TRƯỜNG ĐẠI HỌC KINH TẾ

**KHOA THỐNG KÊ – TIN HỌC**

A blue and white logo

Description automatically generated

**BÁO CÁO THỰC TẬP NGHỀ NGHIỆP**

**Xây dựng hệ thống xử lý và lưu trữ trên Microsoft Azure**

Sinh viên thực hiện : **Trần Nhật Hoàng**

Lớp : **46k21.2**

Đơn vị thực tập : **TMA Solution Bình Định**

Giảng viên hướng dẫn : **Ths. Cao Thị Nhâm**

\

***Đà Nẵng, 8/202******3***

MỤC LỤC

[NHẬN XÉT CỦA ĐƠN VỊ THỰC TẬP i](#_Toc154159934)

[DANH MỤC HÌNH ẢNH i](#_Toc154159935)

[DANH MỤC BẢNG BIỂU iii](#_Toc154159936)

[DANH MỤC CÁC TỪ VIẾT TẮT iv](#_Toc154159937)

[LỜI MỞ ĐẦU 1](#_Toc154159938)

[CHƯƠNG 1. GIỚI THIỆU CÔNG TY 2](#_Toc154159939)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 2](#_Toc154159940)

[2.1. Data warehouse ,Data lake, Lake House 2](#_Toc154159941)

[2.1.1. Data warehouse 2](#_Toc154159942)

[2.1.2. Data Lake 5](#_Toc154159943)

[2.1.3. LakeHouse 6](#_Toc154159944)

[2.2. Apache Spark 6](#_Toc154159945)

[2.3. Medallion Architecture and data Change Feed 7](#_Toc154159946)

[2.3.1. Bronze layer (raw data) 7](#_Toc154159947)

[2.3.2. Silver layer (cleansed and conformed data) 8](#_Toc154159948)

[2.3.3. Gold layer (curated business-level tables) 8](#_Toc154159949)

[2.4. Giới thiệu về Microsoft Azure 8](#_Toc154159950)

[2.4.1. Microsoft Azure là gì? 8](#_Toc154159951)

[2.4.2. Các dịch vụ Azure cung cấp 9](#_Toc154159952)

[2.5. Các dịch vụ demo 10](#_Toc154159953)

[2.5.1. Azure blob 10](#_Toc154159954)

[2.5.2. Key vault 12](#_Toc154159955)

[2.5.3. Azure DataBricks 12](#_Toc154159956)

[2.5.4. Data Factory 13](#_Toc154159957)

[CHƯƠNG 3. Tổng quan về hệ thống xử lý dữ liệu trên Microsoft azure 14](#_Toc154159958)

[3.1. Tổng quan về xây dựng hệ thống: 14](#_Toc154159959)

[3.2. Quy trình xử lý của hệ thống: 15](#_Toc154159960)

[3.2.1. Kích hoạt Trigger tự động: 15](#_Toc154159961)

[3.2.2. Xử lý dữ liệu qua các databricks trong pipeline: 15](#_Toc154159962)

[3.2.3. Kết thúc quy trình xử lý và lặp lại nếu người dùng tiếp tục tải dữ liệu lên 15](#_Toc154159963)

[3.2.4. Công nghệ được sử dụng 16](#_Toc154159964)

[3.3. Tính năng của lakehouse và Delta Lake trong quá trình xử lý dữ liệu 16](#_Toc154159965)

[3.4. Mô tả về bộ dữ liệu dùng để Demo cho hệ thống 17](#_Toc154159966)

[CHƯƠNG 4. Xây dựng hệ thống 19](#_Toc154159967)

[4.1. Quy trình xây dựng 19](#_Toc154159968)

[4.1.1. Tạo stogare account 19](#_Toc154159969)

[4.1.2. Tạo Key Vault 20](#_Toc154159970)

[4.1.3. Xử lý bằng Databricks 21](#_Toc154159971)

[4.1.4. Tạo SQL Server và database trên azure 26](#_Toc154159972)

[4.1.5. Sử Data Factory xây dựng các đường ống xử lý dữ liệu theo quy trình 26](#_Toc154159973)

[4.2. Các bước trong hàm xử lý dữ liệu của DataBricks 27](#_Toc154159974)

[4.2.1. Bảng customer 27](#_Toc154159975)

[4.2.2. Bảng Sale 27](#_Toc154159976)

[4.3. Kiểm tra và chạy thử hoạt động của hệ thống sau khi hoàn thành gồm trigger, pipeline và databricks 28](#_Toc154159977)

[4.3.1. Lần chạy đầu tiên với rawdata 28](#_Toc154159978)

[4.3.2. Lần chạy tiếp theo với các tệp data-log 30](#_Toc154159979)

[4.3.3. Cuối cùng kiểm tra dữ liệu sau khi được lưu vào cơ sở dữ liệu SQL server 32](#_Toc154159980)

[CHƯƠNG 5. TẠO DASHBOARD BẰNG POWER BI 33](#_Toc154159981)

[5.1. Tạo Dashboard bằng Power Bi 33](#_Toc154159982)

[5.1.1. Liên kết dữ liệu trên SQL Azure Database vào PowerBi Desktop 33](#_Toc154159983)

[5.2. Tạo Dashboard 34](#_Toc154159984)

[5.2.1. Dashboard thông tin khách hàng 35](#_Toc154159985)

[5.2.2. Dashboard về hành vi khách hàng 36](#_Toc154159986)

[TÀI LIỆU THAM KHẢO 37](#_Toc154159987)

[PHỤ LỤC 38](#_Toc154159988)

NHẬN XÉT CỦA ĐƠN VỊ THỰC TẬP

**A paper with a check box

Description automatically generated with medium confidence**

**A paper with text and a red stamp

Description automatically generated**

# DANH MỤC HÌNH ẢNH

[Hình 2.1 Kiến trúc WareHouse, Data lake ,Lake house 2](#_Toc154159777)

[Hình 2.2 Kiến trúc Medallion Architecture 6](#_Toc154159778)

[Hình 2.3 Blob storage 10](#_Toc154159779)

[Hình 3.1 Tổng quan về Pipeline 13](#_Toc154159780)

[Hình 4.1 Tổng quát về resource group trên Azure 18](#_Toc154159781)

[Hình 4.2 Storage-Level01 19](#_Toc154159782)

[Hình 4.3 Stogare-Level-2 19](#_Toc154159783)

[Hình 4.4 Tạo Key vault 19](#_Toc154159784)

[Hình 4.5 Tạo scope trên databricks 20](#_Toc154159785)

[Hình 4.6 Sử dụng Scope truy cập vào các Storage 20](#_Toc154159786)

[Hình 4.7 Tạo tham số đầu vào cho file dữ liệu 21](#_Toc154159787)

[Hình 4.8 Đọc dữ liệu lần đầu 21](#_Toc154159788)

[Hình 4.9 Đọc dữ liệu sau lần đầu tiên 21](#_Toc154159789)

[Hình 4.10 Lưu dữ liệu vào lớp Bronze 22](#_Toc154159790)

[Hình 4.11 Tạo Scope 23](#_Toc154159791)

[Hình 4.12 Sử dụng scope và keyvault truy cập vào Storage 23](#_Toc154159792)

[Hình 4.13 Lưu dữ liệu vào silver ở lần đầu 24](#_Toc154159793)

[Hình 4.14 Xây dựng luồng dữ liệu trên Data Factory 24](#_Toc154159794)

[Hình 4.15 Xây dựng luồng dữ liệu trên Data Factory 25](#_Toc154159795)

[Hình 4.16 Hoạt động của trigger Raw 27](#_Toc154159796)

[Hình 4.17 Pipeline chạy thành công 27](#_Toc154159797)

[Hình 4.18 Xem kết quả trả về ua DataBricks 28](#_Toc154159798)

[Hình 4.19 Kết quả Bronze\_Processing 29](#_Toc154159799)

[Hình 4.20 Kết quả Silver\_Processing 29](#_Toc154159800)

[Hình 4.21 Trigger-Log chạy thành công 29](#_Toc154159801)

[Hình 4.22 Pipeline do trigger-log kích hoạt chạy thành công 30](file:///E:\Users\Download\Project-ETL%20on%20Azure.docx.docx#_Toc154159802)

[Hình 4.23 Xem kết quả ở Bronze\_Processing 30](#_Toc154159803)

[Hình 4.24 Xem kết quả ở Silver\_Processing 31](#_Toc154159804)

[Hình 4.25 Kiểm tra dữ liệu nạp vào Database 31](#_Toc154159805)

[Hình 5.1 Cung cấp địa chỉ IP của máy cho networking server 32](#_Toc154159806)

[Hình 5.2 LIên kết dữ liệu DataBase với PowerBI 33](#_Toc154159807)

[Hình 5.3 Dashboard thông tin khách hàng 34](#_Toc154159808)

[Hình 5.4 Dashboard về hành vi khách hàng 35](#_Toc154159809)

# DANH MỤC BẢNG BIỂU

[Bảng 2.1 So sánh ETL và ELT 4](#_Toc154159767)

[Bảng 3.1 Thông tin về bộ dữ liệu 17](#_Toc154159768)

# DANH MỤC CÁC TỪ VIẾT TẮT

**ADF** : Azure Factory

**CdC** : Change Data Capture

# LỜI MỞ ĐẦU

1. **Mục tiêu nghiên cứu của đề tài**

* Đề tài này nghiên cứu về những kỹ năng và kiến thức cần có của Data Engineer
* Tìm hiểu những kiến thức để phát triển được hệ thống xử lý và lưu trữ dữ liệu
* Lên được kịch bản và xây dựng được về hệ thống xử lý và lưu trữ dữ liệu trên các nền tảng Cloud

1. **Nhiệm vụ của đề tài**

* Tìm hiểu những công việc cần làm vị trí Data Engineer
* Tìm hiểu và học hỏi những kiến thức kỹ năng cần có với vị trí Data Engineer
* Nghiên cứu về các kiến trúc dữ liệu, nền tảng lưu trữ dữ liệu trên Cloud
* Thực hiện xây dựng demo hệ thống xử lý và lưu trữ dữ liệu trên Microsoft Azure

1. **Kết cấu của đề tài**

Đề tài được tổ chức gồm phần mở đầu, 5 chương nội dung và phần kết luận.

* Mở đầu
* **Chương 1**: Tổng quan về công ty và vị trí thực tập
* **Chương 2**: Cơ sở lý thuyết
* **Chương 3**: Tổng quan về hệ thống xử lý dữ liệu trên Microsoft azure
* **Chương 4**: Xây dựng hệ thống
* **Chương 5**: Xây dựng DashBoard bằng Power Bi
* Kết luận và hướng phát triển

# GIỚI THIỆU CÔNG TY

# CƠ SỞ LÝ THUYẾT

## Data warehouse ,Data lake, Lake House

**A screenshot of a data lake

Description automatically generated**

Hình 2.1 Kiến trúc WareHouse, Data lake ,Lake house

### Data warehouse

#### Định nghĩa

Kho dữ liệu: là tập các phương pháp, kỹ thuật và công cụ có thể kết hợp lại để cung cấp thông tin cho người dùng dựa trên việc tích hợp dữ liệu từ nhiều nguồn, nhiều môi trường khác nhau

**Đặc tính ACID của DW: Đảm bảo tính toàn của dữ liệu**

***Atomicit*y** (Tính nguyên tử): Đặc tính này đảm bảo rằng một giao dịch được thực hiện hoàn toàn hoặc không được thực hiện. Trong DW, thay vì xử lý từng giao dịch riêng lẻ, dữ liệu được xử lý theo quá trình ETL (Extract, Transform, Load) hoặc ELT (Extract, Load, Transform) dưới dạng hàng loạt. Do đó, khái niệm atomicity không áp dụng trực tiếp cho DW.

***Consistency*** (Tính nhất quán): DW thường chứa dữ liệu từ nhiều nguồn khác nhau và được sử dụng cho mục đích phân tích kinh doanh. Trong quá trình ETL/ELT, dữ liệu từ nguồn được chuyển đổi và hợp nhất để đảm bảo tính nhất quán. Tuy nhiên, DW không đảm bảo tính nhất quán ngay lập tức sau mỗi giao dịch, mà thường được cập nhật theo lịch trình hoặc theo yêu cầu.

***Isolation (Tính cô lập):*** Đặc tính này đảm bảo rằng các giao dịch không tác động lẫn nhau. Trong DW, do quá trình ETL/ELT thường chạy theo hàng loạt và không tương tác trực tiếp với người dùng cuối, khái niệm isolation không áp dụng một cách trực tiếp.

***Durability (Tính bền vững):*** Đặc tính này đảm bảo rằng dữ liệu đã được lưu trữ sẽ không bị mất đi sau khi giao dịch hoàn tất. DW thường lưu trữ dữ liệu trong một hệ thống cơ sở dữ liệu hoặc hệ thống tệp tin có tính bền vững cao. Dữ liệu trong DW được duy trì và sao lưu định kỳ để đảm bảo tính bền vững.

Tóm lại, trong DW, tính toàn vẹn của dữ liệu được đảm bảo thông qua quá trình ETL/ELT và các biện pháp bảo mật dữ liệu mà không dựa trên các đặc tính ACID truyền thống như trong hệ thống cơ sở dữ liệu giao dịch.

#### ETL

**ETL** là bước rất quan trọng trong quá trình xây dựng kho dữ liệu, là quy trình sao chép dữ liệu từ một hoặc nhiều nguồn vào hệ thống đích đại diện cho dữ liệu khác với nguồn.

**Quy trình triển khai.** Bao gồm 3 bước chính là trích xuất dữ liệu (Extraction); Chuyển đổi dữ liệu (Transformation); Nạp dữ liệu vào kho dữ liệu (Loading).

Extract/Trích xuất là quá trình đọc dữ liệu từ cơ sở dữ liệu. Trong giai đoạn này, dữ liệu được thu thập, thường là từ nhiều nguồn khác nhau. Trích cái gì, trích ở đâu, cho ai, trích để làm gì. (vd trích dl trong 3 năm gần nhất, 3 cột này liên quan đến bán hàng). Phải có yêu cầu phân tích

Transform/Biến đổi là quá trình chuyển đổi dữ liệu được trích xuất từ biểu mẫu trước đó thành biểu mẫu cần có để có thể được đặt vào cơ sở dữ liệu khác. Chuyển đổi xảy ra bằng cách sử dụng các quy tắc hoặc bảng tra cứu hoặc bằng cách kết hợp dữ liệu này với dữ liệu khác. Tính toán thêm các cột, clean, chuyển đổi….chuyển đổi cái gì, vì sao chuyển đổi, chuyển cho ai, để làm gì

Load/Tải là quá trình ghi chép dữ liệu vào cơ sở dữ liệu đích. lưu vào đâu, lưu như thế nào, khả năng lưu trữ, load như thế nào? Có làm index không…

|  |  |
| --- | --- |
| **ETL (Extract, Transform, Load)** | **ELT (Extract, Load, Transform)** |
| Làm sạch dữ liệu,  sau đó mới thực hiện quá trình transform rồi mới load vào | Làm sạch dữ liệu, làm giàu, load dữ liệu vào data lake xong mới transform (nhược điểm là có thể public dữ liệu không cần thiết ra) |
| Transform tất cả dữ liệu vào DW | Thực hiện transform khi cần |
| * Thời gian load vào DW lâu hơn (vì thực hiện transform trước) * Phân tích dữ liệu nhanh hơn (dữ liệu đã sạch rồi nên lấy ra và phân tích nhanh hơn) | * Thời gian load vào nhanh hơn * Phân tích lâu hơn (muốn phân tích phải transform) |
| Không thể dùng cho việc transform các dữ liệu không cấu trúc và lưu vào DW | Có thể load tất cả dữ liệu không có cấu trúc vào Data Lake, ta có thể dùng chúng để phân tích |

Bảng 2.1 So sánh ETL và ELT

### Data Lake

Data lake là một hệ thống lưu trữ dữ liệu cho phép các tổ chức lưu trữ các lượng lớn dữ liệu không cấu trúc, bán cấu trúc và cấu trúc trong định dạng nguyên thủy của chúng. Khác với data warehouse, được thiết kế cho dữ liệu có cấu trúc và có một lược đồ được định nghĩa trước, data lake có thể lưu trữ bất kỳ loại dữ liệu nào, bao gồm dữ liệu nguyên thô từ nhiều nguồn khác nhau như mạng xã hội, cảm biến và các tệp nhật ký.

Data lake được thiết kế để có khả năng mở rộng, linh hoạt và tiết kiệm chi phí. Nó có thể lưu trữ dữ liệu dưới nhiều định dạng khác nhau, chẳng hạn như văn bản, hình ảnh, video và âm thanh. Data lake thường sử dụng Hadoop distributed file system (HDFS) hoặc lưu trữ đối tượng dựa trên đám mây, chẳng hạn như Amazon S3, Microsoft Azure Blob Storage hoặc Google Cloud Storage.

Data lake sử dụng phương pháp schema-bon-read, có nghĩa là dữ liệu không được cấu trúc cho đến khi nó được đọc bởi người dùng hoặc ứng dụng. Điều này cho phép người dùng phân tích dữ liệu dưới dạng nguyên thủy và áp dụng các kỹ thuật xử lý dữ liệu khác nhau như làm sạch, chuyển đổi và phân tích dữ liệu.

### LakeHouse

DataLake house là 1 kiến trúc quản lý dữ liệu kiểu mở, nó cho phép lưu trữ dữ liệu linh hoạt ở dạng không cấu trúc, bán cấu trúc và có cấu trúc từ nhiều nguồn khác nhau. Đồng thời, Data Lakehouse sử dụng framework là Delta Lake để quản lý dữ liệu và transaction ACID

Data lake house cho phép các nhà phát triển và nhà quản lý dữ liệu truy cập vào dữ liệu một cách nhanh chóng và hiệu quả hơn. Nó cũng cho phép các chuyên gia phân tích và khoa học dữ liệu tự do khám phá, khai thác và phân tích dữ liệu một cách linh hoạt hơn

## Apache Spark

Khái niệm: Apache Spark là một framework mã nguồn mở được phát triển bởi Apache Software Foundation để xử lý dữ liệu lớn (big data). Spark cung cấp một môi trường tính toán phân tán và được thiết kế để xử lý các tập dữ liệu lớn trên một cụm máy tính.

***Thành phần:***

* **Spark Core**: là thành phần cốt lõi của Spark và cung cấp các tính năng cơ bản của Spark như xử lý dữ liệu phân tán, quản lý bộ nhớ, lập lịch và phân phối tác vụ.
* **Spark SQL:** cung cấp các công cụ để truy vấn dữ liệu bằng SQL trên các tập dữ liệu có cấu trúc.
* **Spark Streaming**: cung cấp khả năng xử lý dữ liệu dòng (streaming data) trong thời gian thực.
* **Spark MLlib**: là thư viện học máy tích hợp sẵn trong Spark để xây dựng các mô hình học máy trên dữ liệu lớn.
* **GraphX:** cung cấp các công cụ để xử lý dữ liệu đồ thị lớn.

PySpark là một thư viện Spark được viết bằng Python để chạy các ứng dụng Python sử dụng khả năng của Apache Spark, sử dụng PySpark chúng ta có thể chạy song song các ứng dụng trên cụm phân tán (nhiều nút)

Pyspark cũng hỗ trợ các ngôn ngữ lập trình khác như Scala, Java và R. Pyspark được thiết kế để tương thích với các công nghệ lưu trữ dữ liệu phổ biến như Hadoop Distributed File System (HDFS), Apache Cassandra, Amazon S3...

Apache Spark là một công cụ xử lý phân tích cho các ứng dụng máy học và xử lý dữ liệu phân tán mạnh mẽ quy mô lớn

PySpark đã được sử dụng rất nhiều trong cộng đồng các nhà khoa học dữ liệu & máy học; nhờ vào các thư viện máy học Python rộng lớn. Spark chạy các hoạt động trên hàng tỷ nghìn tỷ dữ liệu trên các cụm phân tán nhanh hơn 100 lần so với các ứng dụng python truyền thống

## Medallion Architecture and data Change Feed

**A diagram of data quality

Description automatically generated**

Hình 2.2 Kiến trúc Medallion Architecture

### Bronze layer (raw data)

Lớp Đồng là nơi thu thập tất cả dữ liệu từ các hệ thống nguồn bên ngoài. Cấu trúc bảng trong lớp này tương ứng với cấu trúc bảng hệ thống nguồn "nguyên trạng", cùng với bất kỳ cột siêu dữ liệu bổ sung nào nắm bắt ngày/giờ tải, ID quy trình, v.v. Trọng tâm trong lớp này là Thu thập dữ liệu thay đổi nhanh và khả năng cung cấp kho lưu trữ lịch sử của nguồn (kho lạnh), dòng dữ liệu, khả năng kiểm tra, xử lý lại nếu cần mà không cần đọc lại dữ liệu từ hệ thống nguồn.

### Silver layer (cleansed and conformed data)

Trong lớp Silver của lakehouse, dữ liệu từ lớp Đồng được so khớp, hợp nhất, tuân thủ và làm sạch ("vừa đủ") để lớp Silver có thể cung cấp "Chế độ xem doanh nghiệp" về tất cả các thực thể kinh doanh chính, khái niệm và giao dịch. (ví dụ: khách hàng chính, cửa hàng, giao dịch không trùng lặp và bảng tham chiếu chéo).

Lớp Silver đưa dữ liệu từ các nguồn khác nhau vào chế độ xem Doanh nghiệp và cho phép phân tích tự phục vụ cho báo cáo đặc biệt, phân tích nâng cao và ML. Nó đóng vai trò là nguồn để các Nhà phân tích, Kỹ sư dữ liệu và Nhà khoa học dữ liệu của Bộ phận tiếp tục tạo các dự án và phân tích để giải đáp các vấn đề kinh doanh thông qua các dự án dữ liệu của bộ phận và doanh nghiệp trong Lớp Gold.

### Gold layer (curated business-level tables)

Dữ liệu trong lớp Gold của lakehouse thường được tổ chức trong cơ sở dữ liệu "dành riêng cho dự án" sẵn sàng sử dụng. Lớp Gold dùng để báo cáo và sử dụng các mô hình dữ liệu được tối ưu hóa đọc và không chuẩn hóa hơn với ít kết nối hơn. Lớp cuối cùng của quy tắc chuyển đổi dữ liệu và chất lượng dữ liệu được áp dụng tại đây. Lớp trình bày cuối cùng của các dự án như Phân tích khách hàng, Phân tích chất lượng sản phẩm, Phân tích hàng tồn kho, Phân khúc khách hàng, Đề xuất sản phẩm, Phân tích đánh dấu/bán hàng, v.v. phù hợp với lớp này. Chúng tôi thấy rất nhiều mô hình dữ liệu dựa trên lược đồ ngôi sao theo phong cách Kimball hoặc Data mart theo phong cách Inmon phù hợp với Lớp Gold này của nhà hồ.

## Giới thiệu về Microsoft Azure

### Microsoft Azure là gì?

Azure là một nền tảng điện toán đám mây (cloud computing platform) và một cổng trực tuyến (online portal) cho phép bạn truy cập và quản lý các dịch vụ (service) và tài nguyên (resource) đám mây do Microsoft cung cấp. Các dịch vụ và tài nguyên này bao gồm lưu trữ và chuyển đổi dữ liệu của bạn, tùy thuộc vào yêu cầu của bạn là gì. Để có quyền truy cập vào các tài nguyên và dịch vụ này, tất cả những gì bạn cần là kết nối internet và khả năng kết nối với Azure portal..

### Các dịch vụ Azure cung cấp

Azure cung cấp hơn 200 dịch vụ (services), được chia thành 18 loại (categories) bao gồm computing, networking, storage, IoT, migration, mobile, analytics, containers, artificial intelligence, machine learning, integration, management tools, developer tools, security, databases, DevOps, media identity và web services. Một số dịch vụ Azure chính theo danh mục:

Computing

**Virtual Machine**: Dịch vụ này cho phép bạn tạo một máy ảo trong Windows, Linux hoặc bất kỳ cấu hình nào khác trong vài giây.

**Cloud Service**: Dịch vụ này cho phép bạn tạo các ứng dụng có thể mở rộng trong đám mây. Sau khi ứng dụng được triển khai, mọi thứ, bao gồm load balancing, health monitoring đều do Azure đảm nhận.

**Service Fabric**: Với dịch vụ này, quá trình phát triển một microservice được đơn giản hóa rất nhiều. Microservice là một ứng dụng có chứa các ứng dụng nhỏ hơn khác được đóng gói.

**Functions**: Với Functions, bạn có thể tạo ứng dụng bằng bất kỳ ngôn ngữ lập trình nào. Phần tốt nhất về dịch vụ này là bạn không cần phải lo lắng về các yêu cầu phần cứng trong khi phát triển ứng dụng vì Azure sẽ lo điều đó. Tất cả những gì bạn cần làm là viết code.

Networking

**Azure CDN**: Azure CDN (Mạng phân phối nội dung/Content Delivery Network của Azure) giúp cung cấp nội dung cho người dùng, sử dụng băng thông cao và nội dung có thể được chuyển đến bất kỳ người nào trên toàn cầu. Dịch vụ CDN sử dụng một mạng lưới các máy chủ được đặt chiến lược trên toàn cầu để người dùng có thể truy cập dữ liệu càng sớm càng tốt.

**Express Route**: Dịch vụ này cho phép bạn kết nối mạng on-premises của mình với đám mây Microsoft hoặc bất kỳ dịch vụ nào khác mà bạn muốn, thông qua kết nối riêng tư (private). Vì vậy, thông tin liên lạc duy nhất sẽ xảy ra ở đây sẽ là giữa mạng doanh nghiệp và dịch vụ mà bạn muốn kết nối.

**Virtual network**: Mạng ảo cho phép bạn có bất kỳ dịch vụ Azure nào giao tiếp với nhau một cách riêng tư và an toàn.

**Azure DNS**: Dịch vụ này cho phép bạn lưu trữ các DNS domain hoặc system domain trên Azure.

Storage

**Disk Storage**: Dịch vụ này cho phép bạn chọn từ HDD hoặc SSD làm tùy chọn lưu trữ cùng với máy ảo của bạn.

**Blob Storage**: Dịch vụ này được tối ưu hóa để lưu trữ một lượng lớn dữ liệu phi cấu trúc (unstructured data), bao gồm văn bản (text) và thậm chí cả dữ liệu nhị phân (binary data).

**File Storage**: Đây là dịch vụ lưu trữ tệp được quản lý có thể được truy cập thông qua SMB (server message block) protocol.

**Queue Storage**: Với dịch vụ này, bạn có thể cung cấp xếp hàng chờ cho tin nhắn (message queuing) ổn định cho một khối lượng công việc (workload) lớn. Dịch vụ này có thể được truy cập từ mọi nơi trên thế giới

## Các dịch vụ demo

### Azure blob

Storage Azure Blob là giải pháp lưu trữ đối tượng của Microsoft cho đám mây. Bộ lưu trữ Blob được tối ưu hóa để lưu trữ một lượng lớn dữ liệu phi cấu trúc. Dữ liệu phi cấu trúc là dữ liệu không tuân theo mô hình hoặc định nghĩa dữ liệu cụ thể, chẳng hạn như dữ liệu văn bản hoặc dữ liệu nhị phân.

Blob storage resources gồm

* Storage account
* Container trong Storage account
* Blob trong container

A diagram of a computer

Description automatically generated with medium confidence

Hình 2.3 Blob storage

#### Storage account

Storage Azure Là một tài khoản, tại đây sẽ chứa tất cả các đối tượng dữ liệu như: blob, files, queues, table và disks. Dữ liệu trong này có thể truy cập được từ mọi nơi trên thế giới qua HTTP hoặc HTTPS. Dữ liệu trong storage account có tính bền, khả dụng cao, can toàn và có thể mở rộng lớn

***Các loại storage account:***

General-purpose v2 accounts: Một loại tài khoản lưu trữ cơ bản cho: blobs, files, queues, và tables. Và được đề xuất cho hầu hết các trường hợp sử dụng Azure Storage

General-purpose v1 accounts: dùng để sử dụng mở rộng cho purpose v2 accounts khi cần BlockBlobStorage accounts: Tài khoản lưu trữ cao cấp có hiệu suất cao cho block blobs và append blobs.

FileStorage accounts: tài khoản lưu trữ chỉ dành cho file có hiệu suất cao

BlobStorage accounts

#### Container trong Storage account

Là một vùng chứa để tập hợp các blob, tương tự như một thư mục trong hệ thống tệp. Một tài khoản lưu trữ có thể không giới hạn số lượng container và một container có thể lưu trữ không giới hạn số lượng các blobs.

#### Blob: Azure hỗ trợ 3 loại blob

***Block blobs:*** lưu trữ dữ liệu dạng text và dữ liệu nhị phân.

***Append blobs****:* lý tưởng cho việc ghi dữ liệu từ máy ảo

***Page blobs***: lưu trữ các tệp truy cập ngẫu nhiên có kích thước lên đến 8 TB

### Key vault

Microsoft Azure Key Vault là dịch vụ bảo mật dựa trên đám mây được Microsoft cung cấp như một phần của nền tảng Azure. Nó cung cấp một giải pháp lưu trữ tập trung và an toàn cho các khóa và bí mật mật mã , chẳng hạn như mật khẩu , chứng chỉ và khóa được sử dụng để mã hóa.

Cả ứng dụng và Microsoft đều không có quyền truy cập trực tiếp vào khóa. Thay vào đó, người dùng cấp quyền cho các ứng dụng của riêng họ và của bên thứ ba để sử dụng các khóa khi cần.

***Lợi ích của việc sử dụng Azure Key Vault để quản lý các “Key”***

* + ***Lưu trữ tập trung các “key” 1 cách an toàn:*** Azure Key Vault cung cấp một vị trí tập trung và an toàn để lưu trữ thông tin nhạy cảm, chẳng hạn như mật khẩu, chứng chỉ và khóa mã hóa. Lưu trữ an toàn này làm giảm nguy cơ vi phạm dữ liệu và duy trì tính bảo mật của hệ thống và dữ liệu.
  + ***Tăng tính năng bảo vệ cho các dịch vụ Azure khác***. Azure Key Vault có thể được tích hợp với các dịch vụ Azure khác, cho phép các tổ chức giảm hơn nữa nguy cơ vi phạm dữ liệu và cải thiện tính bảo mật của các tài nguyên dựa trên đám mây của họ.

### Azure DataBricks

Dịch vụ dữ liệu và AI từ Databricks có sẵn thông qua Microsoft Azure để lưu trữ tất cả dữ liệu của bạn trên một hồ chứa mở đơn giản và hợp nhất tất cả khối lượng công việc phân tích và AI của bạn

Azure Databricks được tối ưu hóa cho Azure và được tích hợp chặt chẽ với Azure Data Lake Storage, Azure Data Factory, Azure Synapse Analytics, Power BI và các dịch vụ Azure khác để lưu trữ tất cả dữ liệu của bạn trên một hồ chứa mở, đơn giản và hợp nhất tất cả khối lượng công việc phân tích và AI của bạn.

***Ưu điểm nổi bật:***

* + Hiệu suất gấp 50 lần cho khối lượng công việc Apache Spark
  + Dễ sử dụng
  + Hàng triệu giờ máy chủ mỗi ngày

### Data Factory

ADF không lưu trữ bất kỳ dữ liệu nào. Nó cho phép bạn tạo các quy trình công việc dựa trên dữ liệu để điều phối việc di chuyển dữ liệu giữa các kho dữ liệu được hỗ trợ, sau đó xử lý dữ liệu bằng các dịch vụ điện toán ở các khu vực khác hoặc trong môi trường tại chỗ. Nó cũng cho phép bạn theo dõi và quản lý quy trình công việc bằng cả cơ chế lập trình và giao diện người dùng.

Dịch vụ Data Factory cho phép tạo các đường ống dữ liệu để di chuyển và chuyển đổi dữ liệu, sau đó chạy các đường ống theo lịch trình cụ thể (hàng giờ, hàng ngày, hàng tuần, v.v.). Điều này có nghĩa là dữ liệu được sử dụng và tạo ra bởi quy trình làm việc là dữ liệu được chia theo thời gian và có thể chỉ định chế độ quy trình theo lịch trình (một lần một ngày) hoặc một lần.

***Trường hợp sử dụng:***

* + Hỗ trợ di chuyển dữ liệu
  + Nhận dữ liệu từ máy chủ của khách hàng hoặc dữ liệu trực tuyến tới Hồ dữ liệu Azure
  + Thực hiện các quy trình tích hợp dữ liệu khác nhau
  + Tích hợp dữ liệu từ các hệ thống ERP khác nhau và tải dữ liệu vào Azure Synapse để báo cáo

# Tổng quan về hệ thống xử lý dữ liệu trên Microsoft azure

## Tổng quan về xây dựng hệ thống:

Hệ thống xử lý và lưu trữ dữ liệu tự động được xây dựng trên nền tảng Cloud MicroSoft Azure. Hệ thống sẽ giải quyết vấn đề khi dữ liệu thường xuyên cần được tải lên để thêm mới hoặc cập nhật của người một cách tự động hóa thông qua các Pipeline được thiết kế sẵn.

Khi có Dữ liệu mới lên các trigger tự động sẽ kích hoạt các Pipeline được thiết kế sẵn để xử lý dữ liệu qua từng lớp. Ở Bronze\_Processing hệ thống sẽ truy cập và nhận dữ liệu tải lên từ người dùng sau đó sẽ lưu vào lớp Bronze, tiếp theo tại Silver\_Processing tại đây sẽ truy cập và lấy dữ liệu từ trong lớp Bronze sau đó sẽ xử lý làm sạch và lưu vào lớp Silver. Nguồn dữ liệu mới khi tải lên sẽ được merge với các dữ liệu cũ để cập nhật (thêm, xóa, cập nhật) lại dữ liệu cũ hiện có trong các lớp lưu trữ. Có thể truy xuất và phục hồi các phiên bản trong lịch sử nếu cần nhờ các tính năng “ChangeDataFeed” được sử dụng tron quá trình thiết kế hệ thống.

Quá trình trình truy cập vào các kho lưu trữ đảm bảo được tính bảo mật của dữ liệu thông qua việc sử dụng các keyvault các mật khẩu sẽ được đã mã hóa để sử dụng và không dùng trực tiếp các mật khẩu hiển thị trên quá trình truy cập các Storage

A diagram of a blockchain

Description automatically generated

Hình 3.1 Tổng quan về Pipeline

## Quy trình xử lý của hệ thống:

### Kích hoạt Trigger tự động:

Để kích hoạt trigger thì người dùng chỉ tải dữ liệu mới được thu thập trong quá trình hoạt động thường ngày của các bộ phận trong doanh nghiệp lên storage (Data Lake) của hệ thống, ngay lập tức trigger tự động sẽ kích hoạt quá trình xử lý, truy cập và lấy dữ liệu mới tải lên đưa dữ liệu qua các Pipeline tương ứng đã được xây dựng sẵn để xử lý và lưu trữ tại StorageLevel-02( Lake House).

### Xử lý dữ liệu qua các databricks trong pipeline:

**Tại Bronze\_Processing: S**ẽ sử dụng keyvault chứa key đã được mã hóa của StorageLevel-01 (Data Lake) để có thể truy cập và tải dữ liệu vừa được cập nhật lên hệ thống để xử lý. Dữ liệu sẽ được lưu qua StorageLevel-02 (Lake House) tại lớp Bronze dưới dạng delta và được kích hoạt tính năng “DataChangeFeed”, để có thể truy xuất lịch sử phiên bản của dữ liệu

**Tại Bronze\_Processing:** cũng sẽ sử dụng keyvault chứa key của StorageLevel-02 để truy cập vào lớp Bronze và lấy các phiên bản dữ liệu. Tại đây sẽ sử dụng tính năng history của Data Change Feed để lấy phiên bản mới nhất trong lớp Bronze để đưa vào hàm xử lý (Silver\_Processing). Sau khi xử lý dữ liệu sẽ được merger và lưu trữ vào lớp Silver, dữ liệu trong đây đã được làm sạch và cập nhật với nguồn dữ liệu cũ có sẵn cho quá trình phân tích.

### Kết thúc quy trình xử lý và lặp lại nếu người dùng tiếp tục tải dữ liệu lên

Dữ liệu sau khi đi qua các luồng xử lý sẽ được sử dụng cho quá trình phân tích và xây dựng các mô hình ML và Dashboard

***Quy trình sau khi nếu tiếp tục upload dữ liệu mới lên:***

Khi có dữ liệu mới được thêm vào ở storage\_level-1: trigger sẽ tự kích hoạt hệ thống, lấy dữ liệu mới xử lý sao đó lấy xử liệu cũ đã được xử lý merger lại với nhau thành tập dữ liệu mới hoàn chỉnh và tiếp tục lưu vào các storage tương ứng.

Để đảm bảo hiệu suất truy vấn tệp dữ liệu mới khi vừa tải lên thì sẽ sử dụng các tham số với giá trị file kích hoạt trigger để đưa vào trong quá trình xử lý của databricks, trong thực tế nếu có 1000 file thì sẽ thấy rõ được hiệu suất.

### Công nghệ được sử dụng

Trên nền tảng Azure Microsoft: Strange Account, DataFactory, DataBricks, Key Vault, power BI

Công nghệ : Delta Lake, Data lake, Data Change Feed

Ngôn ngữ lập trình: Scala Spark, SQL

## Tính năng của lakehouse và Delta Lake trong quá trình xử lý dữ liệu

**1. ACID Transaction:** Hỗ trợ ACID transaction, đảm bảo tính toàn vẹn và nhấtquán của dữ liệu

*Acomcity*: Các giao dịch hoạt động giống 1 nguyên tử, 1 giao dịch có thể hoạt động hoàn toàn hoặc không hoàn toàn hoàn thành. Ví dụ trong quá trình xử lý có 1 bước bị lỗi thì giao dịch sẽ không được hoàn thành và quay lại trạng thái ban đầu.

*Nhất quán*: Đảm bảo tính nhất quán của dữ liệu, nếu trước khi giao dịch dữ liệu ở trạng thái hợp lệ thì sau giao dịch dữ liệu cũng ở trạng thái hợp lệ

*Tính độc lập*: khi một giao dịch đang thực hiện thì các giao dịch khác ko được chăn ngang và ngược lại. Cho phép các giao dịch đảm bảo việc đối chiếu giữa các giao dịch không xảy ra và cung cấp thứ tự giữa các giao dịch.

*Tính bền vững*: Đảm bảo rằng các giao dịch trên dữ liệu của bạn được thực hiện thành công sẽ được lưu lại, ngay cả trong trường hợp lỗi hệ thống

2. Schema enforcement: xác định schema trong tập dữ liệu của mình, khi chúng ta có hoạt động thao tác thêm dữ liệu vào thì nó sẽ xem dữ liệu mới có phù hợp với cấu trúc schema đó không, nếu không thì từ chối và thông báo lỗi

3. Unified Bath & Streaming: cho phép người dùng làm việc với dữ liệu theo cả kiểu batch ( xử lý dữ liệu lớn và đồng bộ) và kiểu streaming (xử lý dữ liệu trực tuyến và không đồng bộ) mà không cần thay đổi cấu trúc lưu trữ hay code

Time Travel/ Data Snapshots: Delta Lake giữ lịch sử thay đổi của dữ liệu thông qua tính năng “Time travel”. Bạn có thể truy cập và phục hồi lại bất kỳ phiên bản nào của dữ liệu từ quá khứ, cho phép xem lại và phục hồi dữ liệu từ các phiên bản trước đó

## Mô tả về bộ dữ liệu dùng để Demo cho hệ thống

Nguồn kaggle: <https://www.kaggle.com/datasets/miclesve/datasetlaptop>

Dataset về dữ liệu mua laptop của sinh viên năm 2020

Dữ liệu gồm 2 sheet: 4

Sheet 1( customer profile): 1101 dòng, 18 cột

Sheet 2 (Sale): 1567 hàng, 10 cột

***Thông tin các cột của dữ liệu***

|  |  |
| --- | --- |
| **Customer profile** | **Giải thích** |
| NamHocThu | Năm học hiện tại của sinh viên |
| ChuyenNganh | Chuyên ngành học của sinh viên |
| NgheNghiepGiaDinh | Nghề nghiệp của gia đình sinh viên |
| ChiTieuHangThang | Số tiền chi tiêu hàng tháng của sinh viên |
| ThoiGianSuDung | Thời gian sử dụng laptop |
| MucDichSuDungLaptop | Các mục đích sử dụng laptop |
| MucDoHaiLong | Mức độ hài lòng của sinh viên đối với laptop đang sử dụng: 1 - Rất không hài lòng, 5 - Rất hài lòng |
| YeuToThuongHieuSP, YeuToCauHinh, YeuToTocDoXuLy, YeuToGia, YeuToThietKe, YeuToUyTinNhaBanle, YeuToBaoMat, YeuToBaoHanh, YeuToKhuyenMai | Các yếu tố cân nhắc khi chọn mua laptop theo mức độ: |
| 1 – Rất không quan trọng |
| 2 – Không quan trọng |
| 3 – Bình thường |
| 4 – Quan trọng |
| 5 – Rất quan trọng |
|  |  |
| **Sale** |  |
| Channel | Kênh mua sắm |
| PaymentType | Hình thức thanh toán |
| Brand | Thương hiệu laptop |
| Shop | Cửa hàng |
| Promotion | Khuyến mãi |
| Reference | Kênh thông tin biết đến sản phẩm và cửa hàng |

Bảng 3.1 Thông tin về bộ dữ liệu

# Xây dựng hệ thống

## Quy trình xây dựng

Quy trình xây dựng gồm 5 bước chính là tạo các Storage account để lưu dữ liệu, tạo key vault để quản lý các key, xử lý dữ liệu trên Databrick Azure, pipeline bằng Data Factory, cuối cùng tạo Sql server Azure để tải dữ liệu vào Database và liên kết với PowerBi trên DeskTop để trực quan hóa dữ liệu. Toàn bộ quy trình được thực hiện trên dịch vụ cloud của Microsoft Azure.

A screenshot of a computer

Description automatically generated

Hình 4.1 Tổng quát về resource group trên Azure

### Tạo stogare account

Tạo các Storage Account theo từng cấp độ tương ứng, trong mỗi storage account sẽ tạo những Containers để chứa dữ liệu với từ tệp dữ liệu. Tại Storage-Level01 sẽ tạo 2 ccontainers data-raw và data\_log cho mỗi file dữ liệu tương ứng (customers, sale).

Data raw: sẽ chứa các dữ liệu thô ban đầu. Dữ liệu tại đây là được dùng cho lần đầu đọc và xử lý dữ liệu đầu tiên của hệ thống

DataLog: là dữ liệu trong quá trình thu thập ghi lại nhật ký thay đổi của dữ liệu. Chẳng hạn như kích thước dữ liệu, sửa đổi, thêm hoặc có thể là cập nhật. Dữ liệu tại đây sẽ được đọc và xử lý cho các lần xử lý tiếp theo

A screenshot of a computer

Description automatically generated

Hình 4.2 Storage-Level01

Trong StogareLevel-2 tạo các containers tương ứng để lưu các lớp dữ liệu sau xử lý gồm lớp Bronze và lớp Silver

A screenshot of a computer

Description automatically generated

Hình 4.3 Stogare-Level-2

### Tạo Key Vault

Tạo key vault để lưu và mã hóa các key của từng Stograge. Đảm bảo tính bảo mật khi dùng để truy cập và load dữ liệu ở DataBricks (không sử dụng key trực tiếp trên code). Trong mỗi KeyVault sẽ chứa các key của các Storage Account và được mã hóa bên trong.

A screenshot of a computer

Description automatically generated

Hình 4.4 Tạo Key vault

### Xử lý bằng Databricks

#### Lớp Bronze- Processing

**Bước 1:** Tạo scope

Tạo Scope có đường dẫn của KeyVault đã tạo để tạo vùng truy cập đến chứa dữ liệu

A screenshot of a computer

Description automatically generated

Hình 4.5 Tạo scope trên databricks

**Bước 2**: Truy cập vào các StorageLevel\_01

Sử dụng scope đã tạo cùng keyvault để cấp quyền truy cập data trong Storage

A screenshot of a computer

Description automatically generated

Hình 4.6 Sử dụng Scope truy cập vào các Storage

**Bước 3:** Tạo tham số để nhập tên file dữ liệu từ tên file mới nhất được tải lên để kích hoạt trigger. File kích mới nhất kích hoạt trigger sẽ được sử dụng để xử lý chứ không phải các file cũ đã tải lên trước đó trong storage\_level01

A close-up of a website

Description automatically generated

Hình 4.7 Tạo tham số đầu vào cho file dữ liệu

Trong lần đầu đọc dữ liệu của hệ thống thì data sẽ được lấy từ container “Raw Data”

A computer code on a white background

Description automatically generated

Hình 4.8 Đọc dữ liệu lần đầu

Trong các lần đọc và xử lý dữ liệu tiếp theo sẽ đọc dữ liệu log từ container “Data-log”

A close-up of a computer code

Description automatically generated

Hình 4.9 Đọc dữ liệu sau lần đầu tiên

**Bước 4: Quy trình xử lý khi lưu dữ liệu vào Bronze**

Trong lần đầu xử lý thì Delta Table với đường dẫn để lưu lớp Bronze của dữ liệu sẽ trống và không ở dạng delta. Vì vậy nếu DelatTable với đường dẫn Bronze trả về Flash thì dữ liệu mới đầu tiên sẽ được tải vào lớp Bronze và sẽ kích hoạt Change Data Feed để có thể sử dụng các tính năng truy cập các phiên bản history .

Trong các lần xử lý tiếp theo, Delta Table của Bronze đã được kích hoạt và trả về True thì dữ liệu sẽ được lấy từ data-log và tiến hành merge dữ liệu mới vào bảng dữ liệu của bronze theo các điều kiện được tạo sẵn trong data-log. Nếu trong ‘Operation’ của nguồn dữ liệu mới là delete thì sẽ tiến hành xóa dòng đó. Khi update sẽ cập nhật lại. Còn các trường hợp còn lại như insert thì sẽ thêm các hàng mới vào trong dữ liệu cũ.

***Giải thích về cột Operation:*** Trong các file mới được update sẽ có thêm cột (Operation có thể là delete hoặc insert) để làm điều kiện merge gọi là file log, đây là file được thu thập trong quá trình hoạt động mà khách hàng thu được mà muốn cập nhật lên lại hệ thống.

**A screenshot of a computer code

Description automatically generated**

Hình 4.10 Lưu dữ liệu vào lớp Bronze

#### Silver- Processing

Bước 1: Tương tự ở Bronze\_Processing, cũng sẽ tạo Scope để có thể truy cập vào lớp dữ liệu được lưu ở lớp Bronze.

A screenshot of a computer

Description automatically generated

Hình 4.11 Tạo Scope

**Bước 2:** Truy cập và tải dữ liệu theo đường dẫn từ vùng truy cập scope và keyvault vừa tạo. Khi sử dụng keyvault sẽ đảm bảo được tính bảo mật vì không sử dụng key trực tiếp trên code, key đã được mã hóa trong keyvault



Hình 4.12 Sử dụng scope và keyvault truy cập vào Storage

**Bước 3:** Tạo hàm làm sạch dữ liệu

Tạo hàm gồm tất cả các quy trình làm sạch xử liệu thô bao gồm như xử lý dữ liệu null, giá trị trùng lặp, schema hay các lỗi có trong tập dữ liệu. Hàm làm sạch dữ liệu sẽ sử dụng để xử lý dữ liệu trước khi đưa dữ liệu vào trong lớp Silver

**Bước 4**: Xây dựng quy trình tải dữ liệu thô từ Bronze sau đó đưa qua hàm làm sạch và lưu vào lớp Silver

Ở lần đầu tải dữ liệu lên lớp đường dẫn lưu trữ của lớp silver vẫn chưa phải dạng Delta table, thì hệ thống sẽ đọc file dữ liệu thô từ lớp Bronze sau đó đưa qua hàm xử lý để làm sạch. Tiếp theo đó là lưu dữ liệu đã làm sạch dưới dạng Delta vào đường dẫn đã tạo và kích hoạt tính năng Change Data Feed.

A computer code with many colorful text

Description automatically generated with medium confidence

Hình 4.13 Lưu dữ liệu vào silver ở lần đầu

Trong những lần tiếp theo sẽ lấy dữ liệu mới nhất vừa được tải lên lớp bronze bằng tính năng history của change data feed mà t đã kích hoạt ở các lớp từ lần đầu xử lý. Sau đó dữ liệu sẽ đưa qua hàm xử lý để làm sạch và cuối cùng sẽ merge với bảng dữ liệu trong Silver.

A screen shot of a computer code

Description automatically generated

Hình 4.14 Xây dựng luồng dữ liệu trên Data Factory

### Tạo SQL Server và database trên azure

### Sử Data Factory xây dựng các đường ống xử lý dữ liệu theo quy trình

Xây dựng luồng dữ liệu trên data Factory: Dữ liệu sau khi kích hoạt trigger sẽ đưa qua 2 quy trình xử lý của databricks là Bronze\_Processing và Silver\_Processing. Sau đó dữ liệu từ Silver\_Processing sẽ được sao chép vào DataBase đã được liên kết. Từ DataBase chúng ta có thể đưa dữ liệu vào PowerBi để trực quan hóa dữ liệu.

A screenshot of a computer

Description automatically generated

Hình 4.15 Xây dựng luồng dữ liệu trên Data Factory

A screenshot of a computer

Description automatically generated

* Tạo các đường liên kết dịch vụ với Storage Account, DataBricks, Azure SQL trên Pipeline
* Tạo các tham số cần thiết để liên kết các file dữ liệu khi được tải lên để kích hoạt Trigger và sử dụng file đó để xử lý trên DataBricks
* Tạo Trigger tự động khi người dùng tải tệp dữ liệu mới lên sẽ kích hoạt hệ thống. File bắt đầu bằng Cust hoặc Sale tùy vào loại dữ liệu và kết thúc bằng .CVS. Sẽ có 2 trigger với cho raw-data và data-log
* Thêm Trigger đã tạo vào pipeline và Publish all cho hệ thống và hoàn thành

## Các bước trong hàm xử lý dữ liệu của DataBricks

### Bảng customer

**Bước 1:** Kiểm tra các giá trị null

* + Đếm các hàng có giá trị null của từng cột
  + Hiển thị các hàng có giá trị null
  + Xóa các hàng có giá trị null

**Bước 2:** Loại bỏ các ký tự lỗi ở các cột QueQuan, NamHocThu, ThoiGianSuDung

**Bước 3:** Lọc các hàng ở các cột đánh giá, có giá trị bé hơn 0 và lớn hơn 5

* Chỉnh các giá trị ngoại lai ở Cột MucDoHaiLong
  + Tính giá trị trung bình của các cột.....
  + Lấy phần nguyên của giá trị trung bình fill vào các hàng có giá trị không hợp lệ

**Bước 4:** Chuyển các giá trị ở cột GioiTinh về Nam và Nữ

* + Xem các giá trị duy nhất trong cột GioiTinh
  + Thay thế tất cả về Nam và Nữ

**Bước 5:** Chuyển các cột có giá trị số nguyên về kiểu dữ liệu integer

### Bảng Sale

**Bước 1**: Chuyển giá trị có ký tự $ thành tiền Việt

* + Lọc ra các hàng ở cột SalePrice có kí tự $
  + Xóa bỏ ký tự $ và nhân 23000 để chuyển thành tiền Việt

**Bước 2**: Thay thế các giá trị null

**Bước 3:** Xóa các hàng trùng lặp

**Bước 4:** Xử lý kiểu dữ liệu của cột sale Date

## Kiểm tra và chạy thử hoạt động của hệ thống sau khi hoàn thành gồm trigger, pipeline và databricks

### Lần chạy đầu tiên với rawdata

#### Hoạt động của trigger và pipeline

Sau khi đẩy dữ liệu thô vào container Raw-data tiến hành vào “Monitor” của “DataFactory” kiểm tra trigger cũng như hoạt động của pipeline có hoạt động hay không. Nếu trigger hoạt động thì kiểm tra pipeline chạy có lỗi hay không. Nếu không thì tiếp tục triển khai lần chạy tiếp theo với các tệp dữ liệu data-log

Trigger raw-customer chạy thành công

A screenshot of a computer

Description automatically generated

Hình 4.16 Hoạt động của trigger Raw

***Pipeline chạy thành công***

A screenshot of a computer

Description automatically generated

Hình 4.17 Pipeline chạy thành công

#### Kiểm tra kết quả trả về của data bricks

Xem chi tiết kết quả thông qua link Output trả về

A screenshot of a computer

Description automatically generated

Hình 4.18 Xem kết quả trả về ua DataBricks

Qua kết quả bên dưới có thể thấy được Parameters khớp với file dữ liệu raw-data đã tải lên để kích hoạt trigger chạy. Và số lượng dòng dữ liệu được ghi vào 2 lớp Bronze và Silver là 501 dòng

A screenshot of a computer

Description automatically generated

Hình 4.19 Kết quả Bronze\_Processing

A screenshot of a computer

Description automatically generated

Hình 4.20 Kết quả Silver\_Processing

### Lần chạy tiếp theo với các tệp data-log

Sau lần đầu tiên ta sẽ tải tệp các tệp dữ liệu data-log có thể gồm insert, delete, update. Trong Demo này sẽ tải một tệp data-log insert với số dòng là.

Trong lần này sẽ kiểm tra trigger-log có hoạt động hay không, và kết quả trả về của databrick có thay đổi đúng với số dòng insert của tệp data-log hay không

#### Hoạt động của trigger-log và pipeline

Trigger chạy thành công

A screenshot of a computer

Description automatically generated

Hình 4.21 Trigger-Log chạy thành công

A screenshot of a computer

Description automatically generatedPipeline hoạt động thành công

Hình 4.22 Pipeline do trigger-log kích hoạt chạy thành công

#### Kiểm tra kết quả trả về

Theo kết quả trả về của databricks Parameters là file insert-log và dữ liệu được tăng thêm lên 555 dòng so với ban đầu là 501 dòng ở cả 2 lớp Bronze và Silver

A screenshot of a computer

Description automatically generated

Hình 4.23 Xem kết quả ở Bronze\_Processing

A screenshot of a computer

Description automatically generated

Hình 4.24 Xem kết quả ở Silver\_Processing

### Cuối cùng kiểm tra dữ liệu sau khi được lưu vào cơ sở dữ liệu SQL server

Sau khi kiểm tra xong các chức năng của hệ thống sẽ kiểm tra dữ liệu cuối cùng được đưa vào trong Database để phân tích

A screenshot of a computer

Description automatically generated

Hình 4.25 Kiểm tra dữ liệu nạp vào Database

# TẠO DASHBOARD BẰNG POWER BI

## Tạo Dashboard bằng Power Bi

### Liên kết dữ liệu trên SQL Azure Database vào PowerBi Desktop

Lấy liên kết và thực hiện liên kết bằng DataBase trên server SQL Azure đã tạo. Sau đó chọn các bảng dữ liệu cần cần phân tích.

Đảm bảo tính bảo mật của sever tất cả các networking muốn kết nối truy cập vào server đều cần cung cấp địa chỉ IP.

A screenshot of a computer

Description automatically generated

Hình 5.1 Cung cấp địa chỉ IP của máy cho networking server

A screenshot of a computer

Description automatically generated

Hình 5.2 LIên kết dữ liệu DataBase với PowerBI

## Tạo Dashboard

Qua Dashboard trên sẽ có cái nhìn một cách trực quan hơn về bộ dữ liệu, tuy nhiên để phân tích sâu hơn về dữ liệu và đưa ra các kết luận chính xác thì cần phải xây dựng thêm các mô hình machine learning, các thuật toán phân cụm phân lớp.

### Dashboard thông tin khách hàng

Ở Dashboard sẽ cho chúng ta thấy được một cách tổng quan nhất về các thông tin của khách hàng như chi tiêu hàng tháng, chuyên ngành, nghề nghiệp gia đình, quê quán, năm học của sinh viên.

A screenshot of a graph

Description automatically generated

Hình 5.3 Dashboard thông tin khách hàng

### Dashboard về hành vi khách hàng

A screenshot of a graph

Description automatically generated

Hình 5.4 Dashboard về hành vi khách hàng

# TÀI LIỆU THAM KHẢO

1. <https://spark.apache.org/>

2. [Microsoft Azure](https://vinsep.com/kien-thuc/microsoft-kien-thuc/azure/tim-hieu-ve-microsoft-azure-cloud-computing/)

3. [Data warehouse](https://viblo.asia/p/kien-thuc-co-ban-ve-kien-truc-dwh-data-warehouse-architecture-zXRJ8n8NVGq)

# PHỤ LỤC

**1. Code Bronze\_Processing**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**2. Code Silver\_Processing**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated