khối doanh thu, khối chi phí, khối sản phẩm, khối khách hàng, ...).

Một số ưu điểm của mô hình dữ liệu đa chiều là:

* Cung cấp khả năng phân tích và tìm kiếm dữ liệu nhanh chóng và hiệu quả.
* Cho phép thực hiện các phân tích đa chiều (multi-dimensional analysis) để đưa ra các quyết định kinh doanh.
* Tối ưu hóa hiệu suất truy vấn dữ liệu bằng cách sử dụng các chỉ số và vùng nhớ đệm (cache) để lưu trữ dữ liệu.

Cụ thể hơn, mô hình hệ thống dữ liệu áp dụng cho doanh nghiệp thương mại điện tử bao gồm các thành phần quan trọng như Data Warehouse, hệ thống báo cáo tự động, các dịch vụ AWS như Amazon S3, Amazon Redshift và Amazon RDS. Đây là một mô hình phức tạp, nhưng có thể mang lại nhiều lợi ích, bao gồm:

* Quản lý dữ liệu hiệu quả: Hệ thống Data Warehouse giúp sàn thương mại điện tử Steam quản lý dữ liệu hiệu quả hơn bằng cách tích hợp các nguồn dữ liệu khác nhau và chuyển đổi chúng thành một cấu trúc dữ liệu duy nhất và chuẩn hóa.
* Tăng tính chính xác của dữ liệu: Bằng cách sử dụng mô hình Data Vault, sàn thương mại điện tử Steam có thể đảm bảo tính chính xác của dữ liệu, giúp ngăn chặn việc xuất hiện các dữ liệu trùng lặp, mâu thuẫn và không nhất quán.
* Cung cấp báo cáo và phân tích dữ liệu chính xác: Hệ thống báo cáo tự động và các công cụ phân tích dữ liệu giúp sàn thương mại điện tử Steam truy cập và phân tích dữ liệu một cách nhanh chóng và chính xác, từ đó giúp họ đưa ra các quyết định kinh doanh chính xác hơn.
* Tăng tính khả dụng và linh hoạt của hệ thống: Sử dụng các dịch vụ AWS như Amazon S3, Amazon Redshift và Amazon RDS giúp tăng tính khả dụng và linh hoạt của hệ thống, cho phép sàn thương mại điện tử Steam mở rộng và thu nhỏ hệ thống dựa trên nhu cầu kinh doanh.

Tuy nhiên, việc chuyển đổi sang mô hình dữ liệu mới đòi hỏi các sàn thương mại điện tử Steam phải đối mặt với nhiều khó khăn và thách thức, bao gồm:

* Chi phí: Việc triển khai một mô hình dữ liệu mới đòi hỏi một khoản đầu tư lớn để phát triển hệ thống và đào tạo nhân viên.
* Độ phức tạp: Mô hình dữ liệu mới đòi hỏi một độ phức tạp cao trong việc xây dựng và bảo trì, đặc biệt là trong việc tích hợp dữ liệu từ các nguồn khác nhau.
* Sự phức tạp về kỹ thuật: Việc triển khai mô hình dữ liệu mới yêu cầu kiến thức về các công nghệ mới và các kỹ thuật xử lý dữ liệu phức tạp, đặc biệt là trong việc tích hợp dữ liệu từ các nguồn khác nhau.
* Đảm bảo tính toàn vẹn và an ninh dữ liệu: Việc chuyển đổi sang mô hình dữ liệu mới đòi hỏi các biện pháp đảm bảo tính toàn vẹn và an ninh dữ liệu, đặc biệt là trong việc xử lý thông tin cá nhân của khách hàng.
* Khả năng thích ứng: Mô hình dữ liệu mới đòi hỏi các sàn thương mại điện tử Steamphải thay đổi cách thức quản lý dữ liệu và thay đổi quy trình kinh doanh để có thể thích ứng với các thay đổi trong môi trường kinh doanh.

Tóm lại, việc chuyển đổi sang mô hình dữ liệu mới là một quá trình phức tạp và đòi hỏi sự đầu tư lớn về thời gian, chi phí và nguồn lực. Tuy nhiên, nếu được triển khai đúng cách, mô hình dữ liệu mới sẽ mang lại nhiều lợi ích cho sàn thương mại điện tử, bao gồm tăng tính toàn vẹn dữ liệu, tăng tính linh hoạt trong việc tích hợp dữ liệu và nâng cao khả năng phân tích dữ liệu để hỗ trợ quyết định kinh doanh.

## Mục tiêu của hệ thống dữ liệu đa chiều áp dụng trong sàn thương mại điện tử Steam

Hệ thống dữ liệu đa chiều đóng vai trò vô cùng quan trọng trong các doanh nghiệp hiện nay, giúp cung cấp thông tin chính xác, nhanh chóng và toàn diện cho quyết định kinh doanh. Mục tiêu của hệ thống dữ liệu đa chiều áp dụng trong ngân hàng là cung cấp các công cụ quản lý thông tin cho các nhà quản lý, giúp họ có thể hiểu được mối quan hệ giữa các thông tin và giúp đưa ra các quyết định kinh doanh đúng đắn.

Với hệ thống hiện tại của, kết hợp với các nền tảng công nghệ hỗ trợ, mục tiêu của hệ thống mô hình datavault cần đặt ra như sau:

* Tích hợp dữ liệu từ các nguồn khác nhau: Hệ thống datavault giúp tổng hợp và tích hợp dữ liệu từ nhiều nguồn khác nhau trong ngân hàng, bao gồm các hệ thống giao dịch, dữ liệu khách hàng, dữ liệu hóa đơn, dữ liệu sản phẩm và dịch vụ, để tạo ra một bộ dữ liệu toàn diện và đồng nhất.
* Cung cấp bộ dữ liệu tin cậy: Hệ thống datavault đảm bảo tính nhất quán và độ chính xác của dữ liệu thông qua quá trình xử lý và kiểm tra dữ liệu liên tục. Điều này giúp sàn thương mại điện tử Steamđảm bảo rằng các quyết định được đưa ra dựa trên thông tin chính xác và tin cậy.
* Cung cấp khả năng phân tích dữ liệu: Hệ thống datavault cung cấp các công cụ phân tích dữ liệu mạnh mẽ để giúp sàn thương mại điện tử Steamphát hiện ra các xu hướng và mô hình tiềm ẩn trong dữ liệu. Điều này giúp sàn thương mại điện tử Steam đưa ra các quyết định thông minh về sản phẩm, dịch vụ và chiến lược kinh doanh.
* Tăng tính linh hoạt: Hệ thống datavault cung cấp khả năng linh hoạt để thay đổi và thích nghi với nhu cầu kinh doanh thay đổi của thị trường. Điều này giúp sàn thương mại điện tử Steam dễ dàng mở rộng và thay đổi hệ thống dữ liệu để đáp ứng nhu cầu kinh doanh mới.

Thực hiện được các mục tiêu này, hệ thống datavault sẽ đóng vai trò quan trọng trong việc giúp sàn thương mại điện tử Steam stạo ra một bộ dữ liệu toàn diện và đáng tin cậy, giúp họ đưa ra các quyết định thông minh và linh hoạt để phục vụ khách hàng tốt hơn.

## Tổng quan về Amazon Web Service (AWS)

Amazon Web Services (AWS) là một nền tảng đám mây hàng đầu hiện nay, được phát triển và quản lý bởi Amazon. AWS cung cấp một loạt các dịch vụ đám mây, bao gồm lưu trữ, tính toán, cơ sở dữ liệu, mạng, truyền thông và các dịch vụ khác. Đây là một giải pháp đáng tin cậy và linh hoạt để xây dựng các ứng dụng, phân tích dữ liệu và triển khai các dịch vụ trực tuyến.

AWS cho phép người dùng tạo ra các tài nguyên đám mây một cách dễ dàng, mà không cần phải đầu tư nhiều vào phần cứng, phần mềm hoặc quản lý hạ tầng. Người dùng có thể dễ dàng mở rộng các tài nguyên này, tùy chỉnh các cấu hình và điều khiển chúng từ bất kỳ đâu.

Các dịch vụ của AWS cung cấp tính năng bảo mật cao, đáp ứng các tiêu chuẩn và quy định an ninh khắt khe nhất trên toàn thế giới. AWS cũng cung cấp các công cụ và dịch vụ giúp người dùng quản lý và kiểm soát bảo mật các ứng dụng và dữ liệu của họ trên nền tảng đám mây.

Các dịch vụ của AWS cũng được tối ưu hóa để chạy các ứng dụng và dịch vụ trực tuyến, cũng như các nhu cầu phân tích dữ liệu lớn. AWS cung cấp các công cụ và dịch vụ để giúp người dùng thu thập, lưu trữ và xử lý dữ liệu với các tính năng chuyên biệt như xử lý dữ liệu trực tuyến, phân tích dữ liệu, học máy và trí tuệ nhân tạo. Cụ thể, AWS cung cấp cho khách hàng của mình một loạt các dịch vụ đám mây, bao gồm:

* Tính toán: AWS cung cấp một loạt các dịch vụ tính toán như Amazon Elastic Compute Cloud (EC2), Amazon Elastic Container Service (ECS), Amazon Lambda, Amazon Lightsail, và Amazon Batch.
* Lưu trữ: AWS cung cấp một loạt các dịch vụ lưu trữ như Amazon Simple Storage Service (S3), Amazon Elastic Block Store (EBS), Amazon Elastic File System (EFS), và Amazon Glacier.
* Cơ sở dữ liệu: AWS cung cấp một loạt các dịch vụ cơ sở dữ liệu như Amazon Relational Database Service (RDS), Amazon DynamoDB, Amazon ElastiCache, và Amazon Redshift.
* Mạng và bảo mật: AWS cung cấp một loạt các dịch vụ mạng và bảo mật như Amazon Virtual Private Cloud (VPC), AWS Direct Connect, AWS Identity and Access Management (IAM), và Amazon GuardDuty.
* Quản lý và giám sát: AWS cung cấp một loạt các dịch vụ quản lý và giám sát như Amazon CloudWatch, AWS CloudFormation, AWS CloudTrail, và AWS Config.
* Trí tuệ nhân tạo và machine learning: AWS cung cấp một loạt các dịch vụ trí tuệ nhân tạo và machine learning như Amazon SageMaker, Amazon Rekognition, và Amazon Comprehend.

Ngoài ra, AWS cũng cung cấp các công cụ quản lý, giám sát và triển khai dịch vụ đám mây của mình như AWS Management Console, AWS Command Line Interface (CLI), và AWS Software Development Kits (SDKs) để phục vụ các khách hàng đa dạng.

Sử dụng AWS giúp các tổ chức giảm chi phí vận hành và bảo trì hệ thống, tăng tính sẵn sàng và độ tin cậy, cải thiện hiệu suất và linh hoạt, đồng thời tăng khả năng mở rộng hệ thống để phục vụ cho nhu cầu phát triển tương lai của tổ chức.

## Các dịch vụ lưu trữ và tính toán

### EC2 (Elastic Compute Cloud):

EC2 là một dịch vụ của AWS cung cấp cho người dùng khả năng thuê các máy ảo trên đám mây để chạy các ứng dụng. EC2 cho phép người dùng tùy chỉnh kích thước và cấu hình của các máy ảo, từ đó đáp ứng các yêu cầu về tính toán của ứng dụng của họ.

Ví dụ: Một công ty khởi nghiệp mới có nhu cầu chạy các máy chủ web để cung cấp dịch vụ cho khách hàng của mình. Thay vì mua và quản lý các máy chủ vật lý, họ có thể sử dụng EC2 để tạo các máy ảo chạy các máy chủ web của họ trên đám mây của AWS.

### S3 (Simple Storage Service):

S3 là dịch vụ lưu trữ đối tượng của AWS, cho phép người dùng lưu trữ và truy xuất các đối tượng dữ liệu (ví dụ như tài liệu, hình ảnh, video) trên đám mây. S3 cung cấp tính năng độ tin cậy cao, khả năng mở rộng và độ phổ biến rộng rãi.

Ví dụ: Một công ty sản xuất phim muốn lưu trữ các bản phim, ảnh và tài liệu liên quan đến các dự án của mình trên đám mây. Họ có thể sử dụng S3 để lưu trữ và phân phối các tệp này trên đám mây của AWS, giúp cho việc truy xuất dữ liệu và chia sẻ nó giữa các thành viên trong nhóm dễ dàng hơn.

### EBS (Elastic Block Store):

EBS là một dịch vụ lưu trữ đĩa ảo của AWS, cho phép người dùng đính kèm các đĩa ảo vào các máy ảo EC2. EBS cho phép các máy ảo EC2 lưu trữ dữ liệu trên đám mây theo cách truyền thống với độ tin cậy cao và khả năng mở rộng linh hoạt.

Ví dụ: Một công ty muốn tạo và quản lý các ổ đĩa lưu trữ được gắn kết với các máy ảo EC2 của họ. Họ có thể sử dụng EBS để tạo và quản lý các khối lưu trữ này trên đám mây của AWS.

### Glacier:

Glacier là dịch vụ lưu trữ dữ liệu dài hạn của AWS, được thiết kế để lưu trữ các bản sao lưu dữ liệu hoặc các tập tin lưu trữ dữ liệu lớn. Glacier cung cấp tính năng lưu trữ dữ liệu bền vững, an toàn và có tính khả dụng cao với chi phí thấp.

Ví dụ: Một công ty bán lẻ có thể sử dụng Glacier để lưu trữ các bản sao lưu hằng ngày của cơ sở dữ liệu của mình. Với chi phí lưu trữ thấp, công ty có thể lưu trữ các bản sao lưu này trong một kho lưu trữ dài hạn mà không cần lo lắng về chi phí cao.

### RDS (Relational Database Service)

RDS là một dịch vụ quản lý cơ sở dữ liệu quan hệ trong đám mây của AWS, hỗ trợ nhiều loại cơ sở dữ liệu quan hệ như MySQL, PostgreSQL, Oracle và SQL Server. Với RDS, người dùng không phải quan tâm đến việc quản lý cơ sở dữ liệu, như backup, sao lưu, giảm thiểu downtime, cân bằng tải, phân vùng dữ liệu... Tất cả các tác vụ này được RDS quản lý và tự động hóa, giúp người dùng tập trung vào việc phát triển ứng dụng của mình.

Ví dụ: Một công ty muốn phát triển ứng dụng của mình và cần sử dụng một cơ sở dữ liệu quan hệ. Họ không muốn phải quản lý cơ sở dữ liệu và tập trung vào phát triển ứng dụng. Bằng cách sử dụng RDS, họ có thể dễ dàng triển khai một cơ sở dữ liệu quan hệ và không cần phải lo lắng về việc quản lý nó.

### Lambda:

Lambda là dịch vụ tính toán của AWS, cho phép người dùng chạy các chức năng trên đám mây mà không cần quản lý các máy chủ. Lambda cung cấp khả năng xử lý các sự kiện (event-driven) một cách tự động và khả năng tự động mở rộng dựa trên nhu cầu.

Ví dụ: Một ứng dụng di động có thể sử dụng Lambda để thực hiện chức năng xử lý hình ảnh. Khi người dùng tải lên hình ảnh, Lambda sẽ thực thi một chương trình xử lý để chỉnh sửa hình ảnh và trả về kết quả cho ứng dụng. Điều này giúp giảm tải cho máy chủ, tăng tốc độ xử lý và tiết kiệm chi phí cho việc quản lý máy chủ.

### Redshift:

Redshift là một dịch vụ cơ sở dữ liệu trực tuyến (OLAP) dựa trên cấu trúc của AWS, được thiết kế để xử lý lượng lớn dữ liệu. Với Redshift, người dùng có thể triển khai một kho dữ liệu có khả năng mở rộng, hiệu quả về chi phí, và có thể truy vấn các tập dữ liệu lớn với tốc độ cao.

Ví dụ: Một công ty muốn phân tích và hiểu được thông tin về hành vi mua hàng của khách hàng của mình. Họ có một lượng lớn dữ liệu cần phải xử lý. Bằng cách sử dụng Redshift, họ có thể triển khai một kho dữ liệu có khả năng mở rộng và truy vấn các tập dữ liệu lớn với tốc độ cao, giúp họ nhanh chóng tìm thấy các thông tin quan trọng để đưa ra quyết định kinh doanh.

### Glue:

AWS Glue là một dịch vụ quản lý ETL (Extract, Transform, Load) được cung cấp bởi Amazon Web Services (AWS). Dịch vụ này cung cấp các công cụ để giúp trích xuất dữ liệu từ các nguồn khác nhau, biến đổi dữ liệu và tải dữ liệu vào các nơi lưu trữ khác nhau. AWS Glue giúp ta giảm thiểu thời gian và công sức cần thiết để xử lý dữ liệu, và nhanh chóng phân tích dữ liệu và trích xuất thông tin cần thiết.

Một ví dụ về việc sử dụng Glue đó là khi chúng ta muốn tích hợp dữ liệu từ các nguồn khác nhau vào một kho dữ liệu duy nhất để phục vụ cho mục đích phân tích. Ví dụ, ta có thể muốn trích xuất dữ liệu từ các tệp CSV được lưu trữ trên Amazon S3, sau đó biến đổi chúng bằng một số bước để chuẩn hóa dữ liệu và tải chúng vào cơ sở dữ liệu Redshift để phân tích.

Với Glue, ta có thể tạo một kế hoạch công việc ETL bằng cách chọn các nguồn dữ liệu và định cấu hình các bước biến đổi. Sau đó, Glue sẽ tự động chạy các kế hoạch công việc đó trên một lịch trình được xác định trước hoặc bằng cách kích hoạt chúng bằng các sự kiện cụ thể như khi có tệp mới được tải lên S3. Ngoài ra, Glue còn cung cấp cho chúng ta các công cụ để quản lý, giám sát và xác định lỗi trong các kế hoạch công việc.

### Autoscaling:

Autoscaling là một dịch vụ quản lý tài nguyên linh hoạt trên AWS, cho phép tự động tăng hoặc giảm số lượng máy chủ (instances) dựa trên tải của ứng dụng. Với tính năng này, khách hàng có thể đảm bảo ứng dụng luôn đáp ứng được nhu cầu sử dụng mà không cần thực hiện thủ công việc tăng hoặc giảm số lượng instances.

Autoscaling hoạt động bằng cách đặt một ngưỡng tải (threshold) cho tài nguyên và tự động thực hiện hành động tăng hoặc giảm số lượng instances nếu tải vượt quá hoặc giảm dưới ngưỡng đó. Tính năng này còn cho phép người dùng tạo ra các bộ điều kiện và luật (policies) để điều chỉnh số lượng instances, điều kiện này có thể được thiết lập dựa trên nhiều yếu tố khác nhau như tải CPU, tải mạng, số lượng requests…

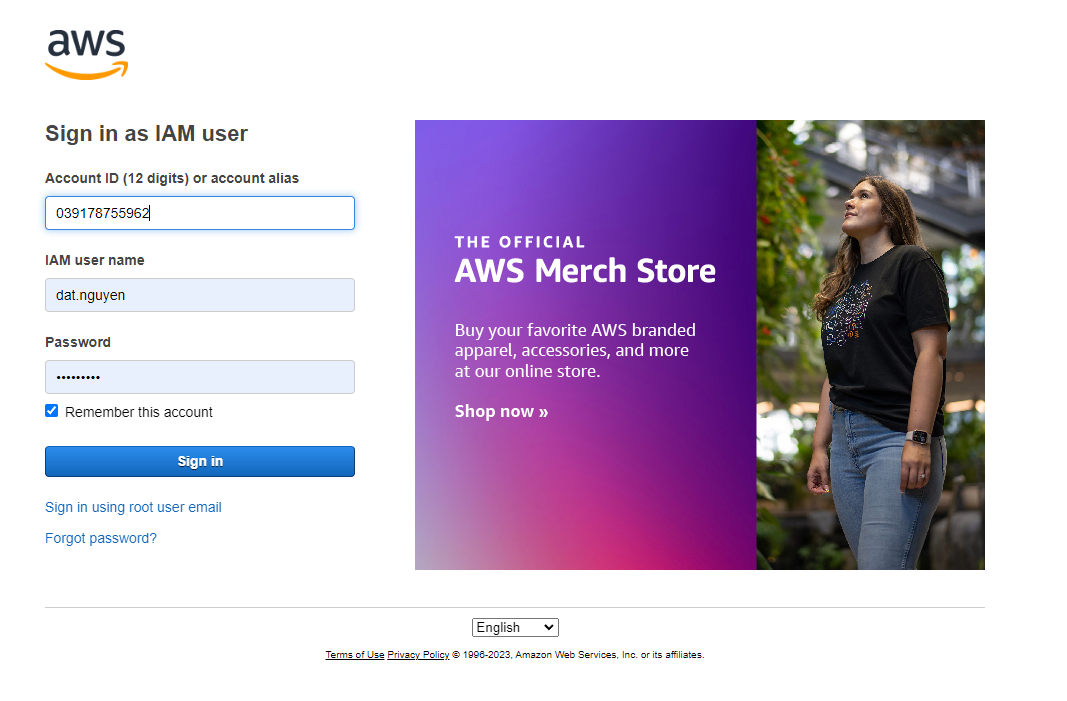
Autoscaling giúp tối ưu hóa việc sử dụng tài nguyên, giảm chi phí và tăng tính sẵn sàng cho ứng dụng. Bên cạnh đó, nó còn giúp cải thiện hiệu suất của hệ thống bằng cách tự động điều chỉnh số lượng instances để đáp ứng được yêu cầu tải khác nhau.

## Triển khai AWS cho mô hình datavault

Hệ thống Cloud Computing đã trở thành một phương tiện quan trọng trong việc xây dựng hệ thống Data Warehouse hiện đại. Trong đó, Amazon Web Services (AWS) là một trong những nền tảng phổ biến nhất được sử dụng bởi các doanh nghiệp để lưu trữ và quản lý dữ liệu. DataVault là một mô hình thiết kế Data Warehouse, có khả năng lưu trữ và xử lý dữ liệu lớn, đa nguồn và phân tán. Để sử dụng được dịch vụ này, cần thực hiện theo các bước sau:

**Bước 1:** Chuẩn bị môi trường AWS

Đầu tiên, cần đăng ký tài khoản AWS và tạo một môi trường đám mây cho dự án. Sau đó, cần cài đặt và cấu hình các công cụ AWS như S3, EC2, Redshift, Glue, Athena để triển khai mô hình Datavault.



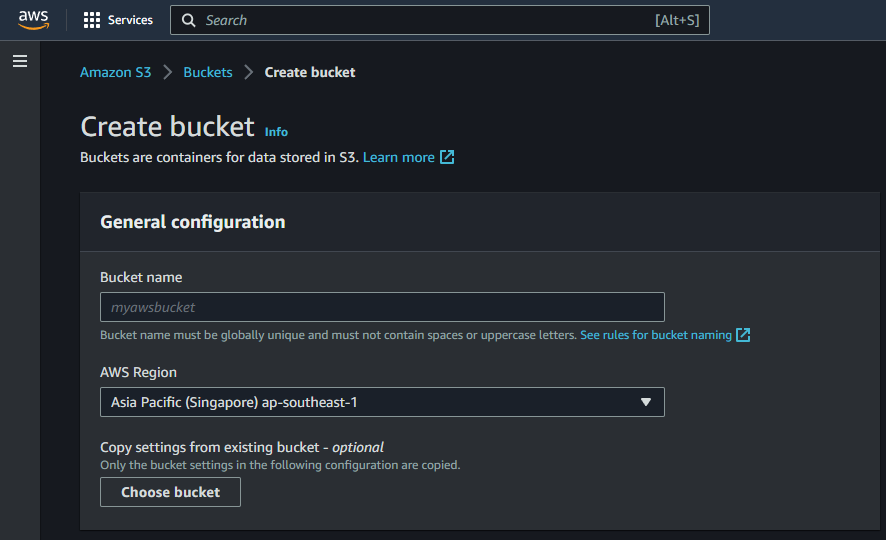
*Hình 2.1. Giao diện đăng nhập /đăng kí vào AWS*

**Bước 2:** Chuẩn bị dữ liệu

Trước khi triển khai Datavault trên AWS, cần chuẩn bị các nguồn dữ liệu. Ngoài ra người dùng cần phân tích các nguồn dữ liệu đó và quyết định sử dụng định dạng lưu trữ nào như CSV, JSON hay Parquet. Hơn nữa, các script ETL để trích xuất, chuyển đổi và tải dữ liệu vào AWS cũng cần chuẩn bị để có thể sử dụng sau này.

**Bước 3:** Tạo Amazon S3 bucket

Bước tiếp theo là tạo một Amazon S3 bucket để lưu trữ dữ liệu của mô hình Datavault. Đầu tiên, truy cập vào AWS Management Console, chọn dịch vụ S3, sau đó nhấn nút "Create Bucket", nhập tên bucket và chọn khu vực lưu trữ, sau đó nhấn "Create".



*Hình 2.2. Giao diện tạo Bucket*

Sau khi tạo được Bucket, việc cần làm tiếp theo là đẩy dữ liệu từ các nguồn dữ liệu của ngân hàng vào S3 Bucket.



*Hình 2.3. Các bảng lưu trữ trong S3*

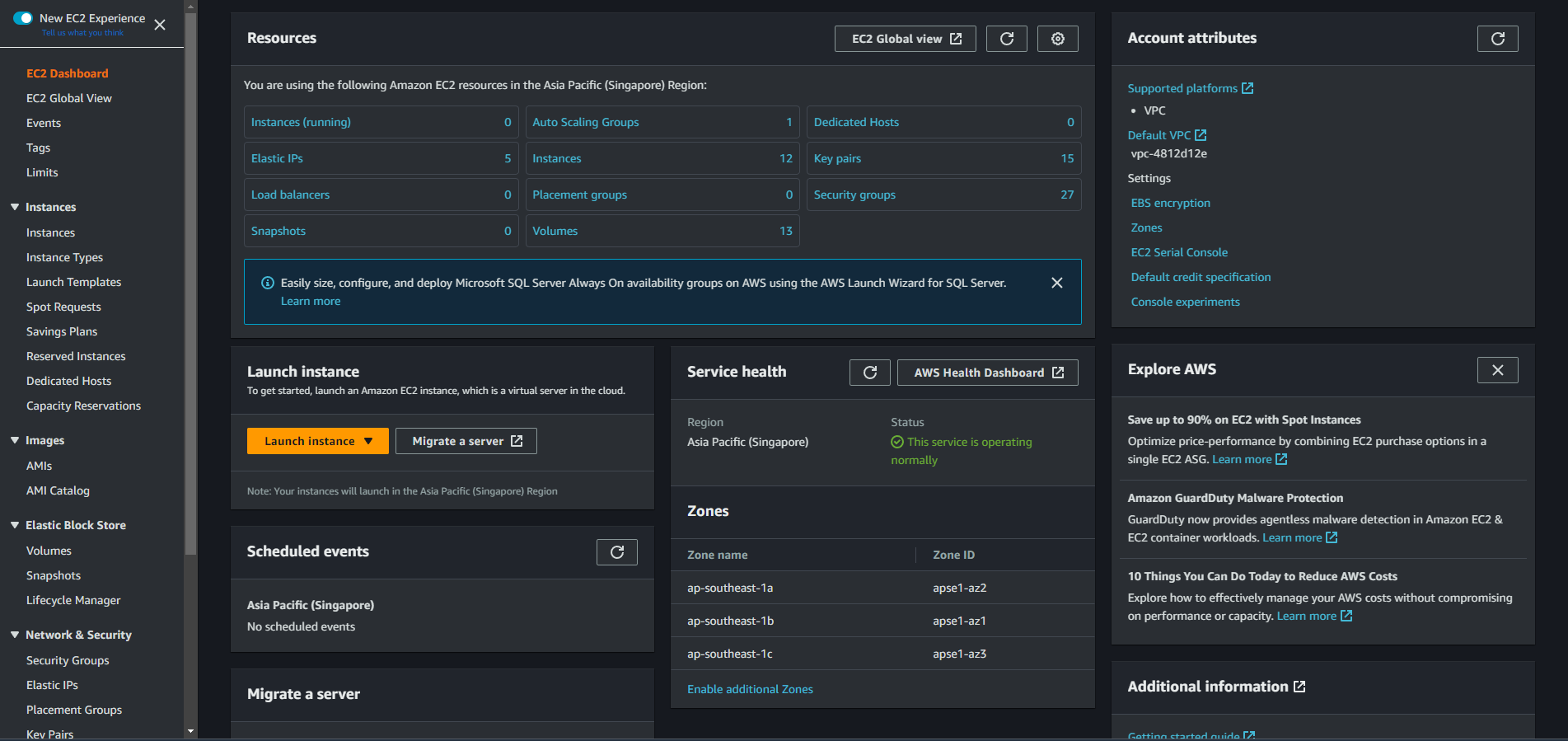
**Bước 4:** Triển khai máy chủ EC2

Amazon Elastic Compute Cloud (EC2) là một dịch vụ cung cấp máy chủ ảo trong đám mây của Amazon Web Services (AWS). Nó cho phép người dùng thuê các máy chủ ảo để chạy các ứng dụng, lưu trữ dữ liệu, hoặc thực hiện các nhiệm vụ khác trên đám mây của AWS. Trong quá trình triển khai mô hình Data Vault trên AWS, Ta có thể cần triển khai các máy chủ EC2 để chạy các công cụ ETL như Glue hoặc các script ETL do Ta viết để trích xuất và tải dữ liệu.

Các bước để triển khai các máy chủ EC2 như sau:

* Đăng nhập vào AWS Management Console và chọn phần EC2 Dashboard.
* Nhấn nút "Launch Instance" để bắt đầu quá trình tạo máy chủ EC2.
* Chọn AMI (Amazon Machine Image) để cài đặt trên máy chủ EC2. AWS cung cấp nhiều AMI khác nhau, bao gồm cả các phiên bản Linux và Windows.
* Chọn loại máy chủ (Instance Type) phù hợp với yêu cầu của Ta. AWS cung cấp nhiều loại máy chủ khác nhau, từ nhỏ đến lớn, và từ thấp đến cao về khả năng xử lý.
* Cấu hình chi tiết các tùy chọn máy chủ EC2, bao gồm kích thước ổ cứng, phân vùng, tường lửa và các thông số mạng khác.
* Kiểm tra lại các thông tin đã cấu hình và nhấn "Launch" để bắt đầu quá trình tạo máy chủ EC2.
* Để truy cập vào máy chủ EC2, Ta có thể sử dụng các phương thức như SSH hoặc Remote Desktop Protocol (RDP) tùy thuộc vào loại hệ điều hành của máy chủ EC2.

Khi Ta đã truy cập vào máy chủ EC2, Ta có thể cài đặt các công cụ ETL như Glue hoặc các script ETL do Ta viết để trích xuất và tải dữ liệu.

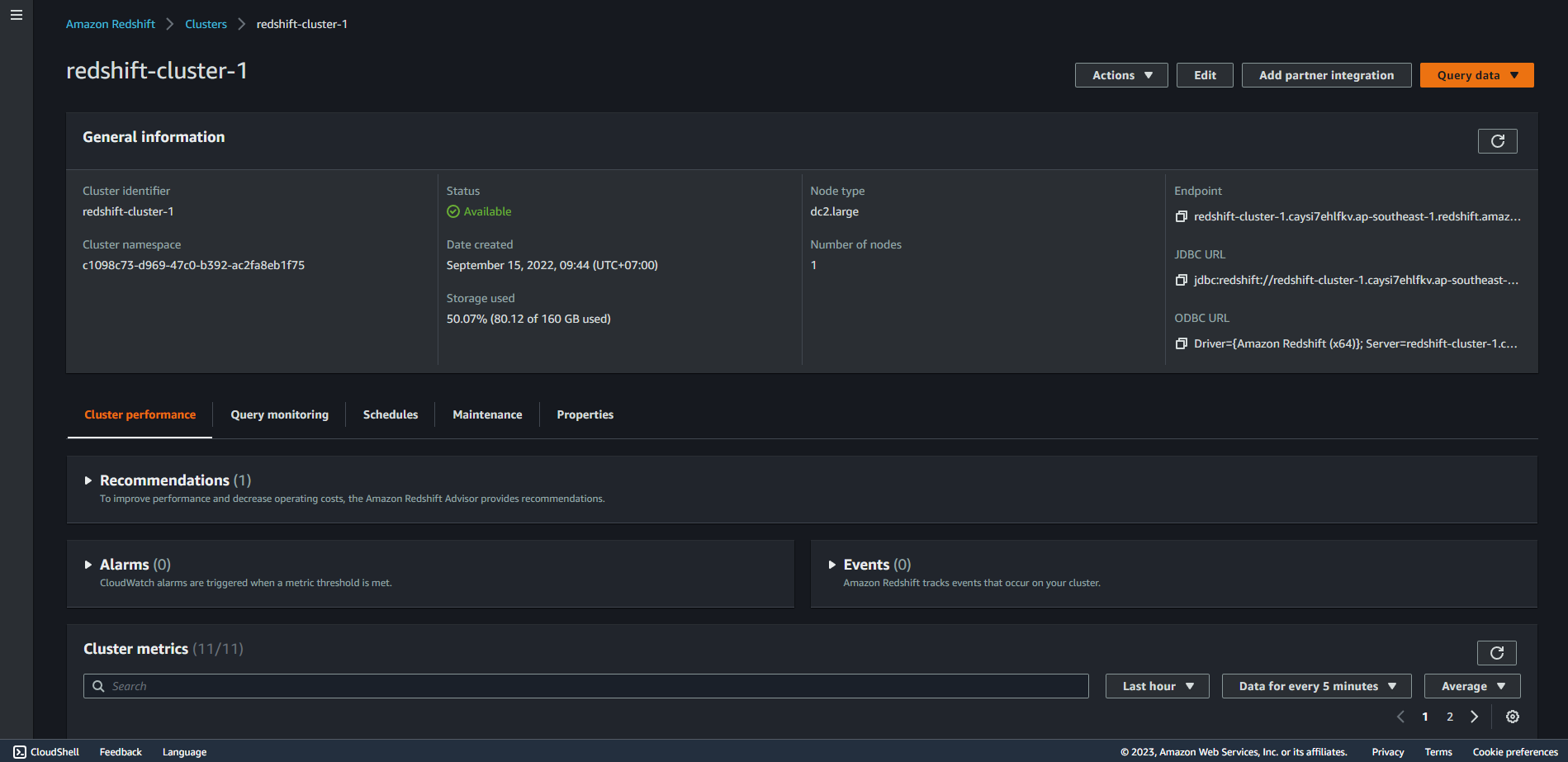


*Hình 2.4. Giao diện máy chủ EC2*

**Bước 5:** Triển khai cụm Redshift để lưu trữ dữ liệu của Datavault

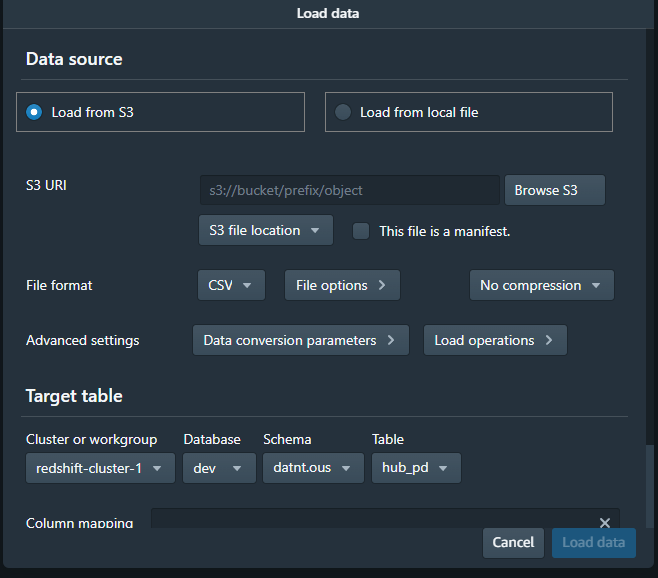
Redshift được xây dựng trên cơ sở công nghệ của PostgreSQL và được tối ưu hóa cho các tập dữ liệu lớn, cho phép người dùng lưu trữ và truy vấn dữ liệu một cách nhanh chóng và hiệu quả. Để triển khai triển khai mô hình Data Vault sử dụng Redshift trên AWS, ta cần thực hiện các bước:

* Tạo Redshift cluster: Sau khi tài khoản AWS được tạo, cần tạo một Redshift cluster để lưu trữ dữ liệu. Ta có thể tạo cluster bằng cách truy cập vào bảng điều khiển Redshift và chọn tùy chọn "Launch cluster". Trong quá trình tạo cluster, ta cần chỉ định kích thước cluster, vùng địa lý và các thông tin khác cần thiết.



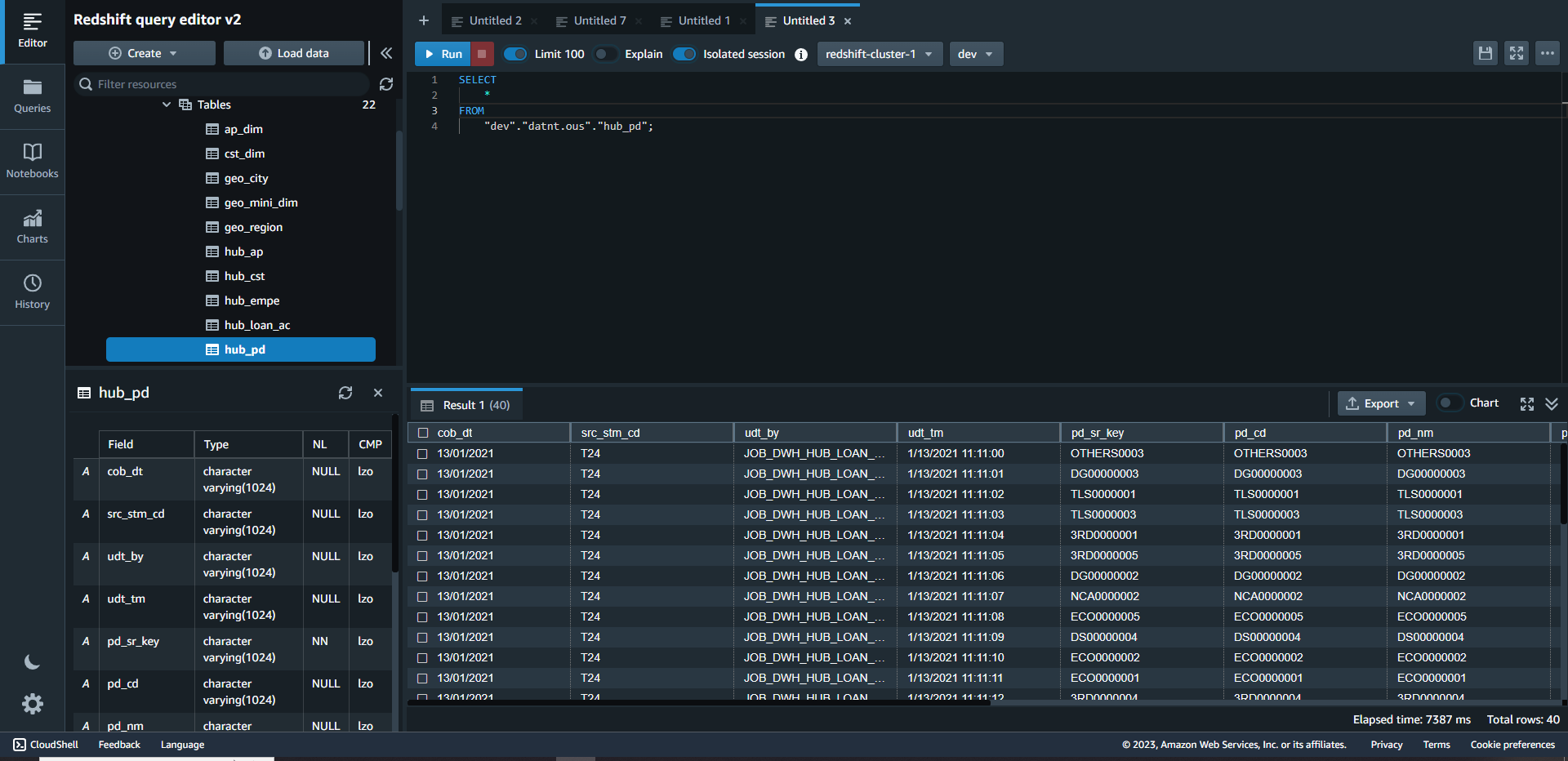
*Hình 2.5. Giao diện Redshift Cluster*

* Tạo các bảng Data Vault: Sau khi cluster đã được tạo, Ta cần tạo các bảng Data Vault trên Redshift. Các bảng này bao gồm các bảng hub, satellite và link.
* Triển khai ETL: Ta cần tạo các kết nối đến các nguồn dữ liệu khác như S3 để tạo các bảng Data Vault, sau đó ta cần triển khai quy trình ETL để trích xuất và tải dữ liệu vào các bảng Data Vault. Quy trình ETL có thể triển khai bằng cách sử dụng Glue, hoặc các script Python.



*Hình 2.6. Load data từ S3 vào Redshift*

Sau khi đẩy dữ liệu từ S3 vào Redshift, ta có các bảng Hub, Sat, cũng như các bảng ở tầng Datamart.



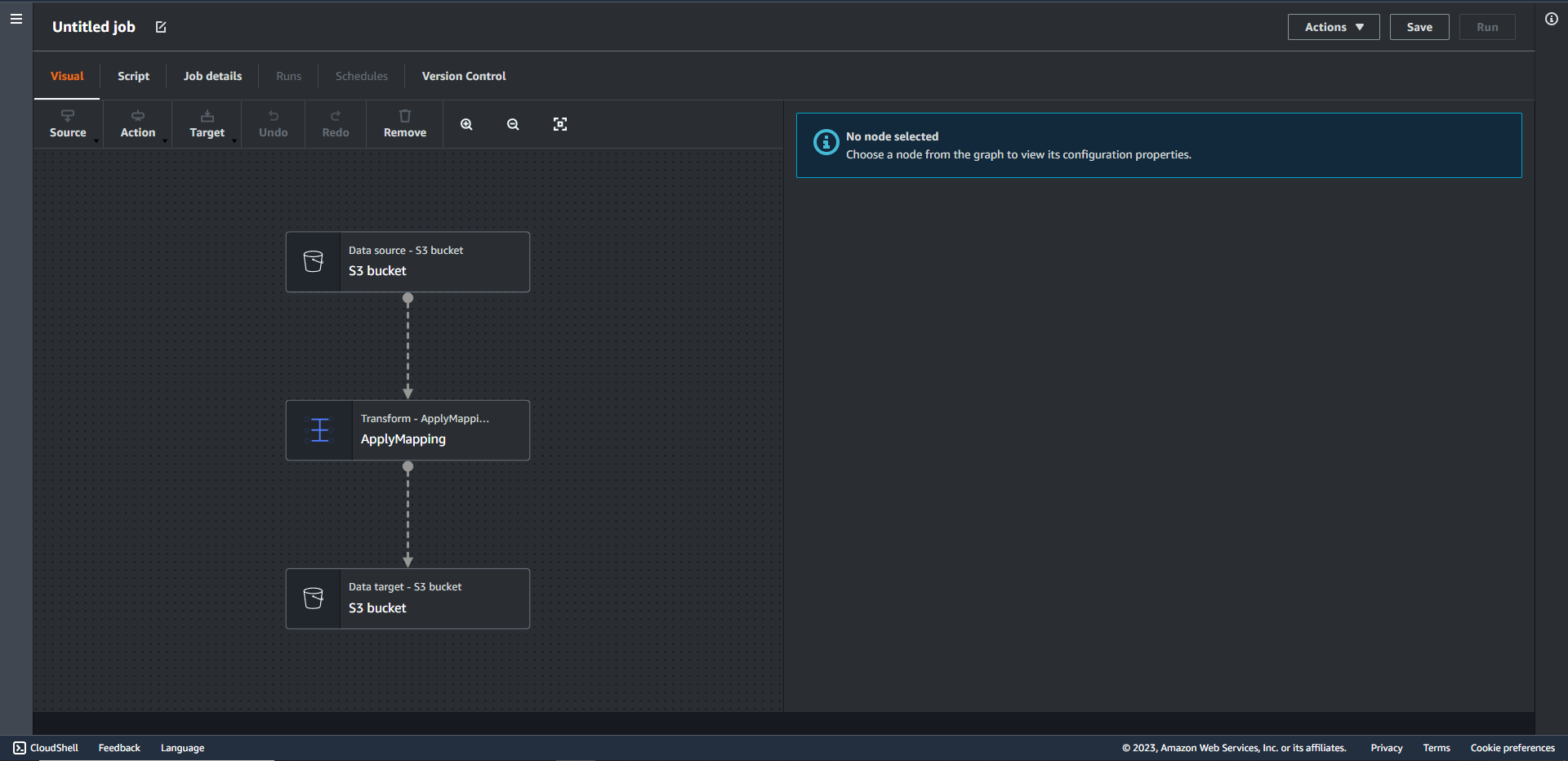
*Hình 2.7. Giao diện querry Redshift*

**Bước 6:** Triển khai và cấu hình công cụ Glue để thực hiện các bước ETL

Công cụ Glue của AWS là một dịch vụ quản lý dữ liệu được thiết kế để giúp cho việc xử lý và biến đổi dữ liệu dễ dàng hơn trong hệ thống AWS. Glue sử dụng Apache Spark làm nền tảng để xử lý dữ liệu và hỗ trợ các ngôn ngữ lập trình phổ biến như Python, Scala và Java.

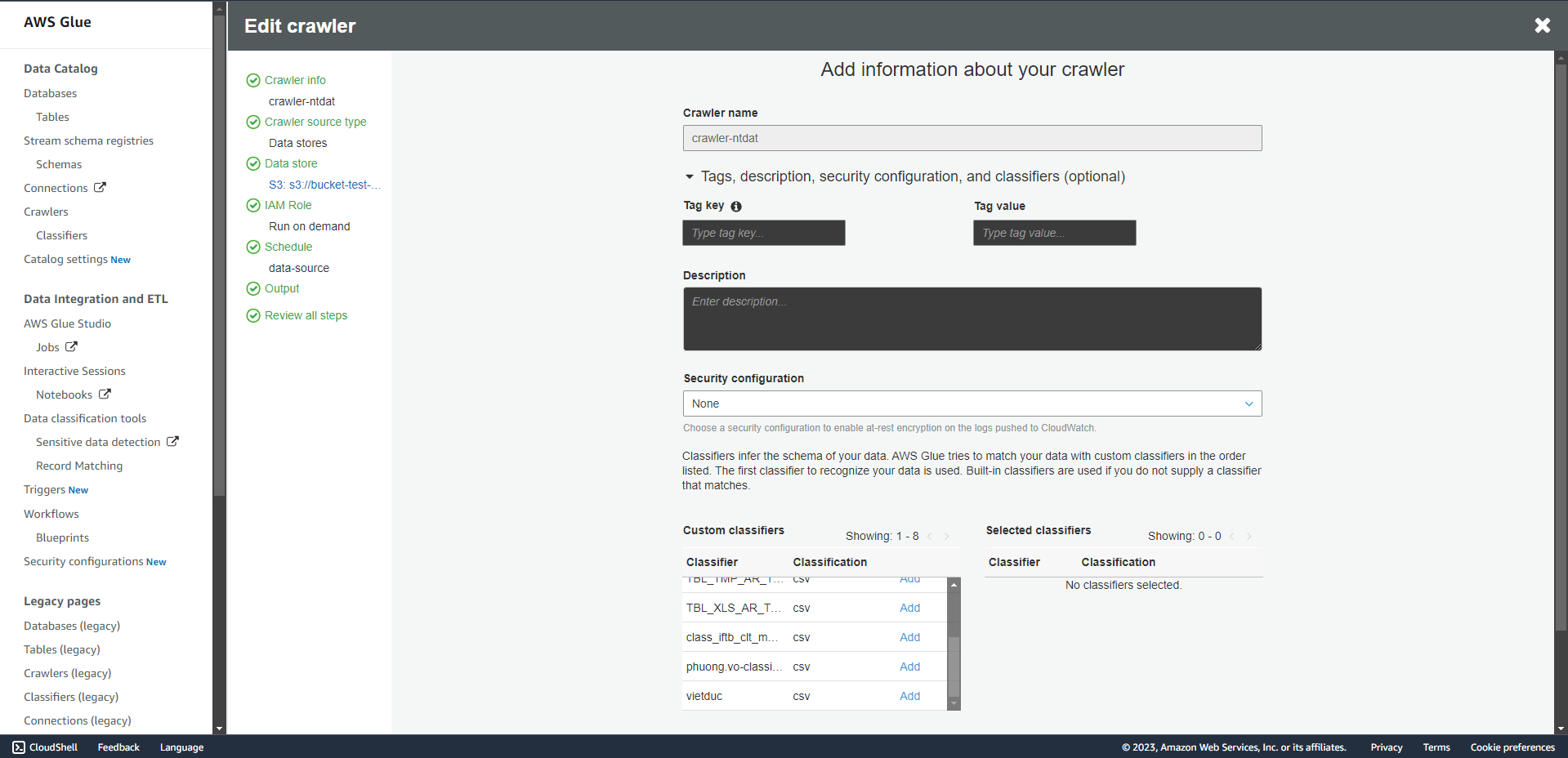
Để triển khai và cấu hình công cụ Glue để thực hiện các bước ETL, Ta có thể thực hiện các bước sau:

* Tạo một định danh IAM (Identity and Access Management) trên AWS để truy cập và quản lý Glue. Đây là bước quan trọng để đảm bảo an toàn cho dữ liệu.
* Tạo một cơ sở dữ liệu trên Amazon S3 hoặc trên một nguồn dữ liệu khác. Các tệp dữ liệu sẽ được lưu trữ trên đó và được trích xuất, biến đổi và tải lên cơ sở dữ liệu mục tiêu.



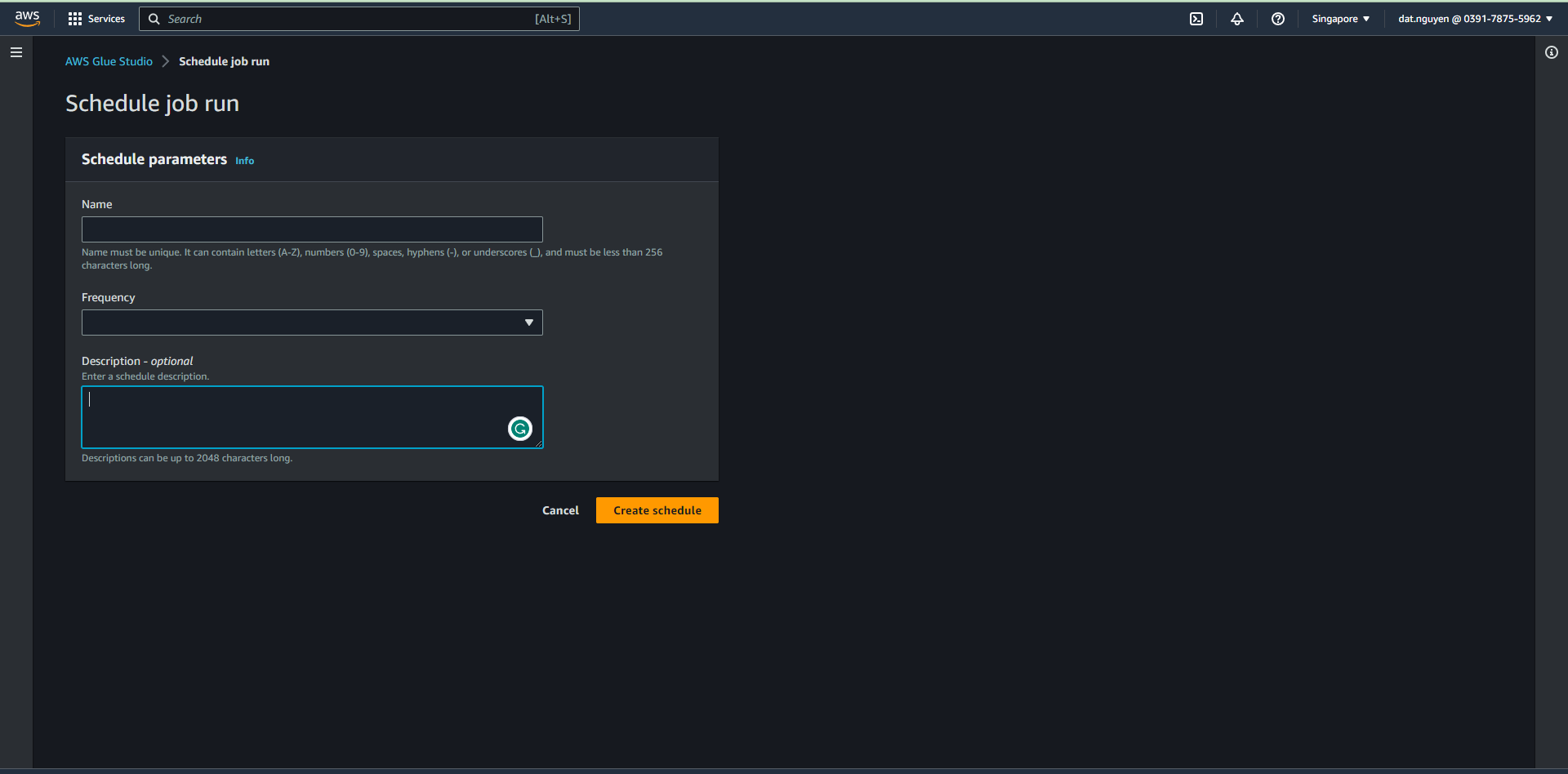
*Hình 2.8. Giao diện Job ETL*

* Tạo một crawler Glue để đọc các tệp dữ liệu và tạo bảng dữ liệu. Glue Crawler có khả năng tự động phát hiện cấu trúc của các tệp dữ liệu và tạo bảng dữ liệu tương ứng trong cơ sở dữ liệu mục tiêu.



*Hình 2.9. Giao diện Glue Crawler*

* Thiết lập các trường biến đổi để xử lý các tệp dữ liệu. Glue hỗ trợ nhiều trường biến đổi để xử lý dữ liệu, bao gồm lọc, ghép nối, chuyển đổi định dạng, v.v.
* Thiết lập lịch trình cho ETL Jobs. Glue hỗ trợ lịch trình tự động cho các ETL Jobs, vì vậy ta có thể đặt lịch cho chúng theo nhu cầu.



*Hình 2.10. Đặt lịch chạy Job ETL*

* Chạy các ETL Jobs. Sau khi đã cấu hình và lên lịch cho các ETL Jobs, ta có thể chạy chúng để thực hiện các bước trích xuất, biến đổi và tải dữ liệu.
* Kiểm tra và giám sát quá trình ETL. Glue cung cấp các công cụ để giám sát các ETL Jobs và xem các lỗi hoặc vấn đề có thể xảy ra.

**Bước 7:** Triển khai mô hình DWH sử dụng Lambda và Step Functions

Lambda là một dịch vụ tính toán đám mây được cung cấp bởi AWS, cho phép chạy mã nguồn một cách tự động mà không cần quản lý máy chủ. Trong hệ thống Data Warehouse, Lambda có thể được sử dụng để thực hiện các chức năng như xử lý và lọc dữ liệu đầu vào, chuyển đổi định dạng dữ liệu, và ghi dữ liệu vào các kho lưu trữ. Ví dụ, khi có dữ liệu mới được thêm vào Data Warehouse, Lambda có thể được sử dụng để tự động xử lý và cập nhật các báo cáo hoặc thông tin liên quan.

Lambda có thể được kết hợp với các dịch vụ khác của AWS như S3, RDS hoặc DynamoDB để truy xuất hoặc lưu trữ dữ liệu. Lambda cũng có thể được kết hợp với Step Functions để tạo ra các luồng công việc phức tạp hơn, ví dụ như thực hiện nhiều tác vụ liên quan đến dữ liệu hoặc thực hiện các tác vụ tự động theo một thứ tự nhất định.

Trong AWS Lambda, ta có thể tạo các trigger để kích hoạt chạy function. Các trigger có thể được kết nối với nhiều dịch vụ khác nhau của AWS, chẳng hạn như S3, DynamoDB, API Gateway, SNS, CloudFormation, CloudWatch, và nhiều hơn nữa.

Khi trigger được kích hoạt, Lambda sẽ tự động thực thi function mà ta đã cấu hình. Các trigger trong Lambda giúp cho việc xử lý dữ liệu trở nên tự động và hiệu quả hơn, giảm thiểu tối đa thời gian và công sức của người quản lý.

Trong đề tài này, Lambda có thể được sử dụng để xử lý các tác vụ tự động như cập nhật dữ liệu, tính toán báo cáo hoặc thông tin thống kê, và tự động cập nhật thông tin cho hệ thống báo cáo tự động. Các bước tiến trình của lambda function có thể như sau:

* Khi có dữ liệu mới được tải lên vào AWS S3 bucket, sự kiện trigger sẽ được kích hoạt.
* Lambda function được triệu hồi để xử lý dữ liệu.
* Lambda function sẽ đọc dữ liệu từ S3 bucket.
* Sau đó, lambda function sẽ xử lý dữ liệu đó để chuẩn hóa và trích xuất các thông tin quan trọng cần thiết.
* Dữ liệu được chuẩn hóa và trích xuất được lưu trữ vào AWS DynamoDB để được sử dụng bởi hệ thống báo cáo tự động.
* Lambda function kết thúc việc xử lý và trả về kết quả cho hệ thống.

AWS Step Functions là một dịch vụ quản lý tiến trình cho phép người dùng xây dựng, chạy và quản lý các ứng dụng máy chủ không đồng bộ và được phân tán. Step Functions cho phép bạn mô hình hóa ứng dụng của mình thành các công việc (tasks), trạng thái (states) và chuyển tiếp giữa các trạng thái, tạo ra một mô hình luồng công việc có thể mở rộng và khả năng xử lý lỗi tốt hơn.

Step Functions cung cấp cho người dùng một giao diện đồ họa để thiết kế các luồng công việc phức tạp bằng cách sử dụng các trạng thái và các hành động được xác định trước. Nó cũng cung cấp API để tự động hóa các hoạt động trong các ứng dụng.

Kết hợp Lambda và Step Functions là một trong những cách tốt nhất để tạo ra một hệ thống hoạt động trơn tru và linh hoạt. Lambda được sử dụng để thực thi các tác vụ xử lý dữ liệu, trong khi Step Functions được sử dụng để quản lý các bước thực hiện và điều khiển luồng làm việc của chúng.

Khi kết hợp Lambda và Step Functions, chúng ta có thể tạo ra các quy trình làm việc phức tạp, bao gồm nhiều bước khác nhau. Các bước này có thể được thực hiện tuần tự hoặc song song, tùy thuộc vào yêu cầu của quy trình.

Các bước trong quy trình có thể gọi đến các Lambda function để xử lý dữ liệu. Khi một Lambda function hoàn thành nhiệm vụ của mình, nó trả về kết quả cho Step Functions, sau đó Step Functions sẽ quyết định bước tiếp theo của quy trình.

Khi sử dụng kết hợp Lambda và Step Function, quy trình xử lý dữ liệu sẽ bao gồm các bước sau:

* Trigger: Một sự kiện được kích hoạt, ví dụ như khi dữ liệu mới được đưa vào một bucket S3 hoặc khi một item mới được thêm vào DynamoDB.
* Lambda Function: Một Lambda Function được kích hoạt để xử lý sự kiện. Lambda Function này sẽ đọc dữ liệu từ S3 hoặc DynamoDB và xử lý nó theo cách được định nghĩa.
* Step Function: Khi Lambda Function hoàn tất xử lý, nó sẽ gọi một Step Function để quản lý quy trình xử lý dữ liệu. Step Function sẽ gọi các Lambda Function khác để xử lý dữ liệu và đưa ra quyết định về việc tiếp tục xử lý dữ liệu hoặc dừng quy trình.
* Đầu ra: Khi quy trình xử lý dữ liệu hoàn tất, kết quả được trả về cho ứng dụng hoặc lưu trữ trong một hệ thống lưu trữ khác như S3 hay DynamoDB.

Việc kết hợp Lambda và Step Function cho phép xử lý dữ liệu theo cách tự động và hiệu quả hơn, giúp tiết kiệm thời gian và tài nguyên cho các nhà phát triển và doanh nghiệp.

**Bước 8:** Xây dựng các báo cáo

Cuối cùng, ta có thể sử dụng các công cụ báo cáo như Amazon QuickSight hoặc Tableau kết nối tới các bảng Data Vault trên Redshift để truy vấn và hiển thị dữ liệu. Việc xây dựng các dashboard và report là một phần quan trọng trong việc triển khai hệ thống Data Warehouse. Các công cụ Business Intelligence (BI) như Tableau, Power BI, v.v. được sử dụng để trực quan hóa dữ liệu và hiển thị các thông tin quan trọng đến người dùng cuối.

Trước khi xây dựng dashboard và report, cần phải định nghĩa các yêu cầu và mục đích của chúng. Sau đó, cần phải thiết kế các mẫu báo cáo và dashboard để đảm bảo chúng đáp ứng được các yêu cầu và mục đích đã đề ra.

Sau khi thiết kế các mẫu báo cáo và dashboard, cần phải xác định các chỉ số và thông tin cần hiển thị. Các chỉ số này thường được lấy từ các bảng fact và dimension trong Data Warehouse. Các công cụ BI được sử dụng để truy vấn dữ liệu từ Data Warehouse và hiển thị các thông tin này trực quan và dễ hiểu.

Để đảm bảo các dashboard và report được cập nhật liên tục, cần sử dụng các công cụ ETL để cập nhật dữ liệu mới vào Data Warehouse và đồng bộ hóa với dashboard và report. Các công cụ ETL giúp tự động hóa việc trích xuất, chuyển đổi và tải dữ liệu, từ đó giảm thiểu thời gian và công sức cho việc cập nhật dữ liệu.

Việc xây dựng các dashboard và report giúp cho các người dùng cuối có thể trực quan hóa dữ liệu và hiểu rõ hơn về các thông tin quan trọng. Nó là một phần quan trọng trong việc đưa ra quyết định và phát triển kinh doanh.

## Công cụ hỗ trợ ETL – Dbeaver

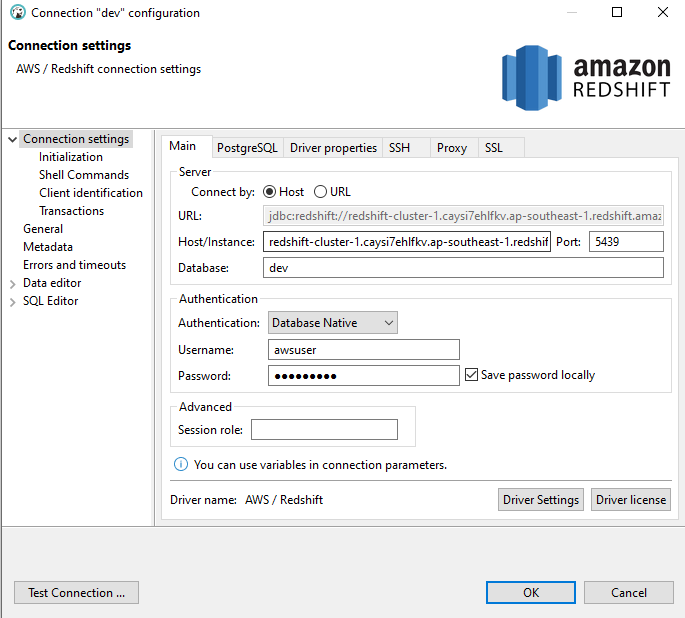
Ngoài việc querry câu lệnh SQL trên AWS Redshift, chúng ta có thể sử dụng DBeaver để làm điều đó. Những lợi ích đem lại có thể kể đến như:

* Dễ dàng sử dụng: Dbeaver cung cấp giao diện đồ họa thân thiện và dễ sử dụng, cho phép người dùng kết nối và quản lý cơ sở dữ liệu Redshift một cách dễ dàng.
* Hỗ trợ đa nền tảng: Dbeaver hỗ trợ đa nền tảng, cho phép người dùng kết nối và quản lý cơ sở dữ liệu Redshift từ nhiều hệ điều hành khác nhau.
* Hỗ trợ quản lý schema: Dbeaver cung cấp khả năng quản lý schema, cho phép người dùng xem và quản lý các schema trong cơ sở dữ liệu Redshift.
* Hỗ trợ nhiều loại câu lệnh: Dbeaver hỗ trợ nhiều loại câu lệnh SQL, cho phép người dùng thực hiện các truy vấn phức tạp trên cơ sở dữ liệu Redshift.
* Hỗ trợ đa kết nối: Dbeaver cho phép người dùng kết nối đến nhiều cơ sở dữ liệu Redshift khác nhau cùng một lúc, giúp tăng hiệu quả quản lý và sử dụng dữ liệu.
* Miễn phí và mã nguồn mở: Dbeaver là một công cụ mã nguồn mở và miễn phí, giúp giảm chi phí cho doanh nghiệp và cho phép người dùng tùy chỉnh và phát triển thêm các tính năng mới nếu cần thiết.

Qua đó, có thể thấy Dbeaver là một công cụ quản lý cơ sở dữ liệu mạnh mẽ và đáng tin cậy, cung cấp nhiều tính năng hữu ích cho việc kết nối và quản lý cơ sở dữ liệu Redshift của AWS.

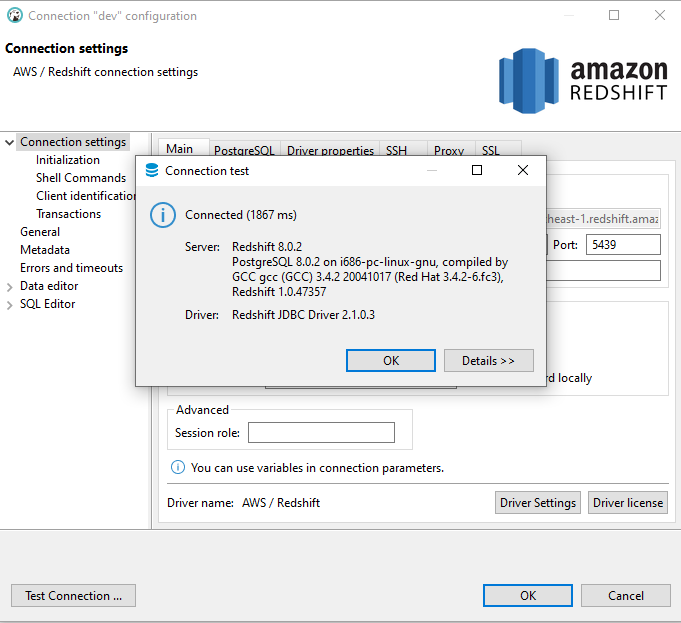
Để làm được điều đó, ta cần:

* Cài đặt JDBC Driver trong DBeaver: Trong DBeaver, chọn Drivers trong menu chính và tạo một driver mới. Sau đó, chọn file jar của JDBC driver vừa tải xuống để cài đặt driver.
* Tạo kết nối Redshift trong DBeaver: Trong DBeaver, chọn Database -> New Connection để tạo một kết nối mới. Chọn driver Redshift và nhập các thông tin kết nối, bao gồm tên máy chủ, cổng kết nối, tên cơ sở dữ liệu, tên đăng nhập và mật khẩu.



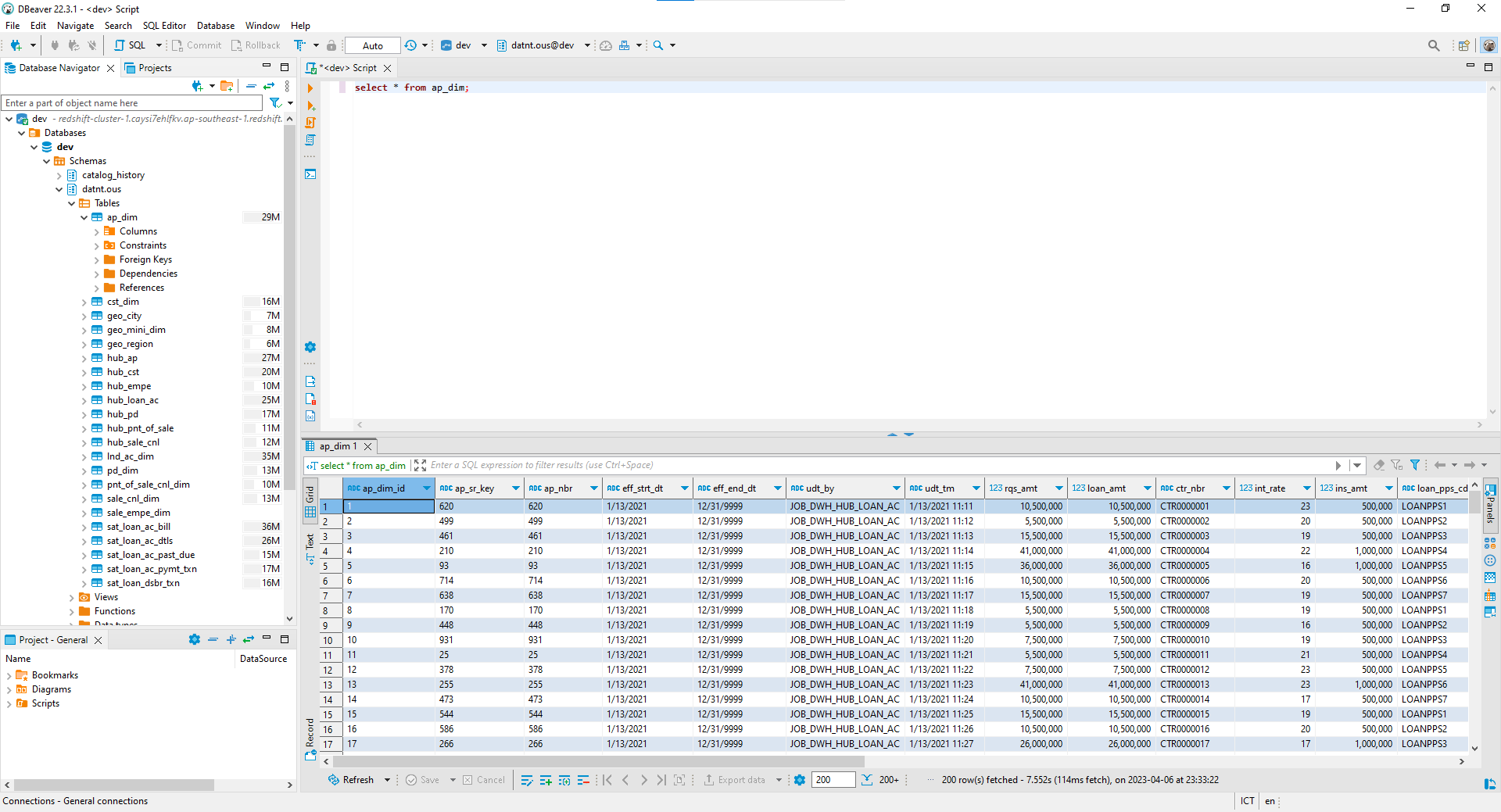
*Hình 2.11. Kết nối DBeaver với Redshift*

* Kiểm tra kết nối: Sau khi cấu hình các thông tin kết nối, Ta có thể kiểm tra kết nối bằng cách bấm nút Test Connection. Nếu kết nối thành công, Ta có thể sử dụng DBeaver để truy cập và quản lý cơ sở dữ liệu Redshift của mình.



*Hình 2.12. Test Connection Redshift trong DBeaver*

Sau khi đã kết nối thành công, chúng ta đã có một môi trường hỗ trợ querry câu lệnh SQL một cách nhanh hơn và quản lí dễ dàng hơn, giúp thực hiện những bước ETL một cách hiệu quả và nâng cao hiệu suất.



*Hình 2.13. Giao diện DBeaver sau khi kết nối Redshift*

## Tổng quan về Tableau

Tableau là một phần mềm phân tích dữ liệu và trực quan hóa dữ liệu được sử dụng rộng rãi trong nhiều lĩnh vực như kinh doanh, tài chính, giáo dục, y tế và chính phủ. Nó cho phép người dùng kết nối, tạo và chia sẻ các báo cáo, biểu đồ và dashboard để giúp họ hiểu và quản lý dữ liệu của mình một cách nhanh chóng và dễ dàng.

Tableau hỗ trợ kết nối với các nguồn dữ liệu đa dạng như Excel, CSV, SQL Server, Oracle, MySQL, Amazon Redshift, và nhiều hơn nữa. Nó cho phép người dùng thực hiện các phép tính toán, phân tích và trực quan hóa dữ liệu thông qua giao diện đồ họa dễ sử dụng.

Tableau cũng cung cấp các tính năng phân tích dữ liệu mạnh mẽ như trích xuất dữ liệu, phân tích định tính và định lượng, phân tích chuỗi thời gian, phân tích tương quan và phân tích đám mây từ khóa. Các kết quả phân tích có thể được trực quan hóa thông qua các biểu đồ như đường, cột, tròn, địa điểm, heatmap, bubble chart, tree map, scatter plot, và nhiều loại biểu đồ khác.

Tableau có nhiều tính năng và ưu điểm, bao gồm:

* Kết nối dữ liệu đa nguồn: Tableau cho phép kết nối dữ liệu từ nhiều nguồn khác nhau, bao gồm các tệp dữ liệu địa phương, các cơ sở dữ liệu trực tuyến, các kho dữ liệu và các ứng dụng SaaS.
* Khả năng thực hiện truy vấn dữ liệu trực tiếp: Tableau cho phép người dùng truy vấn và trực tiếp truy cập vào dữ liệu mà không cần phải sao chép hoặc di chuyển dữ liệu đến một nơi khác.
* Tính linh hoạt: Tableau cho phép người dùng tạo và tùy chỉnh các bảng điều khiển, biểu đồ và bản đồ để hiển thị dữ liệu theo cách phù hợp với nhu cầu và mục đích sử dụng.
* Khả năng tương tác và chia sẻ: Tableau cho phép người dùng tương tác với các bảng điều khiển và dữ liệu, đồng thời chia sẻ chúng với người dùng khác trong tổ chức hoặc bên ngoài tổ chức.
* Hỗ trợ đa nền tảng: Tableau có thể hoạt động trên nhiều nền tảng khác nhau, bao gồm Windows, Mac OS và các thiết bị di động.

Với các tính năng và ưu điểm trên, Tableau đã trở thành một trong những công cụ phân tích dữ liệu phổ biến nhất trên thị trường. Nó được sử dụng trong nhiều lĩnh vực, bao gồm tài chính, bán lẻ, y tế, giáo dục và chính phủ.

Ví dụ thực tế về việc sử dụng Tableau là trong việc phân tích dữ liệu bán lẻ để hiểu và dự đoán xu hướng mua sắm của khách hàng. Tableau cho phép người dùng kết nối với các nguồn dữ liệu bán lẻ như POS, CRM và trang web, và tạo các biểu đồ và dashboard để hiểu các hành vi mua sắm của khách hàng và dự đoán xu hướng mua sắm trong tương lai.

Sau khi đã chuẩn bị môi trường AWS và trải qua các bước ETL, để áp dụng Tableau phân tích dữ liệu ngân hàng, ta cần làm theo các bước như sau:

* Chuẩn bị dữ liệu: Để sử dụng Tableau, Ta cần phải chuẩn bị dữ liệu trước đó. Điều này bao gồm tìm kiếm và thu thập dữ liệu từ nhiều nguồn khác nhau, xử lý và làm sạch dữ liệu để sử dụng được trong Tableau.
* Kết nối dữ liệu: Sau khi chuẩn bị dữ liệu, Ta có thể kết nối dữ liệu vào Tableau. Tableau hỗ trợ kết nối dữ liệu từ nhiều nguồn khác nhau, bao gồm các cơ sở dữ liệu quan hệ, cơ sở dữ liệu không quan hệ, tệp tin văn bản và các dịch vụ đám mây như Amazon Redshift, Google BigQuery và Microsoft Azure.
* Xây dựng bảng tin: Sau khi kết nối dữ liệu vào Tableau, Ta có thể xây dựng các bảng tin để phân tích dữ liệu. Bảng tin là một màn hình dữ liệu trực quan, cho phép Ta tạo các biểu đồ, đồ thị và bản đồ để hiển thị dữ liệu.
* Tạo các biểu đồ và đồ thị: Với bảng tin, Ta có thể tạo các biểu đồ và đồ thị để hiển thị dữ liệu của mình. Tableau cung cấp nhiều loại biểu đồ và đồ thị khác nhau, bao gồm biểu đồ cột, biểu đồ dạng tròn, biểu đồ đường, biểu đồ phân tích thời gian, đồ thị bức xạ và nhiều hơn nữa.
* Tạo bản đồ: Tableau cung cấp tính năng tạo bản đồ để hiển thị dữ liệu dưới dạng bản đồ. Ta có thể tạo bản đồ dạng điểm, bản đồ dạng đường, bản đồ dạng khu vực và nhiều hơn nữa. Ta cũng có thể thêm các dữ liệu địa lý và thực hiện phân tích địa lý.
* Tạo báo cáo: Sau khi tạo các biểu đồ, đồ thị và bản đồ, Ta có thể tạo báo cáo để chia sẻ dữ liệu của mình với người dùng khác. Ta có thể xuất báo cáo dưới dạng tệp PDF, tệp Excel hoặc tệp định dạng khác.
* Chia sẻ bảng điều khiển: Cuối cùng, ta có thể chia sẻ bảng điều khiển của mình với đồng nghiệp hoặc khách hàng bằng cách xuất bảng điều khiển thành tệp hoặc tải lên vào Tableau Server để chia sẻ trực tiếp.

## Tóm tắt

Chương II tập trung vào việc xây dựng hệ thống Data Warehouse cho sàn thương mại điện tử Steam và tích hợp đa nguồn dữ liệu. Chương bắt đầu với một tổng quan về hệ thống kho dữ liệu của Steam, giới thiệu về cách Steam tổ chức và quản lý dữ liệu.

Tiếp theo, chương đề cập đến mục tiêu của hệ thống dữ liệu đa chiều được áp dụng trong sàn thương mại điện tử Steam, nhấn mạnh sự quan trọng của việc tích hợp nhiều nguồn dữ liệu để phân tích và đưa ra quyết định thông minh.

Chương tiếp tục với một tổng quan về Amazon Web Services (AWS), một nền tảng điện toán đám mây phổ biến và mạnh mẽ. Nó giới thiệu các dịch vụ lưu trữ và tính toán quan trọng trong AWS như EC2, S3, EBS, Glacier, RDS, Lambda, Redshift, Glue và Autoscaling. Sau đó, chương giải thích cách triển khai AWS cho mô hình datavault, tận dụng các dịch vụ của AWS để xây dựng một hệ thống Data Warehouse hiệu quả và linh hoạt. Chương cũng giới thiệu công cụ hỗ trợ ETL - Dbeaver, một công cụ phổ biến để thực hiện quá trình trích xuất, biến đổi và tải dữ liệu trong quá trình ETL.

Cuối cùng, chương kết thúc với một tổng quan về Tableau, một công cụ phân tích dữ liệu mạnh mẽ và trực quan. Nó giới thiệu khả năng của Tableau trong việc hiển thị và trực quan hóa dữ liệu từ hệ thống Data Warehouse.

# CHƯƠNG III: XÂY DỰNG HỆ THỐNG BÁO CÁO TỰ ĐỘNG TÍCH HỢP ĐA NGUỒN

1. **Mục tiêu của báo cáo Tableau**

Tableau là một công cụ phân tích dữ liệu mạnh mẽ cho phép người dùng trực quan hóa dữ liệu và tạo ra các báo cáo động và tương tác. Tuy nhiên, trước khi bắt đầu tạo báo cáo Tableau, Ta cần phải xác định mục tiêu của báo cáo. Điều này sẽ giúp Ta tập trung vào các thông tin quan trọng và đảm bảo rằng báo cáo của Ta mang lại giá trị cho người sử dụng.

Để có một bản báo cáo tốt nhất, ta cần xác định những mục tiêu sau:

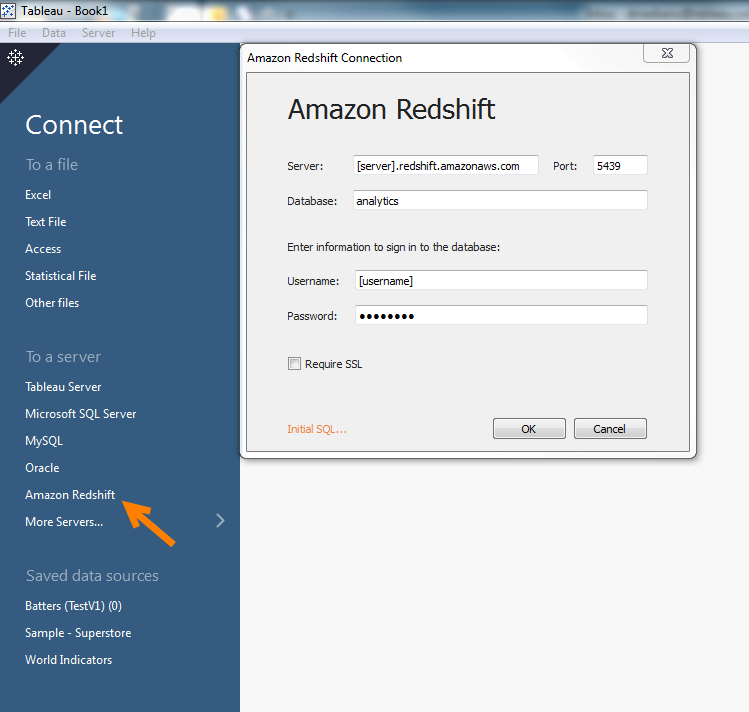
* Xác định đối tượng sử dụng báo cáo: Ta cần biết ai là người sử dụng cuối cùng của báo cáo để đảm bảo rằng báo cáo đáp ứng được nhu cầu và mục đích của họ.
* Xác định mục đích của báo cáo: Ta cần phải biết mục đích chính của báo cáo, ví dụ như tìm kiếm thông tin, đưa ra quyết định hoặc giám sát hiệu suất.
* Xác định dữ liệu cần thiết: Ta cần phải xác định dữ liệu cần thiết để tạo ra báo cáo và đảm bảo rằng dữ liệu này có sẵn và đầy đủ.
* Xác định độ chi tiết cần thiết: Ta cần phải xác định độ chi tiết của báo cáo, bao gồm các loại biểu đồ, bảng, hay đồ thị, và cách thức trình bày thông tin.
* Xác định thời gian cập nhật: Ta cần phải xác định tần suất cập nhật cho báo cáo, đảm bảo rằng thông tin trong báo cáo là chính xác và cập nhật.

Sau khi xác định được mục tiêu báo cáo, Ta có thể tiến hành tạo báo cáo trên Tableau và tinh chỉnh để đảm bảo rằng báo cáo phù hợp với nhu cầu của người sử dụng cuối cùng.

1. **Thu thập và chuẩn bị dữ liệu**

Thu thập và chuẩn bị dữ liệu là một bước quan trọng trong quá trình tạo báo cáo Tableau. Dữ liệu phải được thu thập từ các nguồn khác nhau, được xử lý, làm sạch và chuyển đổi thành dữ liệu có thể sử dụng được trong Tableau. Sau đây là một số bước cơ bản để thu thập và chuẩn bị dữ liệu cho Tableau:

* Xác định nguồn dữ liệu: Xác định các nguồn dữ liệu cần sử dụng để tạo báo cáo, bao gồm các file Excel, cơ sở dữ liệu hoặc các hệ thống CRM, ERP và các ứng dụng khác.
* Thu thập dữ liệu: Thu thập dữ liệu từ các nguồn được xác định ở bước trước. Dữ liệu có thể được thu thập bằng cách sử dụng các công cụ ETL (Extract, Transform, Load) hoặc các phương pháp khác tùy thuộc vào nguồn dữ liệu.
* Xử lý dữ liệu: Xử lý dữ liệu bao gồm loại bỏ các giá trị trùng lặp, các giá trị không hợp lệ và các giá trị thiếu. Điều này đảm bảo rằng dữ liệu được sử dụng trong báo cáo là chính xác và đáng tin cậy.
* Chuyển đổi dữ liệu: Chuyển đổi dữ liệu để phù hợp với cấu trúc dữ liệu của Tableau. Điều này có thể bao gồm thay đổi định dạng ngày tháng, tách các trường dữ liệu hoặc kết hợp các trường dữ liệu.
* Lưu trữ dữ liệu: Lưu trữ dữ liệu trong một nơi có thể truy cập được bởi Tableau, bao gồm các cơ sở dữ liệu hoặc các tệp dữ liệu được lưu trữ trên đĩa cứng hoặc trên các dịch vụ đám mây như Amazon S3 hoặc Google Cloud Storage.
* Kết nối dữ liệu với Tableau: Kết nối dữ liệu được chuẩn bị ở bước trước với Tableau. Điều này cho phép tạo báo cáo và thực hiện phân tích trực quan trên dữ liệu.



*Hình 4.1. Giao diện Tableau khi kết nối Redshift*

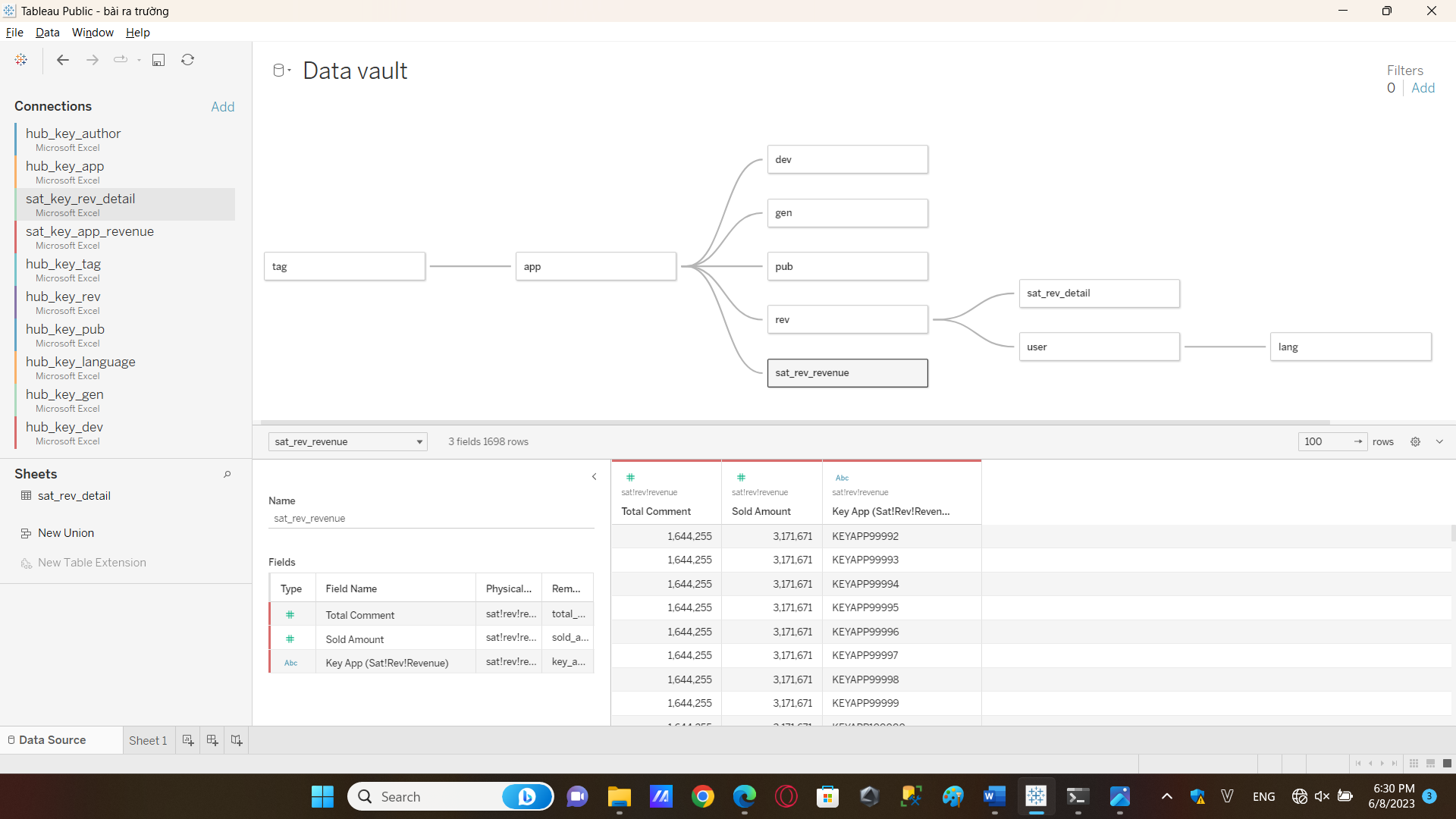
Quá trình thu thập và chuẩn bị dữ liệu là một bước quan trọng trong việc tạo ra báo cáo Tableau chính xác và có giá trị. Việc thu thập và chuẩn bị dữ liệu tốt sẽ đảm bảo rằng báo cáo được tạo ra từ Tableau sẽ truyền đạt được đầy đủ ý nghĩa của dữ liệu nhất.

1. **Thiết kế báo cáo**

Thiết kế báo cáo Tableau là quá trình tạo ra một báo cáo tương tác và trực quan trên Tableau để hiển thị thông tin dữ liệu theo các góc nhìn khác nhau. Thiết kế báo cáo Tableau yêu cầu phải có một kế hoạch cụ thể về việc hiển thị dữ liệu và đảm bảo rằng báo cáo đó đáp ứng được mục tiêu của người dùng.

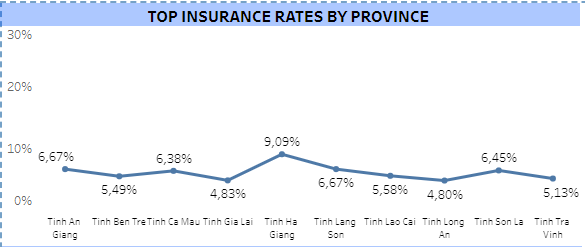
Dưới đây là một số bước cơ bản để thiết kế báo cáo Tableau:

* Xác định yêu cầu báo cáo: Các yêu cầu báo cáo phải được xác định rõ ràng trước khi bắt đầu thiết kế báo cáo Tableau. Điều này đảm bảo rằng báo cáo được tập trung vào việc hiển thị thông tin dữ liệu quan trọng và phù hợp với mục đích sử dụng.
* Lựa chọn bảng dữ liệu và kết nối: Sau khi xác định yêu cầu báo cáo, Ta phải chọn bảng dữ liệu phù hợp và kết nối nó với Tableau. Tableau hỗ trợ nhiều loại định dạng dữ liệu khác nhau, bao gồm cả các nguồn dữ liệu trong môi trường đám mây và các cơ sở dữ liệu quan hệ.

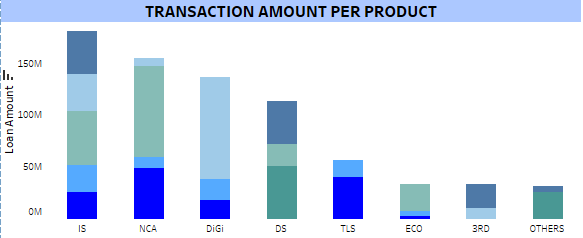


*Hình 4.2. Dữ liệu từ Redshift khi kết nối DBeaver*

* Xác định các chỉ số và số liệu cần thiết: Trước khi bắt đầu thiết kế báo cáo, Ta phải xác định các chỉ số và số liệu cần thiết để hiển thị trong báo cáo.
* Lựa chọn loại biểu đồ: Tableau cung cấp nhiều loại biểu đồ khác nhau để hiển thị thông tin dữ liệu. Ta phải chọn loại biểu đồ phù hợp với dữ liệu và yêu cầu của báo cáo.
* Thiết kế giao diện báo cáo: Ta phải thiết kế giao diện báo cáo sao cho nó trực quan và dễ sử dụng cho người dùng. Giao diện báo cáo bao gồm các tiêu đề, nội dung và các điều khiển để người dùng có thể tương tác với báo cáo.



*Hình 4.3. Line Chart*



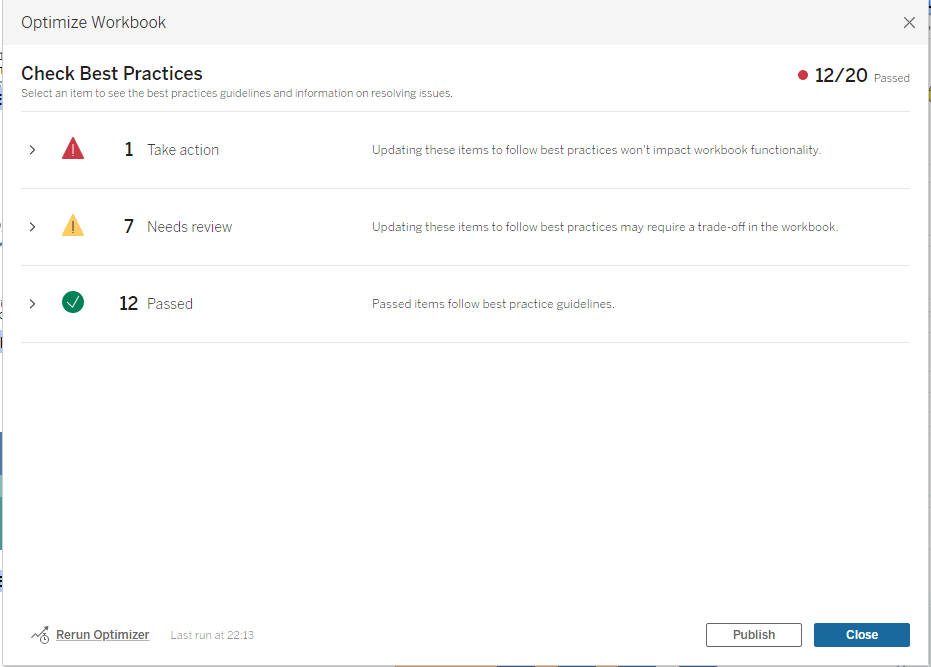
*Hình 4.4. Bar Chart*

1. **Đăng báo cáo và cập nhật báo cáo**

Sau khi đã thiết kế và tạo ra báo cáo Tableau, ta cần đăng báo cáo và cập nhật để đảm bảo người dùng có thể truy cập và sử dụng thông tin trong báo cáo.

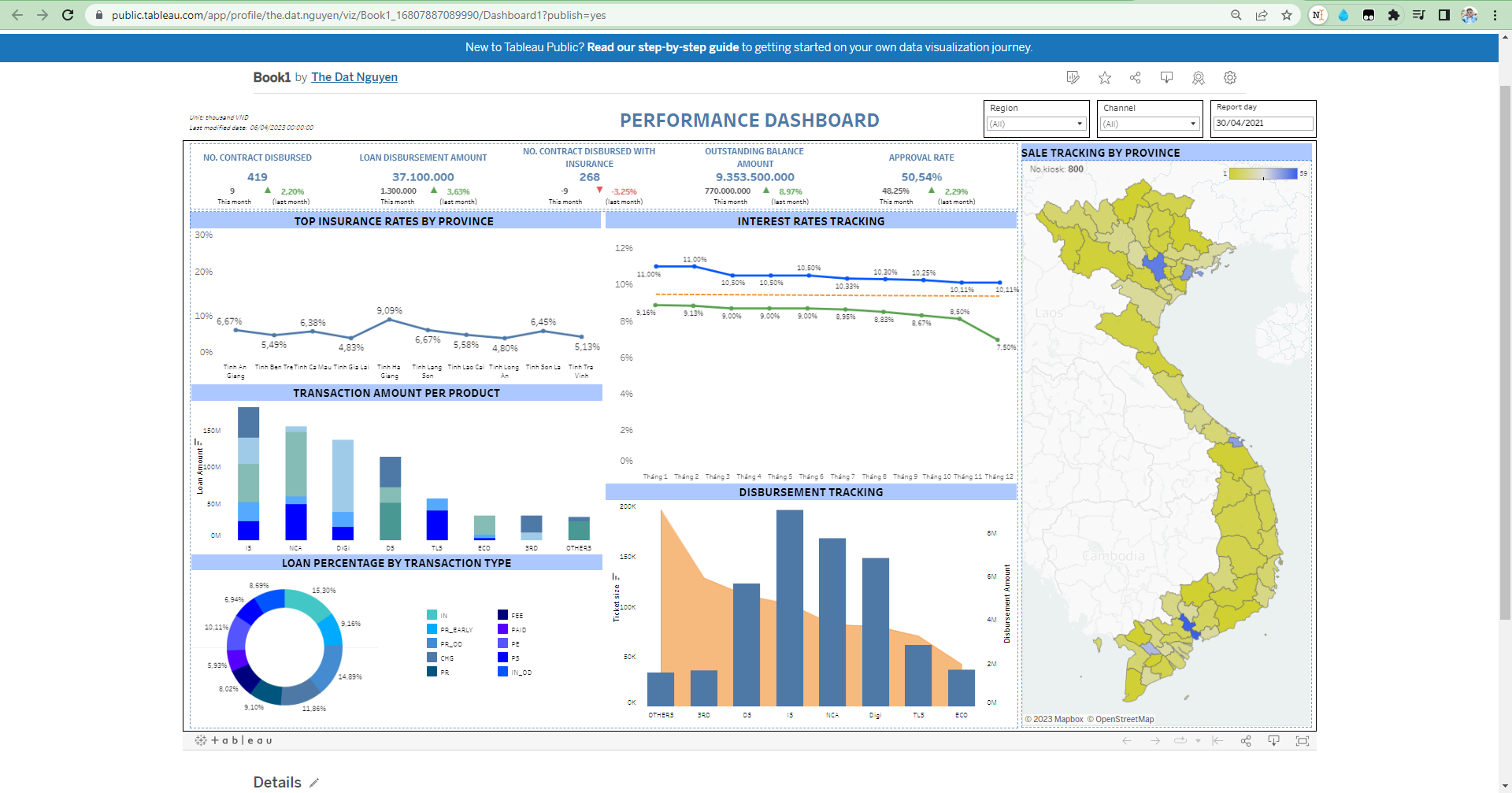
Đăng báo cáo Tableau có thể được thực hiện qua các phương pháp sau:

* Đăng trực tiếp lên máy chủ Tableau Server: Nếu tổ chức của Ta đã triển khai Tableau Server, Ta có thể đăng báo cáo lên máy chủ này để cho phép người dùng truy cập và tương tác với báo cáo. Ta có thể quản lý quyền truy cập của người dùng vào báo cáo thông qua máy chủ Tableau Server.



*Hình 4.5. Upload báo cáo lên server*

* Chia sẻ qua liên kết trực tuyến (URL): Ta có thể tạo liên kết đến báo cáo và chia sẻ liên kết đó với người dùng. Khi người dùng nhấp vào liên kết đó, báo cáo sẽ được hiển thị trên trình duyệt của họ.



*Hình 4.6. Báo cáo sau khi lên Server*

* Xuất báo cáo dưới dạng file: Tableau cho phép xuất báo cáo dưới dạng file PDF, Excel, hoặc image. Ta có thể gửi file báo cáo này cho người dùng hoặc đăng tải lên trang web của tổ chức.

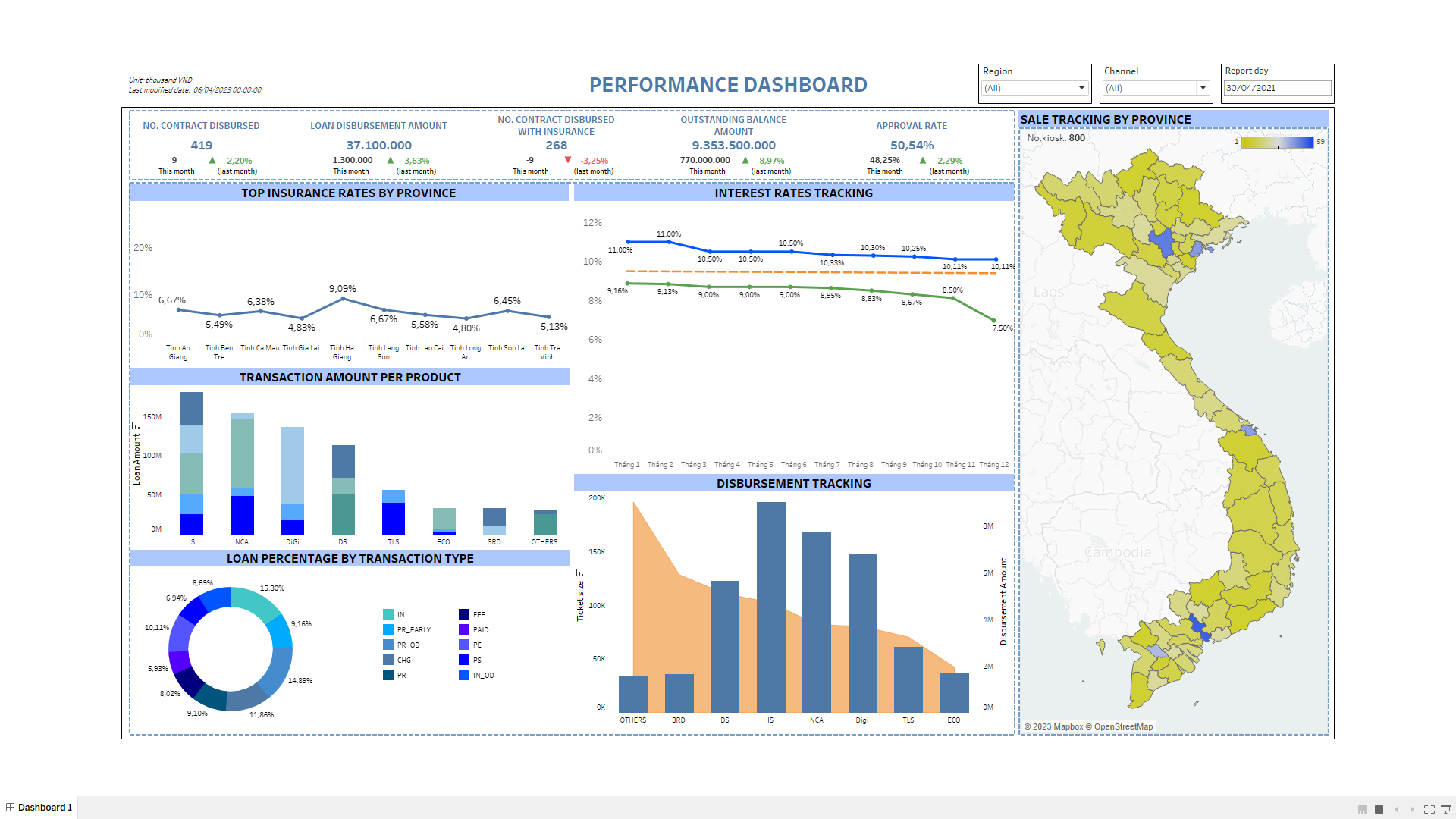
Ngoài ra, khi thông tin trong báo cáo Tableau thay đổi, Ta cần cập nhật báo cáo để đảm bảo rằng người dùng sử dụng thông tin mới nhất. Cập nhật báo cáo Tableau có thể được thực hiện qua các bước sau:

* Cập nhật nguồn dữ liệu: Nếu nguồn dữ liệu đã thay đổi, Ta cần phải cập nhật nguồn dữ liệu đó để đảm bảo rằng thông tin trong báo cáo là chính xác.
* Cập nhật báo cáo: Sau khi cập nhật nguồn dữ liệu, Ta có thể mở báo cáo trong Tableau và cập nhật lại các biểu đồ, bảng dữ liệu, hoặc các phần khác của báo cáo để hiển thị thông tin mới nhất.
* Đăng báo cáo cập nhật: Sau khi cập nhật báo cáo, Ta cần đăng lại báo cáo để cho phép người dùng truy cập và sử dụng thông tin mới nhất trong báo cáo.

Ngoài ra, để đảm bảo tính đồng nhất và đáng tin cậy của dữ liệu, cần thường xuyên cập nhật báo cáo Tableau với các thông tin mới nhất. Việc cập nhật có thể thực hiện thông qua các cập nhật trực tuyến hoặc lên lịch thực hiện tự động vào một khoảng thời gian nhất định.

Nếu có nhiều người dùng sử dụng báo cáo, cần đảm bảo rằng tất cả các phiên bản của báo cáo đều được cập nhật để tránh tình trạng sử dụng thông tin lỗi thời.

Cuối cùng, khi đăng báo cáo Tableau, cần đảm bảo rằng nó được đưa ra một cách trực quan và dễ hiểu cho người dùng. Có thể tạo ra một bản tóm tắt, một bản trình bày PowerPoint hoặc một bản thuyết trình để giải thích và trình bày kết quả và những phát hiện quan trọng trong báo cáo.



*Hình 4.7. Báo cáo hoàn chỉnh sau khi upload*

Nếu muốn chia sẻ báo cáo với người dùng bên ngoài, Tableau cũng cung cấp các tùy chọn để chia sẻ báo cáo, cho phép người dùng truy cập vào báo cáo thông qua một liên kết được chia sẻ hoặc nhúng báo cáo vào trang web hoặc ứng dụng của họ.

Tóm lại, việc đăng báo cáo và cập nhật báo cáo Tableau là một phần quan trọng của quá trình phân tích dữ liệu và cung cấp thông tin cho người dùng. Việc đảm bảo tính đúng đắn và đồng nhất của dữ liệu, thiết kế báo cáo trực quan và dễ hiểu, cùng với việc cập nhật báo cáo định kỳ, sẽ giúp đảm bảo rằng báo cáo Tableau luôn được sử dụng hiệu quả và đóng góp vào sự phát triển của doanh nghiệp.

1. **Phân tích biểu mẫu báo cáo**

****

No. Contract Disbursed: Số lượng Hợp đồng được giải ngân.

Giải ngân là quá trình chuyển đổi số tiền trong hợp đồng vay thành tiền mặt, đưa vào sử dụng để thực hiện mục đích mà hợp đồng vay đã cam kết. Tức là, sau khi ký kết hợp đồng vay, ngân hàng sẽ chuyển khoản số tiền vay vào tài khoản của khách hàng để khách hàng có thể sử dụng số tiền này cho mục đích đã cam kết.

Ý nghĩa: Tháng này đã giải ngân được 419 hợp đồng, tăng 9 hợp đồng và 2.2% so với tháng trước.

****

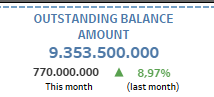
Loan Disbursement Amount: Số tiền đã giải ngân (Triệu đồng)

Ý nghĩa: Tháng này đã giải ngân 37 nghìn tỷ và tăng 1 nghìn 3 trăm tỷ và giảm 3.63% so với tháng trước.

****

No. Contract Disbursed with insurance: Số lượng Hợp đồng được giải ngân có đóng bảo hiểm.

Ý nghĩa: Tháng này đã giải ngân được 268 hợp đồng có đóng bảo hiểm, giảm 9 hợp đồng và 3.25% so với tháng trước.

****

Outstanding Balance Amount: Số tiền dư nợ (Triệu đồng)

Dư nợ là số tiền chưa trả hết của khoản vay hoặc nợ. Nó thường được sử dụng để đo lượng tiền vay hoặc nợ hiện tại của một cá nhân hoặc tổ chức tại một thời điểm nhất định. Khi vay tiền hoặc nhận nợ, người vay hoặc người được nợ sẽ phải trả lại số tiền ban đầu cùng với các khoản lãi phát sinh. Dư nợ là số tiền chưa trả cho đến thời điểm đánh giá và được sử dụng để tính toán các khoản lãi tiếp theo.

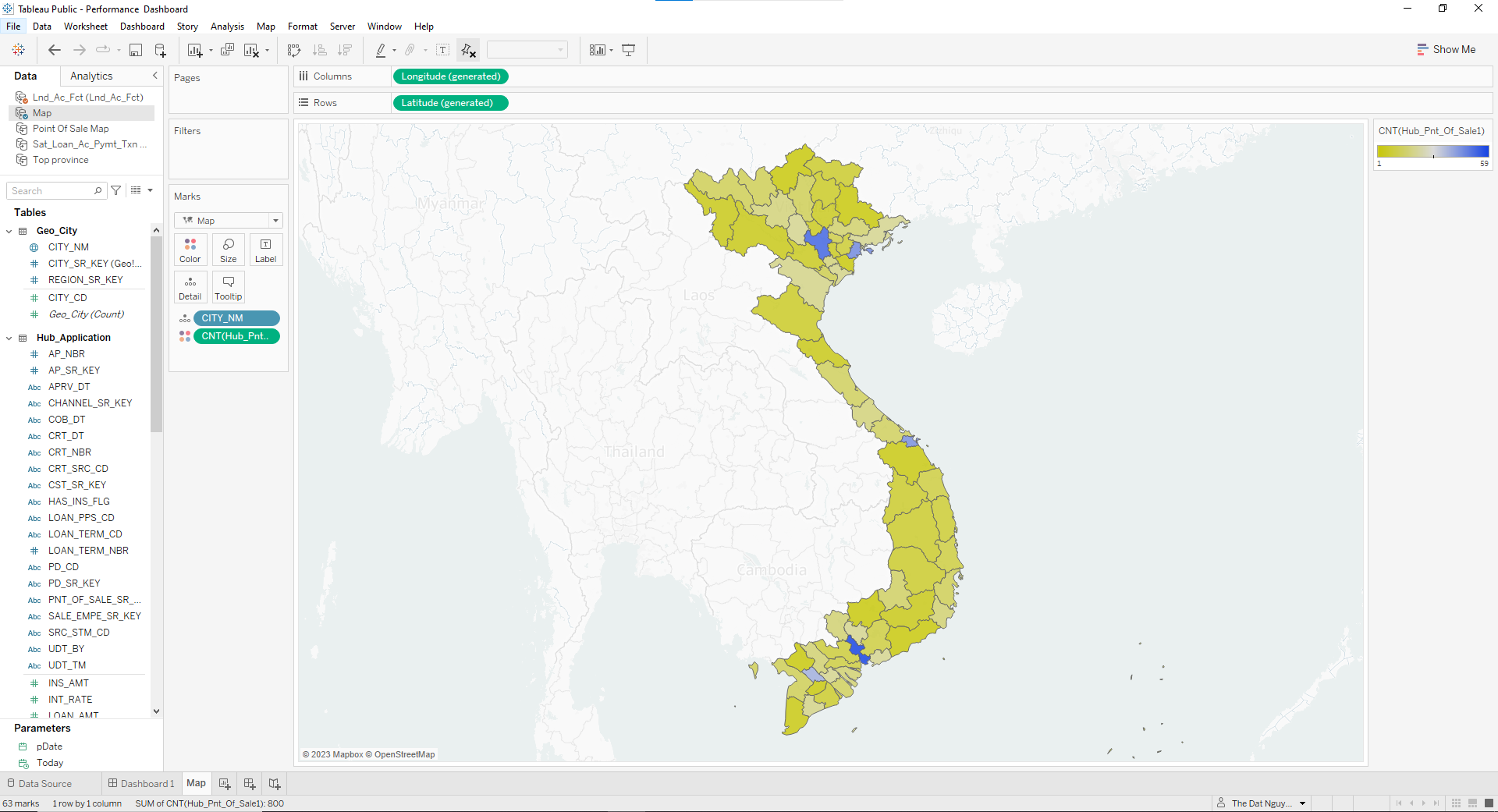
Ý nghĩa: Từ thời điểm thu thập dữ liệu, hiện còn hơn 9 nghìn tỷ chưa được thu hồi lại, tăng 770 tỷ và 8.97% so với tháng trước (tháng này cho vay nhiều hơn nhưng khoản cũ hiện chưa đòi được quá nhiều)

****

Approval Rate: Tỷ lệ phê duyệt

Tỷ lệ phê duyệt hồ sơ là tỷ lệ số hồ sơ được phê duyệt so với tổng số hồ sơ đã được thẩm định. Nó thường được sử dụng để đánh giá hiệu suất của quy trình thẩm định hồ sơ và đo lường mức độ chính xác và đáng tin cậy của quyết định phê duyệt hồ sơ. Tỷ lệ phê duyệt hồ sơ sau thẩm định là một trong những chỉ số quan trọng trong việc đánh giá và cải tiến quy trình thẩm định hồ sơ, giúp tăng cường hiệu quả và nâng cao chất lượng dịch vụ.

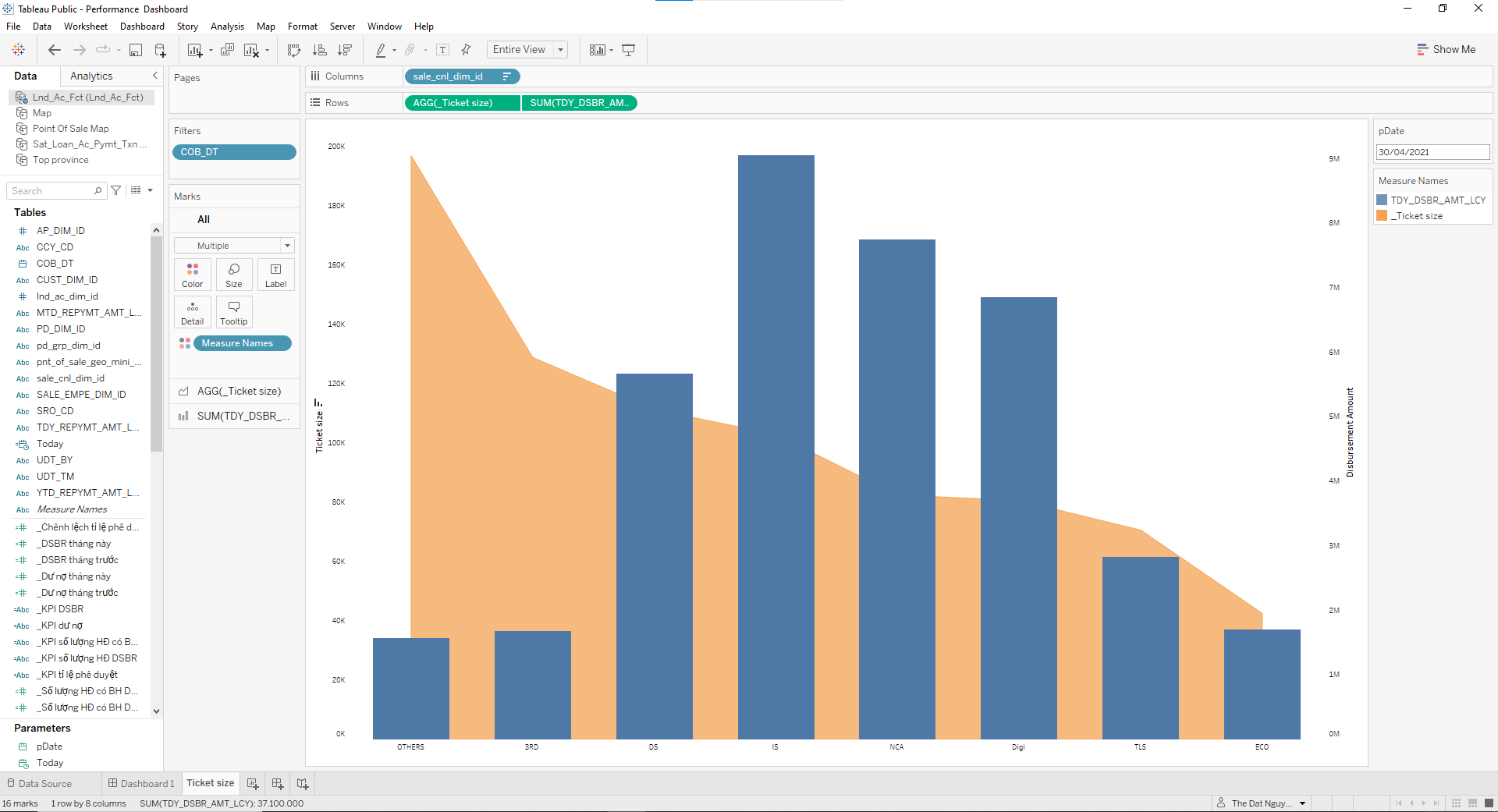
Ý nghĩa: Trong tháng hiện tại, có 50,42% tổng số lượng hồ sơ được phê duyệt để giải ngân, con số này của tháng trước là 48,25% và tăng với tỉ lệ là 2,29%.

****

Sale tracking by province: Theo dõi số lượng bán hàng theo tỉnh thành

Theo dõi số lượng bán hàng theo tỉnh thành là quá trình theo dõi và thu thập thông tin về số lượng sản phẩm được bán ra tại mỗi tỉnh/thành phố. Thông tin này có thể được sử dụng để đánh giá hiệu quả kinh doanh của công ty, phân tích thị trường và đưa ra các quyết định chiến lược cho hoạt động kinh doanh. Việc theo dõi số lượng bán hàng theo tỉnh thành có thể được thực hiện bằng cách sử dụng các công cụ quản lý dữ liệu và phần mềm phân tích dữ liệu để tự động hóa quá trình thu thập và xử lý dữ liệu.

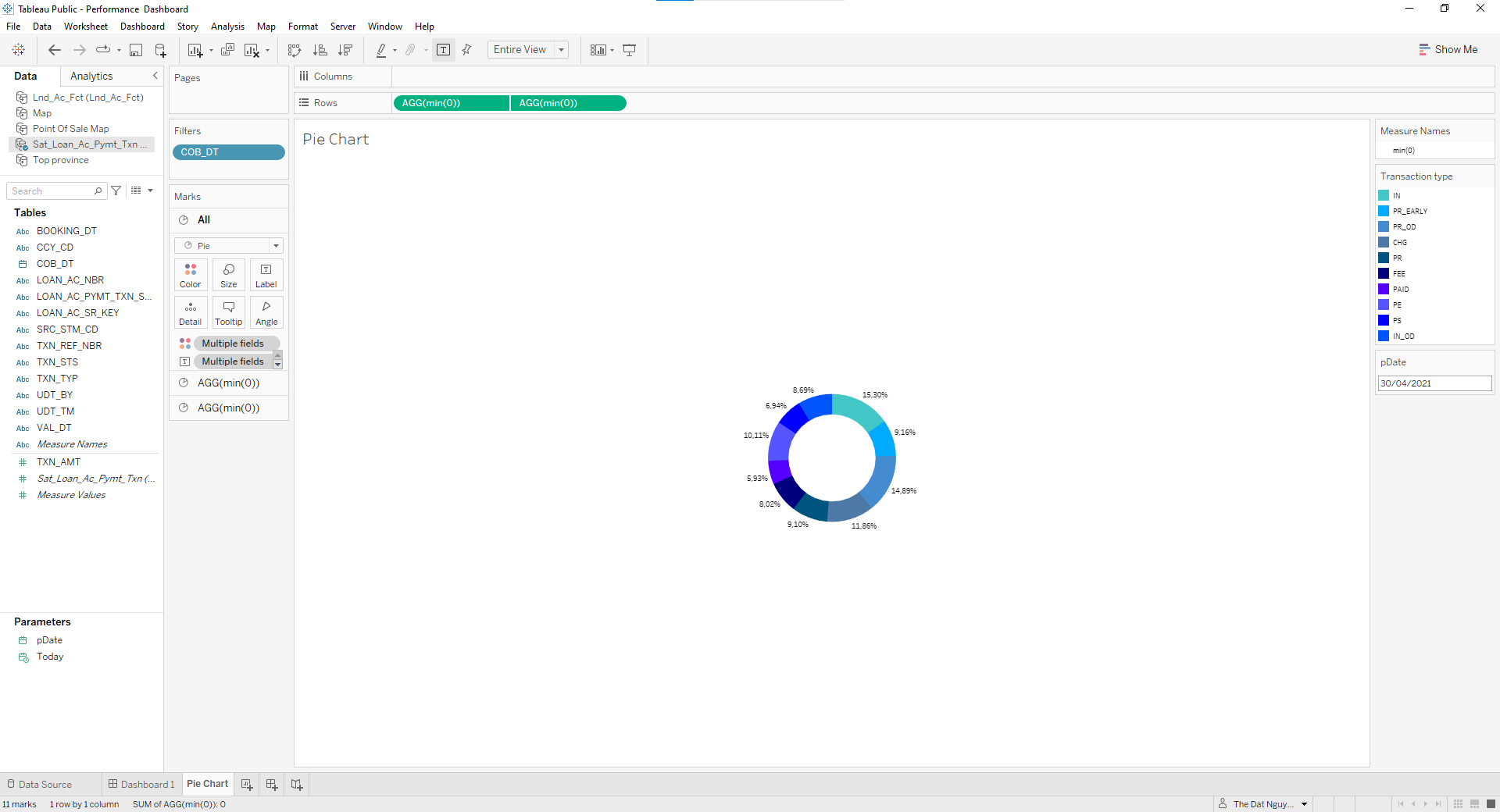
Ý nghĩa: Có tổng 800 địa điểm bán hàng trên toàn quốc, và tập trung nhiều ở những thành phố lớn trên cả nước.



Disbursement tracking: Theo dõi dư nợ

Ticket size là một thuật ngữ trong lĩnh vực tài chính, được sử dụng để chỉ tổng giá trị của một giao dịch hoặc một loạt các giao dịch. Nó thường được sử dụng trong ngành bán lẻ và ngành tài chính để đo lường giá trị trung bình của một đơn hàng hoặc một giao dịch. Ví dụ, nếu một cửa hàng bán hàng điện tử có tổng doanh thu là 100.000 USD và số lượng đơn hàng là 500, thì ticket size của mỗi đơn hàng sẽ là 200 USD. Ticket size là một chỉ số quan trọng để đo lường hiệu quả của chiến lược bán hàng và tiếp thị.

Ý nghĩa: Tại ngày cụ thể, biểu đồ cho thấy 8 kênh bán cũng như dư nợ mỗi kênh bán, ngoài ra còn thể hiện Ticket size của mỗi kênh bán.



Loan percentage by transaction type: Tỉ lệ cho vay theo loại giao dịch

Tỉ lệ cho vay theo loại giao dịch là tỷ lệ phần trăm của số tiền cho vay được phê duyệt trong một loại giao dịch cụ thể (ví dụ: cho vay mua ô tô, cho vay mua nhà, cho vay tiêu dùng). Nó cho biết tỷ lệ của số tiền được cho vay so với tổng số tiền đăng ký vay trong một loại giao dịch. Khi theo dõi tỷ lệ cho vay theo loại giao dịch, ngân hàng có thể đánh giá hiệu quả của các sản phẩm vay cũng như tính khả thi của chính sách cho vay của mình.

Trong đó:

PE - Phạt gốc quá hạn

IN - Trả nợ lãi

PS - Phạt lãi quá hạn

PR - Trả nợ gốc

PR\_OD - hạch toán gốc quá hạn

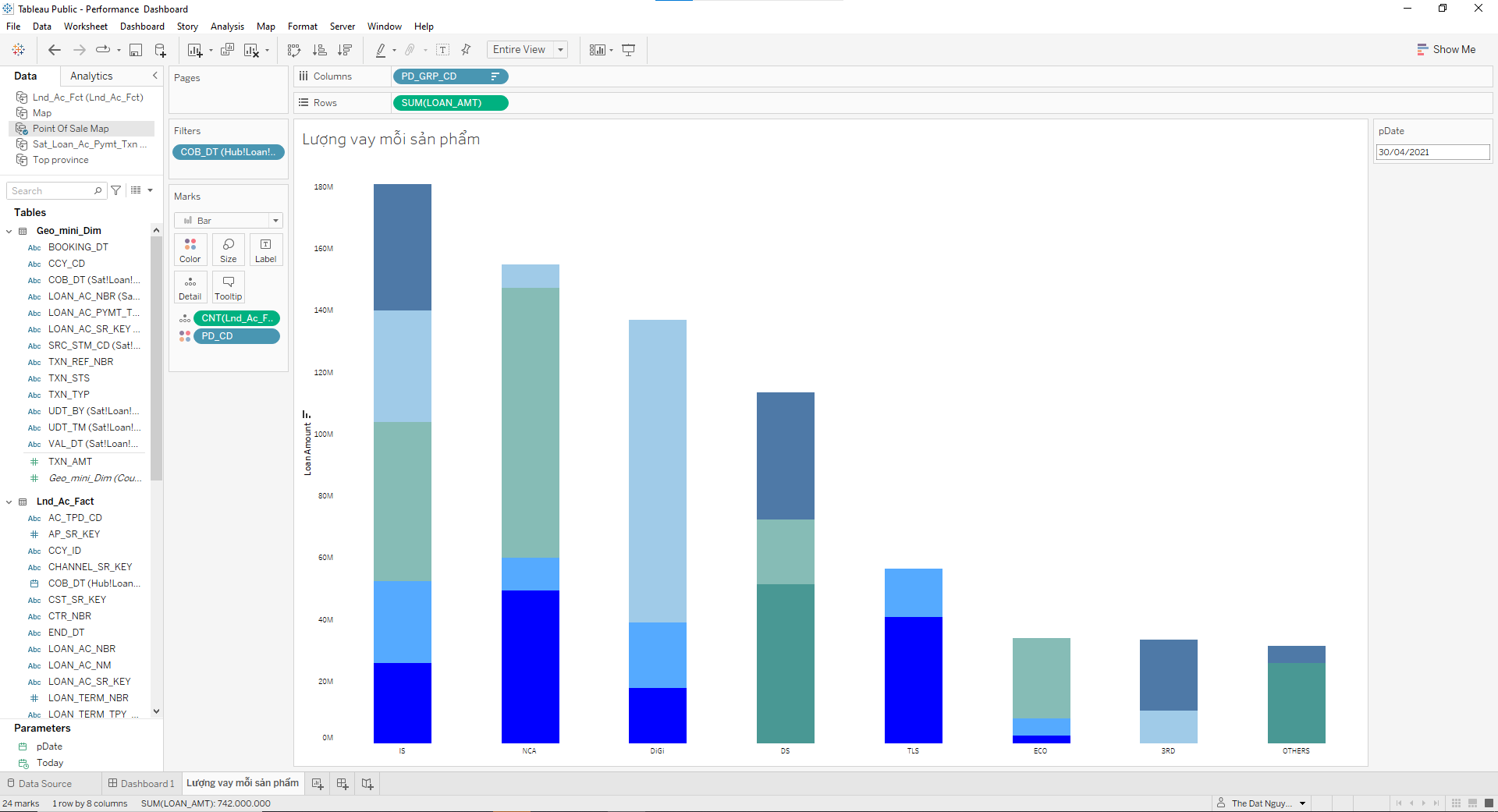
IN\_OD - hạch toán lãi quá hạn

PAID - trả tiền 1 cục

CHG - phí tất toán trước hạn của món

FEE - Phí thu hộ

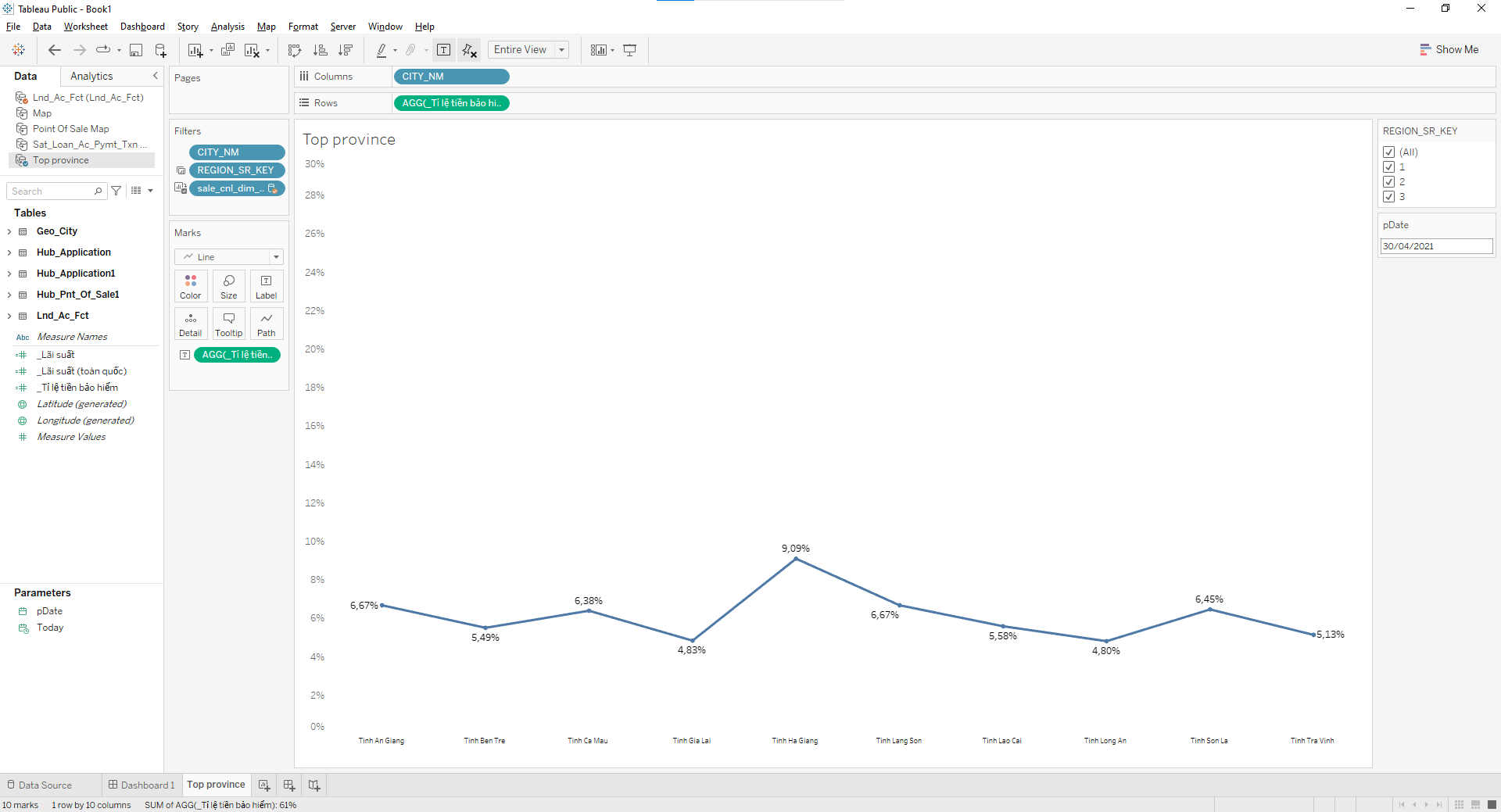
Ý nghĩa: Tại thời điểm cụ thể, biểu đồ thể hiện số lượng tiền cho vay theo tất cả 10 loại giao dịch.



Transaction amount per product: Lượng tiền giao dịch mỗi sản phẩm.

Lượng tiền giao dịch mỗi sản phẩm là số tiền bán hàng trung bình cho mỗi sản phẩm hoặc dịch vụ trong một giao dịch. Nó thường được tính bằng cách chia tổng số tiền giao dịch cho số lượng sản phẩm hoặc dịch vụ được bán trong giao dịch đó. Đây là một chỉ số quan trọng trong phân tích bán hàng, giúp đánh giá hiệu suất bán hàng và tính toán doanh thu trung bình cho mỗi sản phẩm hoặc dịch vụ.

Ý nghĩa: Tại thời điểm cụ thể, biểu đồ cho thấy lượng giao dịch mỗi kênh bán cũng như tỉ lệ lượng giao dịch của từng loại sản phẩm trên mỗi kênh bán.



Top insurance rates by province: Top những tỉnh thành có tỉ lệ lượng tiền dùng để mua bảo hiểm vay nhiều nhất.

Đây là danh sách các tỉnh/thành phố có tỷ lệ phí bảo hiểm cao nhất trong một thời điểm nhất định. Các tỷ lệ này có thể được tính dựa trên nhiều yếu tố như loại hợp đồng, độ tuổi của khách hàng, v.v. Thông tin này có thể được sử dụng để đưa ra quyết định về chiến lược kinh doanh và marketing của công ty bảo hiểm.

Ý nghĩa: Tại thời điểm cụ thể, biểu đồ cho thấy 10 tỉnh thành có tỉ lệ sử dụng tiền để mua bảo hiểm nhiều nhất. Với tỉnh Hà Giang có số tiền mua bảo hiểm tương ứng 9.09% so với tiền vay (tức là vay 100 triệu thì dùng hơn 9 triệu để làm bảo hiểm cho khoản vay).

<https://pickl.ai/blog/azure-data-engineer-jobs/>