

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**KHOA TOÁN - TIN**

**BÁO CÁO THỰC TẬP KỸ THUẬT KỲ 2024.2**

**ĐỖ TRUNG QUÂN**

Email: Quan.DT216873@sis.hust.edu.vn

Mã sinh viên: 20216873

**Chuyên ngành Toán - Tin**

**Đơn vị thực tập:** Công ty Cổ phần Giải pháp Phần mềm Tài chính FSS

**HÀ NỘI, 6/2025**

# Phiếu đánh giá kết quả thực tập

ĐẠI HỌC BÁCH KHOA HÀ NỘI  
KHOA TOÁN – TIN

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM  
Độc lập - Tự do - Hạnh phúc

## PHIẾU ĐÁNH GIÁ KẾT QUẢ THỰC TẬP

Sinh viên : ...Đỗ Trung Quân.....  
Mã số sinh viên : ...20216833..... Lớp : ...Toán Tin 02.....  
Số điện thoại : ...0384.349.556.. Email : ...trungquan1832003@gmail.com  
Địa điểm thực tập : ...Công ty Cổ phần Giải pháp phần mềm tài chính.....  
Thời gian thực tập : ...2 tháng..... Hình thức : Full time / Part time  
Cán bộ hướng dẫn tại cơ sở : .....Nguyễn Xuân Lộc..... (nếu có)

### 1. Nhận xét của cơ sở thực tập

#### a) Nhận xét kết quả thực hiện đợt thực tập

- Về Chuyên môn, nghiệp vụ:  
...Chuyên môn kỹ thuật khá, đáp ứng tốt yêu cầu công việc.....
- Về Kỹ năng (Testing/Nghiên cứu/Viết báo cáo/Thuyết trình, ....):  
...Khả năng tiếp thu, nghiên cứu vấn đề tốt.....
- Về Ứng xử doanh nghiệp:  
...Tuân thủ tốt nội quy nơi làm việc.....  
...Có gắng nâng cao khả năng giao tiếp để hòa nhập tập thể.....


#### b) Ý thức của sinh viên

Sinh viên đã hoàn thành đợt thực tập với ý thức : (Tốt) / Khá / Kém

#### c) Kết quả đạt được

Điểm: ...9.....

Xác nhận của cán bộ hướng dẫn (nếu có)

  
Nguyễn Xuân Lộc

Xác nhận của cơ sở thực tập



GIÁM ĐỐC NHÂN SỰ  
*Trịnh Thị Hồng Vân*

# Mục lục

<b>Chương 1 Tổng quan về đơn vị thực tập</b>	<b>1</b>
1.1 Lịch sử hình thành và phát triển . . . . .	1
1.2 Khách hàng và uy tín trên thị trường . . . . .	1
1.3 Giá trị cốt lõi và văn hóa doanh nghiệp . . . . .	2
1.4 Các mảng sản phẩm và dịch vụ chính . . . . .	2
<b>Chương 2 Tổng quan về kiến thức thực tập</b>	<b>4</b>
2.1 Tổng quan về AWS . . . . .	4
2.1.1 Amazon S3 (Simple Storage Service) . . . . .	4
2.1.2 Amazon Athena . . . . .	5
2.1.3 AWS Glue . . . . .	5
2.1.4 Amazon DynamoDB . . . . .	6
2.1.5 AWS CodePipeline . . . . .	6
2.1.6 AWS Step Functions . . . . .	6
2.1.7 Amazon CloudWatch . . . . .	7
2.2 Tổng quan về Databricks . . . . .	7
2.2.1 Workspace . . . . .	8
2.2.2 SQL . . . . .	10
2.2.3 Data Engineering . . . . .	12
<b>Chương 3 Nội dung thực tập</b>	<b>14</b>
3.1 Xây dựng quy trình xử lý dữ liệu trên nền tảng AWS Glue . . . . .	14
3.1.1 Thiết lập và cấu hình Entry Job trong DynamoDB . . . . .	14
3.1.2 Thiết lập và cấu hình Glue Job . . . . .	15

3.1.3	Thực hiện chạy Job thông qua AWS Lambda . . . . .	16
3.1.4	Đánh giá hiệu quả và vai trò . . . . .	17
3.2	Xây dựng quy trình xử lý dữ liệu trên nền tảng Databricks Workflow .	17
3.2.1	Job thủ công . . . . .	17
3.2.2	Job tự động bằng trigger file arrival (job sẽ tự động chạy mỗi khi volume được chỉ định trong storage location nhận file mới)	22
3.2.3	Job tự động bằng depend on . . . . .	25
<b>Kết luận</b>		<b>28</b>

# **Chương 1**

## **Tổng quan về đơn vị thực tập**

### **1.1 Lịch sử hình thành và phát triển**

Công ty Cổ phần Giải pháp Phần mềm Tài chính (FSS) được thành lập vào ngày 18 tháng 3 năm 2008 bởi một nhóm chuyên gia giàu kinh nghiệm trong lĩnh vực công nghệ thông tin, đặc biệt trong các lĩnh vực tài chính, ngân hàng và chứng khoán tại Việt Nam cũng như trong khu vực.

Trải qua hơn 16 năm xây dựng và phát triển, FSS đã từng bước khẳng định vị thế trên thị trường công nghệ tài chính, với đội ngũ hơn 400 nhân sự có trình độ chuyên môn cao. Công ty hiện là một trong những đơn vị hàng đầu cung cấp các giải pháp phần mềm tài chính – ngân hàng tại Việt Nam.

### **1.2 Khách hàng và uy tín trên thị trường**

FSS đã thiết lập được mối quan hệ hợp tác lâu dài với nhiều khách hàng lớn, tiêu biểu là các ngân hàng và công ty chứng khoán uy tín như BIDV, TechcomBank, VPBank, MBBank, ACB, Maritime Bank, VNDirect, BSC, BVSC, VCBS,...

Nhờ vào chất lượng sản phẩm và dịch vụ, FSS luôn giữ vững tỷ lệ thành công 100% trong việc triển khai các dự án, qua đó xây dựng được uy tín vững chắc trên thị trường công nghệ tài chính trong nước.

### 1.3 Giá trị cốt lõi và văn hóa doanh nghiệp

FSS hoạt động dựa trên những giá trị cốt lõi được duy trì và phát triển xuyên suốt quá trình vận hành doanh nghiệp, cụ thể như sau:

- **Cam kết dài hạn – Hỗ trợ 24/7:** FSS đặc biệt chú trọng đến chất lượng dịch vụ hậu mãi, luôn duy trì đội ngũ kỹ thuật sẵn sàng hỗ trợ khách hàng liên tục bất kể thời gian.
- **Chuyên nghiệp – Hiểu rõ nghiệp vụ:** Đội ngũ kỹ sư và chuyên gia tại FSS không chỉ giỏi về công nghệ mà còn am hiểu sâu sắc nghiệp vụ tài chính – ngân hàng – chứng khoán, tạo nên lợi thế trong việc triển khai các hệ thống phức tạp như core banking, hệ thống giao dịch, báo cáo thông minh (BI), v.v.
- **Tiêu chuẩn hóa cao – Linh hoạt tùy biến:** Các sản phẩm phần mềm của FSS đều được xây dựng dựa trên các tiêu chuẩn quốc tế, đồng thời vẫn có khả năng tùy biến linh hoạt để phù hợp với yêu cầu và điều kiện thực tế tại Việt Nam.
- **Quy trình chất lượng – Tùy chỉnh theo dự án:** FSS áp dụng hệ thống quản lý chất lượng theo tiêu chuẩn ISO 9001 và phát triển các quy trình nội bộ phù hợp với từng loại dự án nhằm đảm bảo tiến độ và hiệu quả tối ưu.

### 1.4 Các mảng sản phẩm và dịch vụ chính

FSS tập trung hoạt động trong ba lĩnh vực sản phẩm – dịch vụ chính, với đội ngũ chuyên trách và công nghệ hiện đại, đáp ứng các nhu cầu đa dạng từ khách hàng trong ngành tài chính:

#### 1. Phát triển hệ thống giao dịch chứng khoán (Core chứng khoán)

FSS cung cấp các hệ thống hỗ trợ toàn bộ quy trình nghiệp vụ giao dịch chứng khoán, bao gồm: khớp lệnh, thanh toán, lưu ký, quản lý tài khoản, phân tích dữ liệu giao dịch,... Các hệ thống này được thiết kế nhằm đảm bảo tính an toàn, tốc độ xử lý cao và khả năng mở rộng khi cần thiết.

## **2. Xây dựng hệ thống Kho dữ liệu và phân tích thông minh (Data Warehouse & BI)**

Dịch vụ này hỗ trợ khách hàng – đặc biệt là các ngân hàng – triển khai kho dữ liệu tập trung, kết hợp với công cụ phân tích thông minh nhằm phục vụ báo cáo nội bộ, phân tích hành vi khách hàng, cũng như hỗ trợ các quyết định chiến lược, quản lý rủi ro và tăng trưởng kinh doanh.

## **3. Phát triển phần mềm theo yêu cầu (Custom Software Development)**

FSS thực hiện các dự án phát triển phần mềm theo yêu cầu riêng của từng khách hàng, bao gồm các cơ quan nhà nước như Tổng cục Thuế, Trung tâm Lưu ký Chứng khoán (VSD), cũng như các tổ chức tài chính – bảo hiểm. Sản phẩm có thể là hệ thống quản lý thuế, hệ thống giao dịch vàng/hàng hóa, hoặc phần mềm tuân thủ các tiêu chuẩn báo cáo quốc tế như IFRS.

## Chương 2

# Tổng quan về kiến thức thực tập

### 2.1 Tổng quan về AWS

Amazon Web Services (AWS) là nền tảng điện toán đám mây toàn diện và được sử dụng rộng rãi nhất hiện nay, cung cấp hơn 200 dịch vụ từ các trung tâm dữ liệu toàn cầu. Với khả năng mở rộng linh hoạt, độ tin cậy cao, bảo mật và hiệu suất mạnh mẽ, AWS giúp các tổ chức triển khai hệ thống một cách nhanh chóng và tiết kiệm chi phí. AWS phục vụ đa dạng đối tượng khách hàng từ các doanh nghiệp lớn, công ty khởi nghiệp cho đến các tổ chức chính phủ và giáo dục.

AWS cho phép người dùng chỉ phải trả tiền cho tài nguyên mà họ sử dụng, đồng thời hỗ trợ tự động mở rộng và co giãn tài nguyên theo nhu cầu thực tế. Nhờ vậy, các hệ thống xây dựng trên nền tảng AWS có thể đáp ứng được các yêu cầu khắt khe về hiệu năng, độ tin cậy và tính sẵn sàng cao.

#### 2.1.1 Amazon S3 (Simple Storage Service)

Amazon S3 là dịch vụ lưu trữ đối tượng có khả năng mở rộng gần như vô hạn, cho phép người dùng lưu trữ và truy xuất dữ liệu từ bất kỳ đâu thông qua Internet. Dữ liệu trong S3 được tổ chức dưới dạng các bucket và đối tượng, hỗ trợ nhiều lớp lưu trữ như Standard, Intelligent-Tiering, Glacier, đáp ứng nhu cầu lưu trữ từ thường xuyên đến lưu trữ lâu dài.

Amazon S3 đóng vai trò quan trọng trong các hệ thống dữ liệu lớn khi được dùng



để lưu trữ dữ liệu thô, kết quả xử lý trung gian, bản ghi log, ảnh, video và các tệp cấu hình. Với khả năng tích hợp chặt chẽ với các dịch vụ như AWS Glue, Athena và EMR, S3 thường được xem là “data lake” – kho lưu trữ trung tâm cho toàn bộ dữ liệu phân tích.

### **2.1.2 Amazon Athena**

Amazon Athena là dịch vụ truy vấn dữ liệu tương tác không cần máy chủ, cho phép người dùng sử dụng cú pháp SQL để truy xuất trực tiếp dữ liệu lưu trữ trên S3. Athena được xây dựng dựa trên Presto (hoặc Trino), một engine truy vấn phân tán mạnh mẽ, giúp thực hiện các truy vấn phức tạp mà không cần triển khai hệ thống cơ sở dữ liệu.

Athena đặc biệt phù hợp cho các nhu cầu phân tích ad-hoc (truy vấn theo yêu cầu), kiểm tra dữ liệu trong quá trình ETL hoặc khám phá dữ liệu chưa được cấu trúc rõ ràng. Việc tính phí của Athena dựa trên dung lượng dữ liệu được truy vấn, khuyến khích người dùng tối ưu định dạng dữ liệu (như Parquet, ORC) để giảm chi phí và tăng hiệu suất.

### **2.1.3 AWS Glue**

AWS Glue là một dịch vụ ETL (Extract, Transform, Load) không máy chủ, được thiết kế để hỗ trợ xử lý dữ liệu lớn trên quy mô rộng. Glue giúp người dùng khám phá, chuẩn hóa, phân tích và chuyển đổi dữ liệu một cách dễ dàng thông qua giao diện trực quan hoặc lập trình với Python/Scala.

Một thành phần quan trọng của Glue là Glue Data Catalog – nơi quản lý metadata (schema) cho dữ liệu nằm trong S3 hoặc các nguồn dữ liệu khác. Các script ETL trong Glue có thể được khởi chạy theo lịch trình hoặc tích hợp trong các pipeline phức tạp. AWS Glue phù hợp với các bài toán xử lý dữ liệu dạng batch hoặc streaming, và là trung tâm của các giải pháp data lake hiện đại.

## 2.1.4 Amazon DynamoDB

Amazon DynamoDB là một dịch vụ cơ sở dữ liệu NoSQL được quản lý hoàn toàn, cung cấp hiệu suất cao với độ trễ thấp ở quy mô lớn. DynamoDB hỗ trợ mô hình dữ liệu key-value và document, giúp dễ dàng mở rộng theo chiều ngang để xử lý hàng triệu truy vấn mỗi giây.

Với khả năng tự động chia phân vùng, mã hóa dữ liệu, và tích hợp với DynamoDB Streams, dịch vụ này thường được sử dụng trong các ứng dụng yêu cầu thời gian phản hồi nhanh như: hệ thống quản lý phiên người dùng, giỏ hàng thương mại điện tử, bảng xếp hạng thời gian thực hoặc dịch vụ thông báo. DynamoDB cũng hỗ trợ TTL (Time-to-Live), backup, restore và chế độ global tables cho triển khai đa khu vực.

## 2.1.5 AWS CodePipeline

AWS CodePipeline là dịch vụ tích hợp liên tục và triển khai liên tục (CI/CD) giúp tự động hóa toàn bộ quy trình xây dựng, kiểm thử và triển khai phần mềm. CodePipeline kết nối nhiều công cụ và dịch vụ như GitHub, CodeBuild, CodeDeploy, CloudFormation,... để hình thành một chuỗi các giai đoạn (stages) trong pipeline.

Việc sử dụng CodePipeline đảm bảo mọi thay đổi mã nguồn đều được kiểm thử và triển khai nhất quán theo quy trình tự động, từ đó giảm thiểu lỗi thủ công và tăng tốc độ phát hành phần mềm. Đặc biệt, khi kết hợp với các kiểm thử tự động hoặc phân phối đa môi trường (Dev, UAT, Prod), CodePipeline trở thành một công cụ CI/CD mạnh mẽ trong hệ sinh thái AWS.

## 2.1.6 AWS Step Functions

AWS Step Functions là dịch vụ điều phối quy trình công việc (workflow orchestration) cho phép xây dựng các luồng xử lý phức tạp bằng cách kết nối nhiều dịch vụ AWS với nhau. Step Functions sử dụng mô hình trạng thái (state machine) để định nghĩa từng bước trong quy trình và điều kiện chuyển tiếp.

Ứng dụng phổ biến của Step Functions bao gồm: quy trình xử lý dữ liệu nhiều bước (multi-step ETL), chuỗi thao tác theo điều kiện logic, xử lý đơn hàng, kiểm thử đa giai

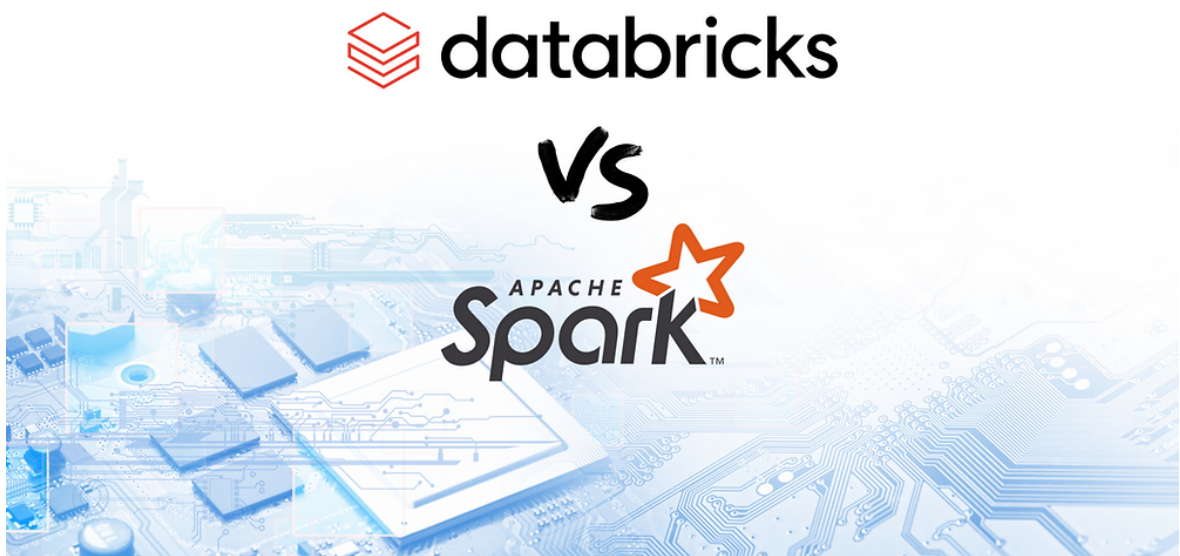
đoạn, hoặc kết hợp giữa thủ công và tự động. Các bước trong Step Functions có thể là lambda function, job EMR, Glue ETL, hoặc thậm chí là các chờ điều kiện từ hệ thống bên ngoài.

### 2.1.7 Amazon CloudWatch

Amazon CloudWatch là dịch vụ giám sát và quan sát hệ thống AWS, cung cấp khả năng thu thập log, theo dõi số liệu (metrics) và cảnh báo tự động. CloudWatch tích hợp sâu với các dịch vụ như EC2, Lambda, RDS, S3, Glue,... giúp người quản trị có cái nhìn toàn cảnh về trạng thái và hiệu suất hệ thống.

CloudWatch Logs có thể lưu trữ và tìm kiếm log ứng dụng, trong khi CloudWatch Metrics theo dõi các chỉ số như CPU, bộ nhớ, số lượng bản ghi, độ trễ,... Dịch vụ này cũng hỗ trợ thiết lập cảnh báo (alarms) và hành động phản ứng (như gửi email, gọi Lambda function hoặc tự động scale tài nguyên). Đây là công cụ không thể thiếu trong việc vận hành và đảm bảo tính ổn định của các hệ thống lớn trên AWS.

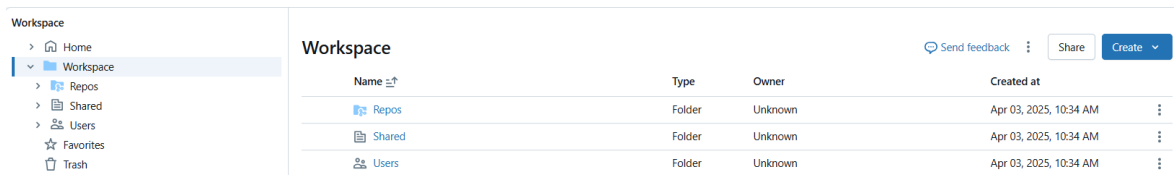
## 2.2 Tổng quan về Databricks



Databricks là một nền tảng được triển khai trên các dịch vụ cloud (AWS) để xử lý dữ liệu lớn - xây dựng ETL pipelines, phân tích dữ liệu, và triển khai machine learning.

## 2.2.1 Workspace

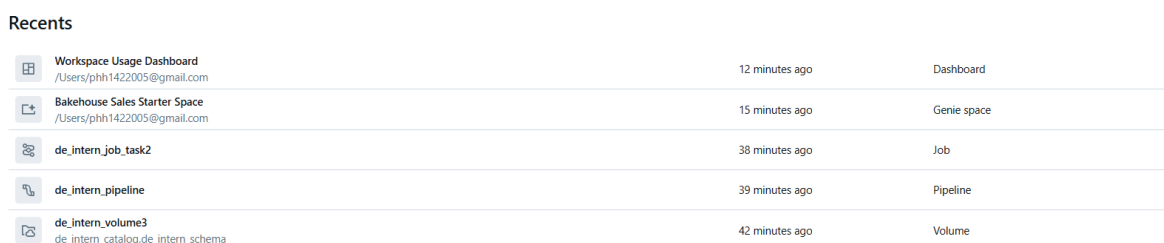
**Workspace:** Không gian làm việc để tổ chức và quản lý các notebook, tệp dữ liệu, thư mục và lược đồ dự án.



Name	Type	Owner	Created at
Repos	Folder	Unknown	Apr 03, 2025, 10:34 AM
Shared	Folder	Unknown	Apr 03, 2025, 10:34 AM
Users	Folder	Unknown	Apr 03, 2025, 10:34 AM

- Home: Là khu vực cá nhân của người dùng, nơi chứa các notebook, tệp dữ liệu, và thư mục mà người dùng sở hữu hoặc đã tạo.
- Workspace (repo, shared, users):
  - Repo: Lưu trữ các mã nguồn và dự án dưới dạng repository
  - Shared: Chứa các tài nguyên được chia sẻ với nhóm hoặc tổ chức, có thể truy cập và chỉnh sửa chung.
  - Users: Chứa các tài nguyên được tổ chức theo từng người dùng
- Favorite: Nơi lưu trữ các tài nguyên, notebook hoặc thư mục yêu thích để dễ dàng truy cập lại.
- Trash: Nơi chứa các tài nguyên đã bị xóa, cho phép khôi phục lại nếu cần thiết.

**Recents:** Hiển thị các notebook, tệp dữ liệu và thư mục đã làm việc gần đây.



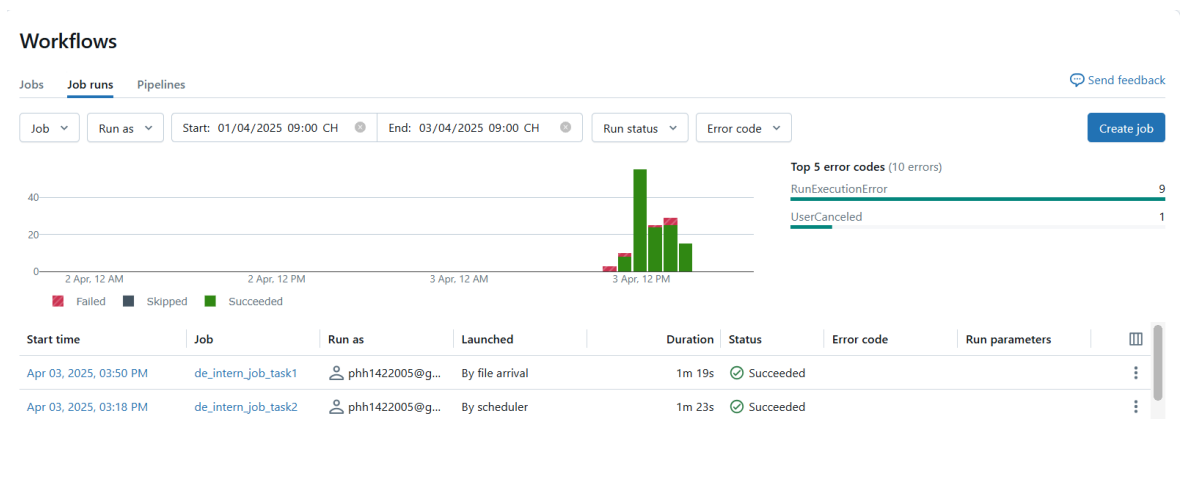
Item	Time	Type
Workspace Usage Dashboard /Users/phh1422005@gmail.com	12 minutes ago	Dashboard
Bakehouse Sales Starter Space /Users/phh1422005@gmail.com	15 minutes ago	Genie space
de_intern_job_task2	38 minutes ago	Job
de_intern_pipeline	39 minutes ago	Pipeline
de_intern_volume3 de_intern_catalog.de_intern_schema	42 minutes ago	Volume

**Catalog:** Quản lý cơ sở dữ liệu, chứa các schema, schema chứa table (dữ liệu có cấu trúc - DeltaTable) hoặc volume (dữ liệu thô) hoặc model. DeltaTable giúp đảm bảo ACID Transactions (đảm bảo commit, rollback), Schema Enforcement (đảm bảo dữ liệu đúng định dạng), TimeTravel (truy vấn các phiên bản trước đó). 1 DeltaTable có metadata definition ở catalog, còn dữ liệu vật lý lưu trữ ở DeltaLake.

The screenshot displays the Databricks Catalog Explorer interface. On the left, a sidebar shows the catalog structure under 'Serverless Starter Warehouse'. The main panel is divided into two sections. The top section shows the 'de\_intern\_volume' details, including its owner, tags, and a list of files: 'department.csv' (114.00 B, 18 hours ago) and 'employee.csv' (22.00 KB, 16 hours ago). The bottom section shows the 'employee\_income' table details, including its type (MANAGED), storage location (Default Storage), and properties (delta: enableDeletionVectors: "true", feature.appendOnly: "supported", feature.deletionVectors: "supported", feature.invariants: "supported", lastCommitTimestamp: "1743654677000", lastUpdateVersion: "0", minReaderVersion: "3", minWriterVersion: "7").

**Workflows:** Công cụ tự động hóa và lên lịch các tác vụ như chạy pipelines và tạo jobs.

The screenshot shows the Databricks Workflows page. It includes a sidebar with 'Jobs', 'Job runs', and 'Pipelines' tabs. The main panel displays a list of jobs with columns: Name, Tags, Created by, Trigger, Recent runs, and Actions. The jobs listed are 'de\_intern\_job\_task1', 'de\_intern\_job\_task2', 'de\_intern\_job\_task3', and 'de\_intern\_job\_trigger\_task1'. The 'Recent runs' column shows the status of the last five runs for each job, with green circles indicating successful runs and red circles indicating failed runs.



**Workflows**

Jobs Job runs Pipelines Send feedback

Filter by pipeline name   ☐ Only my pipelines Type:  Create pipeline

Name	Recent updates	ID	Run as
de_intern_pipeline	⊗ ⊗ ⊗ ⊗ ⊗	0f22506a-b027-414a-b845-d2344b70cc35	phh1422005@gmail.com

**Compute:** Quản lý các cluster và máy chủ tính toán. Ví dụ: SQL Starter Warehouse để chạy truy vấn SQL trên Databricks, tự động bật khi truy vấn/tắt compute khi không truy vấn để tiết kiệm tài nguyên.

**Compute**

SQL warehouses Vector Search Apps Send feedback

Filter SQL warehouses  ☐ Only my SQL warehouses Created by:  Size:  Status:  Create SQL warehouse

Status	Name	Created by	Size	Active / Max	Type
⊗	Serverless Starter Warehouse	phh1422005@gmail.com	Medium	0 / 1	Serverless

**Marketplace:** Nơi cung cấp các ứng dụng, công cụ và giải pháp để tích hợp.

## 2.2.2 SQL

**SQL Editor:** Viết và thực thi truy vấn SQL. Ví dụ:

**Catalog**  ×

For you: ☐ All ☐ My organization

- workspace
- system
- de\_intern\_catalog
- Delta Shares Received
- samples

**New Query 2025-04-03 9:46pm** +

Run (1000)  workspace: default ⋮ ☆ Serverless Starter... Serverless M Save\* Schedule Share

```
1 | GRANT ALL PRIVILEGES ON CATALOG de_intern_catalog TO 'huyt39034@gmail.com';
```

**Raw results** + 🔍 🔼 🔽 📄 🔗 ×

result
1   GRANT ALL PRIVILEGES ON CATALOG de_intern_catalog TO 'huyt39034@gmail.com' was successfully executed.

Xin cấp tất cả các quyền (ALL PRIVILEGES) trên catalog mới tạo (có tên de\_intern\_catalog) cho người dùng có địa chỉ email là huyt39034@gmail.com.

**Queries:** Những truy vấn mà người dùng lưu lại trong SQL Editor.

Queries

Open editor

Create query

Filter queries

My queries

Favorites

All queries

Name	Tags	Created by	Created at
New Query 2025-04-03 11:14am		phh1422005@gmail.com	Apr 03, 2025, 11:14 AM

1-1 < Previous Next >

**Dashboards:** Tạo và hiển thị báo cáo dữ liệu.

**Genie:** Sinh truy vấn SQL dựa trên câu hỏi.

New Space

huyt39034@gmail.com

"Write a complex SQL query on Databricks to analyze data from the de\_intern\_catalog.de\_intern\_schema.employee\_income table. The query should perform the following steps:  
  
Filter data: Select only employees whose salary is above the average salary of all employees.  
  
Categorize income: Classify income into three groups:  
  
'Low' (below 40% of the highest salary).  
  
'Medium' (between 40% and 80% of the highest salary).  
  
'High' (above 80% of the highest salary).  
  
Department statistics: Calculate the total number of employees, the average salary, and the number of employees in the 'High' income category for each department.  
  
Sort results: Display the department list in descending order based on the number of 'High' income employees.  
  
Ensure that the query is optimized and makes use of advanced SQL functions such as AVG(), CASE, COUNT(), GROUP BY, and ORDER BY."

Genie

This analysis provides insights into employee salaries by department, focusing on those earning above average. It categorizes these employees into low, medium, and high income levels based on their salaries relative to the maximum salary in the organization. The results include the total number of employees, the average salary, and the count of high-income employees for each department, sorted by the number of high-income employees.

What is the average salary of employees in each department? How many employees are in each in

Ask your question...

Always review the accuracy of responses.

Send feedback + New chat Monitoring Configure Share

Context Settings

Data Instructions SQL Queries

Data

Link existing or upload new data to start discovering insights.

Name

Type

employee\_income

de\_intern\_catalog.de\_intern\_schema

Table

New Space

Genie

This analysis provides insights into employee salaries by department, focusing on those earning above average. It categorizes these employees into low, medium, and high income levels based on their salaries relative to the maximum salary in the organization. The results include the total number of employees, the average salary, and the count of high-income employees for each department, sorted by the number of high-income employees.

employee\_income

	department_name	total_employees	average_salary	high_income_employees
1	Finance	7	147,298	
2	Purchase	9	140,963.67	
3	Operations	7	149,866.86	
4	Technology	6	138,102.5	
5	Marketing	2	146,032.5	
6	Sales	6	125,287.17	

6 rows

1

WITH filtered\_employees AS (

2

SELECT

3

\*

4

FROM

5

de\_intern\_catalog.de\_intern\_schema.employee\_income

6

WHERE

What is the average salary of employees in each department? How many employees are in each in

Ask your question...

Always review the accuracy of responses.

Send feedback + New chat Monitoring Configure Share

Context Settings

Data Instructions SQL Queries

Data

Link existing or upload new data to start discovering insights.

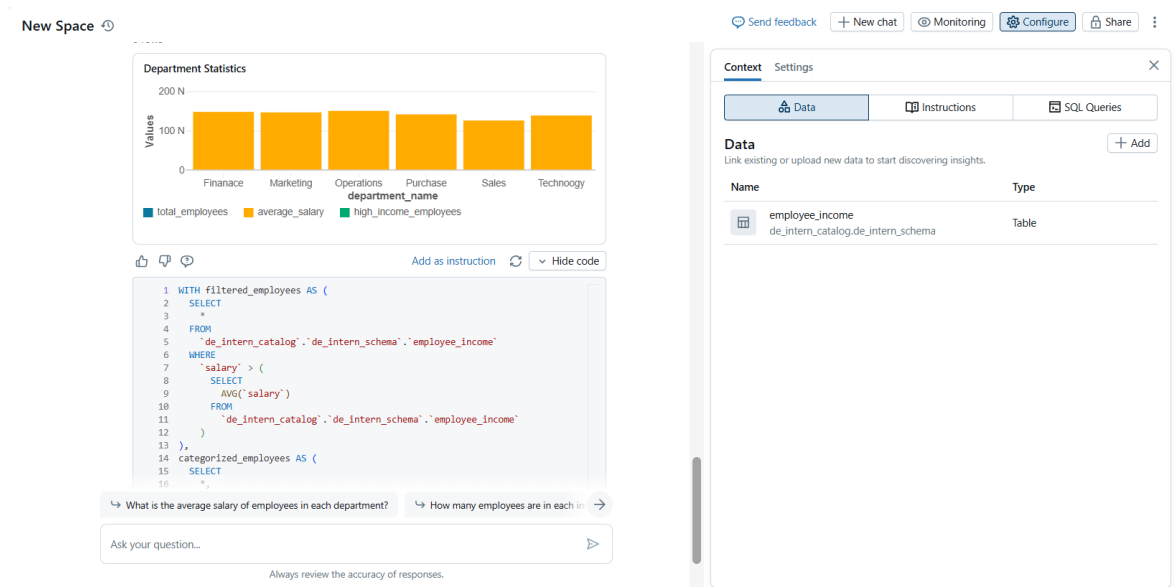
Name

Type

employee\_income

de\_intern\_catalog.de\_intern\_schema

Table



**Alerts:** Thiết lập cảnh báo khi có thay đổi dữ liệu bất thường.

**Query History:** Thông tin tất cả câu lệnh trong notebook và truy vấn SQL đã thực hiện.

Query History

User: Me (phh1422005@gmail.com) Last 7 days Compute Duration Status Statement Statement ID 100+ queries Reset filters

Query	Started at	Duration	Source	Compute	User
> final_df.write.format("delta").mode("overwrite")...	4/03/2025, 03:5...	6.74 s	> de_inter...ob_task1 / ... / Cell(ID...190): 55	Serverless comp...	phh1422005@gm...
> salary_stats = employee_df.select( min("salary")...	4/03/2025, 03:5...	6.12 s	> de_inter...ob_task1 / ... / Cell(ID...190): 32	Serverless comp...	phh1422005@gm...
> final_df.write.format("delta").mode("overwrite")...	4/03/2025, 03:1...	2.47 s	> de_inter...ob_task2 / ... / Cell(ID...190): 55	Serverless comp...	phh1422005@gm...
> salary_stats = employee_df.select( min("salary")...	4/03/2025, 03:1...	1.01 s	> de_inter...ob_task2 / ... / Cell(ID...190): 32	Serverless comp...	phh1422005@gm...
> df_new.coalesce(1).write.mode("overwrite").optio...	4/03/2025, 03:1...	805 ms	> de_inter...ob_task2 / ... / Cell(ID...192): 54	Serverless comp...	phh1422005@gm...
> department_ids = [row["department_id"] for row i...	4/03/2025, 03:1...	743 ms	> de_inter...ob_task2 / ... / Cell(ID...192): 25	Serverless comp...	phh1422005@gm...
> select sum(id) from de_intern_catalog.de_intern_...	4/03/2025, 03:1...	6.08 s	New Query 2025-04-03 11:17am	Serverless Starte...	phh1422005@gm...
> select sum(id) from de_intern_catalog.de_intern_...	4/03/2025, 03:1...	414 ms	New Query 2025-04-03 11:17am	Serverless Starte...	phh1422005@gm...

**SQL Warehouses:** Tự động bật tắt compute khi truy vấn.

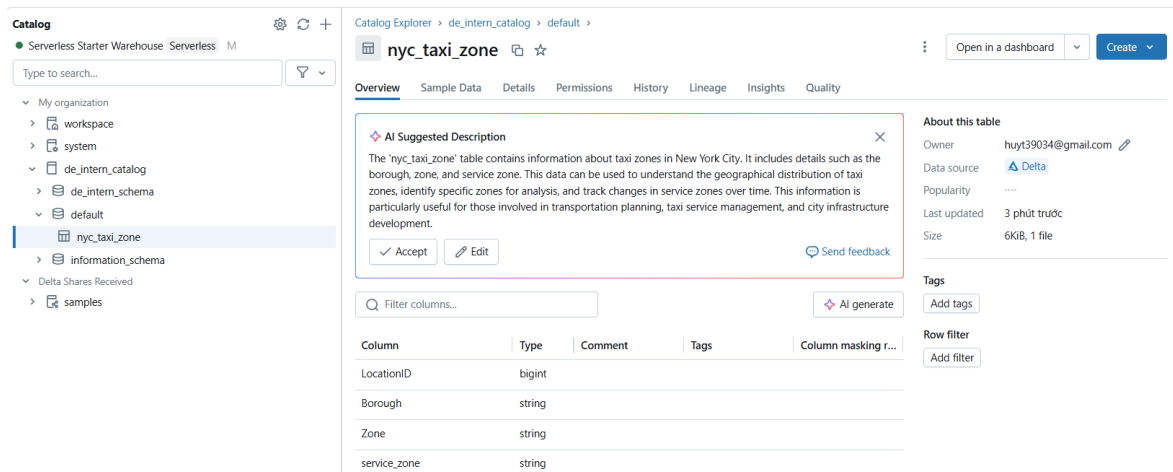
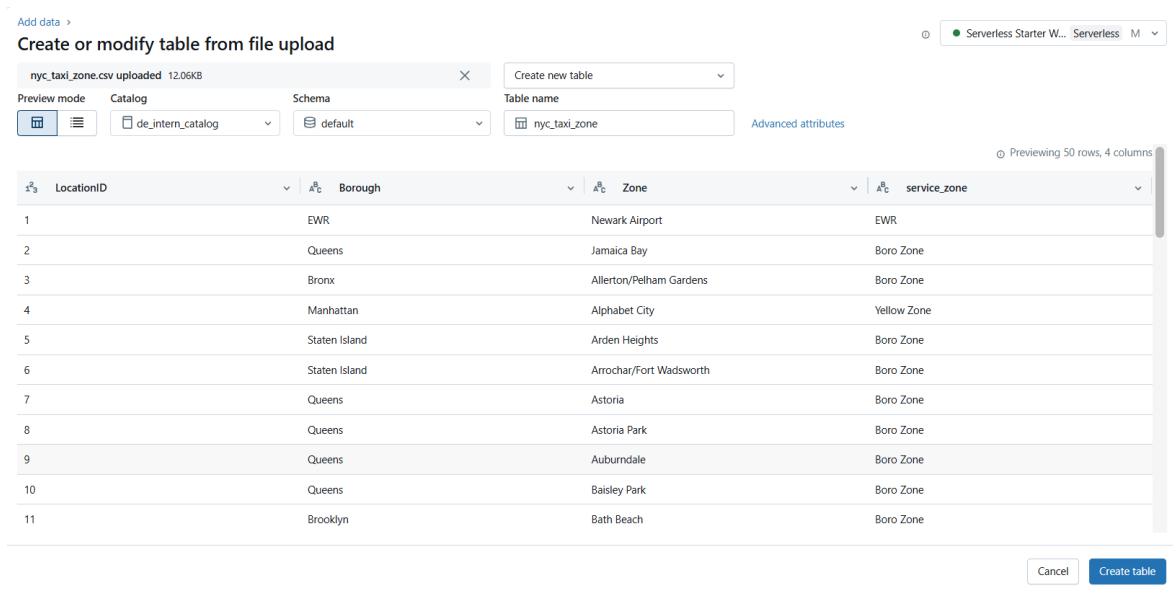
## 2.2.3 Data Engineering

**Jobruns:** Quản lý và theo dõi các jobs đã chạy.

**Data Ingestion:** Quá trình nhập dữ liệu từ các nguồn khác nhau vào Databricks.

- *Upload Tabular Data Files:* Tải lên các tệp dữ liệu có cấu trúc (như CSV) để tạo hoặc thay thế một bảng DeltaTable.





- Upload Files to a Volume: Thêm dữ liệu thô vào volume.
- Create Table from Amazon S3: Tạo bảng DeltaTable từ dữ liệu trong các object lưu trữ trên Amazon S3.

**Pipelines:** Quản lý quy trình xử lý dữ liệu tự động.

# Chương 3

## Nội dung thực tập

### 3.1 Xây dựng quy trình xử lý dữ liệu trên nền tảng AWS Glue

AWS Glue là dịch vụ ETL không máy chủ (serverless) giúp tự động hoá quá trình trích xuất, biến đổi và tải dữ liệu. Trong quá trình thực tập, em đã tham gia thiết kế và xây dựng workflow xử lý dữ liệu bao gồm ba thành phần chính: khai báo metadata job trong DynamoDB, cấu hình Glue Job, và thực thi job qua AWS Lambda và Step Functions. Dưới đây là chi tiết quy trình:

#### 3.1.1 Thiết lập và cấu hình Entry Job trong DynamoDB

DynamoDB được sử dụng làm nơi lưu trữ các metadata liên quan đến job, đóng vai trò như một kho tham chiếu trung tâm cho các thông tin đầu vào và cấu hình của từng job. Mỗi bản ghi trong bảng DynamoDB tương ứng với một entry job, bao gồm các trường sau:

- **Job Name:** Tên định danh của job, dùng để ánh xạ và truy xuất khi thực thi.
- **Input Path:** Đường dẫn đến file đầu vào trên S3 hoặc vị trí dữ liệu nguồn.
- **Output Path:** Vị trí lưu kết quả đầu ra sau khi job hoàn tất xử lý.
- **Step Function Name:** Tên của Step Function dùng để gọi job tương ứng.
- **Default Parameters:** Các tham số mặc định như ngày xử lý, vùng dữ liệu, định dạng file,...

- **Job Type:** Loại job (ETL/Report/Validation,...).
- **Trigger Type:** Loại trigger thực thi (manual, scheduled,...).

Việc cấu hình entry job trong DynamoDB đảm bảo hệ thống có thể mở rộng dễ dàng, cho phép người dùng thêm, sửa hoặc cập nhật workflow một cách linh hoạt mà không cần chỉnh sửa trực tiếp mã nguồn. Đồng thời, khi Lambda function được gọi, nó sẽ dựa vào name trong event để truy xuất thông tin từ bảng này và khởi tạo job tương ứng.

### 3.1.2 Thiết lập và cấu hình Glue Job

#### Bước 1: Tạo ETL Job trong AWS Glue

Truy cập AWS Management Console và chọn dịch vụ AWS Glue. Tại phần *ETL Jobs*, có thể tạo job mới hoàn toàn hoặc sử dụng chức năng **Clone Job** để sao chép từ một job đã được thiết lập trước. Việc clone giúp tiết kiệm thời gian cấu hình lại các thông tin như thư viện, môi trường và thông số mặc định.

#### Bước 2: Cấu hình chi tiết Job

Sau khi tạo hoặc clone Job, tiến hành cấu hình các thông tin cần thiết:

- **Job Name:** Tên định danh duy nhất của job.
- **IAM Role:** Role cho phép job truy cập các tài nguyên AWS như S3, DynamoDB,...
- **Glue Version:** Phiên bản runtime cho job (ví dụ: Glue 3.0).
- **Script Path:** Đường dẫn mã ETL script được lưu trên S3.
- **Python library dependencies:** Thư viện cần thiết để thực thi ETL logic.
- **Job Parameters:** Các biến động (dynamic) như ngày, ID vùng, loại báo cáo,...

### **Bước 3: Lưu và xác nhận Job**

Sau khi hoàn thành cấu hình, lưu lại job và kiểm tra tổng thể các thông số để đảm bảo việc thực thi diễn ra chính xác.

### **3.1.3 Thực hiện chạy Job thông qua AWS Lambda**

Toàn bộ quy trình chạy Glue Job được điều khiển qua Lambda Function, giúp tự động hóa và linh hoạt trong việc kích hoạt job bằng sự kiện tùy chỉnh.

#### **Bước 1: Cấu hình Lambda Function**

Truy cập AWS Lambda và tìm đến function `fss-orcs-cloud-dev-handle_single_event`. Đây là function chính nhận sự kiện đầu vào, truy xuất dữ liệu từ DynamoDB và kích hoạt Step Function tương ứng để thực thi Glue Job.

#### **Bước 2: Tạo sự kiện JSON đầu vào**

Tạo một test event có định dạng JSON gồm các trường sau:

- "name": Khóa chính tham chiếu đến bản ghi tương ứng trong DynamoDB.
- "params": Tham số động truyền vào như thời gian chạy, ngày dữ liệu,...
- "event\_type": Loại sự kiện, ví dụ "manual" cho chạy thủ công.

#### **Bước 3: Gửi sự kiện đến Lambda**

Thực hiện kích hoạt Lambda với test event đã tạo. Lambda sẽ xử lý logic và gọi Step Function tương ứng dựa trên thông tin đã lưu trong DynamoDB.

#### **Bước 4: Theo dõi Step Function thực thi**

Tại AWS Step Functions, tìm đến luồng xử lý `fss-dev-orcs_glue_job_v1`. Tại đây, có thể giám sát trạng thái thực thi của các bước như `StartExecution`, `Glue Start Job Run`, `Success/Failure`.

### **Bước 5: Xem log và trạng thái Job**

Sau khi Glue Job hoàn tất, truy cập CloudWatch để xem chi tiết log xử lý, kiểm tra số lượng bản ghi, thời gian chạy, lỗi (nếu có) và thông tin về output.

### **Bước 6: Đánh dấu job hoàn thành**

Sau khi job thành công, hệ thống cập nhật trạng thái vào DynamoDB hoặc hệ thống giám sát để ghi nhận quá trình thực thi đã hoàn tất.

#### **3.1.4 Đánh giá hiệu quả và vai trò**

Việc sử dụng AWS Glue kết hợp cùng DynamoDB, Lambda và Step Functions giúp xây dựng hệ thống xử lý dữ liệu hiện đại, tự động và dễ bảo trì:

- Tách biệt metadata job ra khỏi mã nguồn nhờ DynamoDB.
- Linh hoạt chạy thủ công hoặc theo lịch thông qua Lambda + Event.
- Giám sát và theo dõi rõ ràng qua Step Functions và CloudWatch.
- Khả năng mở rộng cao và dễ tích hợp vào hệ thống hiện có.

Mô hình này đặc biệt phù hợp với các hệ thống xử lý dữ liệu lớn, đa nguồn và yêu cầu tùy biến cao trong việc điều phối các pipeline ETL.

## **3.2 Xây dựng quy trình xử lý dữ liệu trên nền tảng Databricks Workflow**

### **3.2.1 Job thủ công**

**Mô tả**

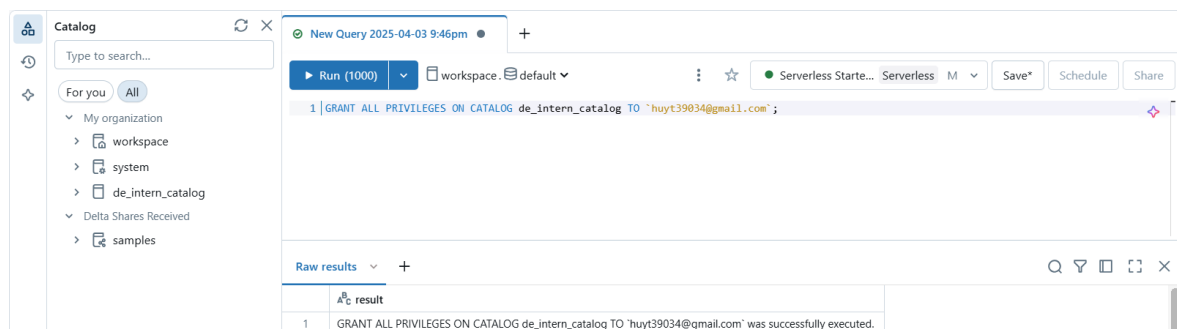
- **Đầu vào:**
  - Bảng employee.csv gồm id, first\_name, last\_name, salary, department\_id.
  - Bảng department.csv gồm department\_id, department\_name.

- **Đầu ra:** Bảng DeltaTable employee\_income gồm: id, fullname, department\_name, salary, income\_level.

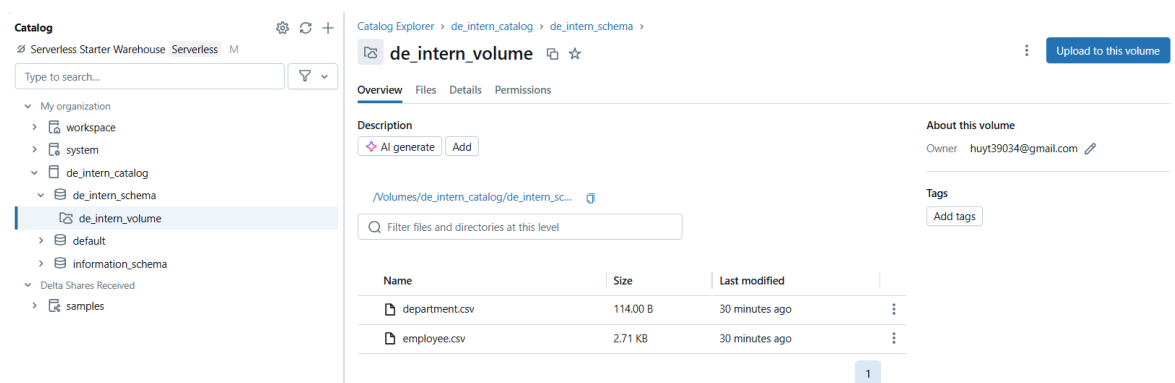
## Quy trình

- Bước 1: Xin cấp tất cả các quyền (ALL PRIVILEGES) trên catalog mới tạo (có tên de\_intern\_catalog) cho người dùng có địa chỉ email là huyt39034@gmail.com.

GRANT ALL PRIVILEGES ON CATALOG de\_intern\_catalog TO tên\_user;



- Bước 2: Tạo catalog de\_intern\_catalog, schema de\_intern\_schema, volume de\_intern\_volume, upload 2 file department.csv và employee.csv. tên\_user;



- Bước 3: Tạo notebook de\_intern\_notebook\_etl và viết script ETL

```
de_intern_notebook_etl Python ☆
File Edit View Run Help Last edit was 46 minutes ago
▶ Run all Connect Schedule (1) Share

1 Python

"""
Import
"""
from pyspark.sql.functions import col, concat_ws, when, regexp_replace, min, max, avg
from pyspark.sql import SparkSession
from pyspark.dbutils import DBUtils

# Khởi tạo SparkSession
spark = SparkSession.builder.appName("Employee Department Processing").getOrCreate()
dbutils = DBUtils(spark)

"""
Extract
"""

# Nhận tham số đầu vào từ Databricks Job Parameters
base_path = dbutils.widgets.get("base_path") # Đường dẫn gốc
input_employee = dbutils.widgets.get("input_employee") # Đường dẫn employee.csv
input_department = dbutils.widgets.get("input_department") # Đường dẫn department.csv
output_table = dbutils.widgets.get("output_table") # Tên bảng lưu trữ

# Đọc dữ liệu từ CSV
employee_df = spark.read.format("csv").option("header", "true").option("inferSchema", "true").load(f"{base_path}/{input_employee}")
department_df = spark.read.format("csv").option("header", "true").option("inferSchema", "true").load(f"{base_path}/{input_department}")

"""
Transform
"""
```

- Bước 4: Tạo job de\_intern\_job1, cấu hình parameters và điền đường dẫn notebook vào path, cấu hình cả Job notifications để nhận thông báo về email khi job chạy.

Workflows > Jobs > **de\_intern\_job1** ☆

Send feedback ⋮ Run now ▾

Runs Tasks

de\_intern\_job1  
...4@gmail.com/de\_intern\_notebook\_etl

+ Add task

Task name\*

Type\*

Source\*

Path\*

Compute\*

Dependent libraries

Parameters

Cancel Save task

**Job details**

Job ID 433292339677568

Creator huyt39034@gmail.com

Run as huyt39034@gmail.com

Tags

Description

Lineage 0 upstream tables, 1 downstream table

**Git**

Not configured

**Schedules & Triggers**

None

**Compute**

Serverless

**Job notifications**

Supported destinations: Email, Microsoft Teams, PagerDuty, Slack, Webhook

Destination	Start	Success	Failure	Duration warning	Streaming backlog
✉ trunquan1832003@gmail.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

☐ Mute notifications for skipped runs

☐ Mute notifications for canceled runs

+ Add notification

Cancel Save

Permissions

huyt39034@gmail.com  
Is Owner

## Bước 5: Chạy job và theo dõi



## Workflows

Jobs **Job runs** Pipelines Send feedback

Job: Run as Start: 02/04/2025 01:00 SA End: 04/04/2025 01:00 SA Run status Error code Create job

1


1

0 2 Apr, 12 PM 3 Apr, 12 AM 3 Apr, 12 PM

Failed Skipped Succeeded

Start time	Job	Run as	Launched	Duration	Status	Error code	Run parameters
Apr 04, 2025, 12:18 AM	de_intern_job1	huyt39034@gmail.c...	Manually	1m 18s	Succeeded		

[216242942714519] Finished run 693419234036434 of 'de\_intern\_job1' Inbox x Print Share

 prod-monitoring@databricks.com to me 1:44 AM (0 minutes ago) Star Smiley Reply More



### A run of this job has completed successfully

#### Run details

Workspace	<a href="#">workspace [216242942714519]</a>
Job	<a href="#">de_intern_job1 [433292339677568]</a>
Job Run	<a href="#">693419234036434</a>
Status	<span>✓ Succeeded</span>

- Bước 6: Thực hiện truy vấn trên SQL Editor kiểm tra dữ liệu trên bảng DeltaTable employee\_income sau khi ETL.

Catalog Type to search...

For you All

- My organization
  - workspace
  - system
  - de\_intern\_catalog
- Delta Shares Received
  - samples

New Query 2025-04-03 9:46pm +

Run (1000) workspace: default New SQL editor: OFF Serverless Starte... Serverless M Save\* Schedule Share

1 | select \* from de\_intern\_catalog.de\_intern\_schema.employee\_income

Raw results + Search Filter Table Close

	id	fullname	department_name	salary	income_level
1	29	Jason Olsen	Marketing	51937	low
2	56	Rachael Williams	Finanace	103585	medium
3	24	William Flores	Technooogy	142674	medium
4	39	Linda Clark	Finanace	186781	medium
5	38	Nicole Lewis	Operations	114079	medium
6	15	Anthony Valdez	Operations	96898	low

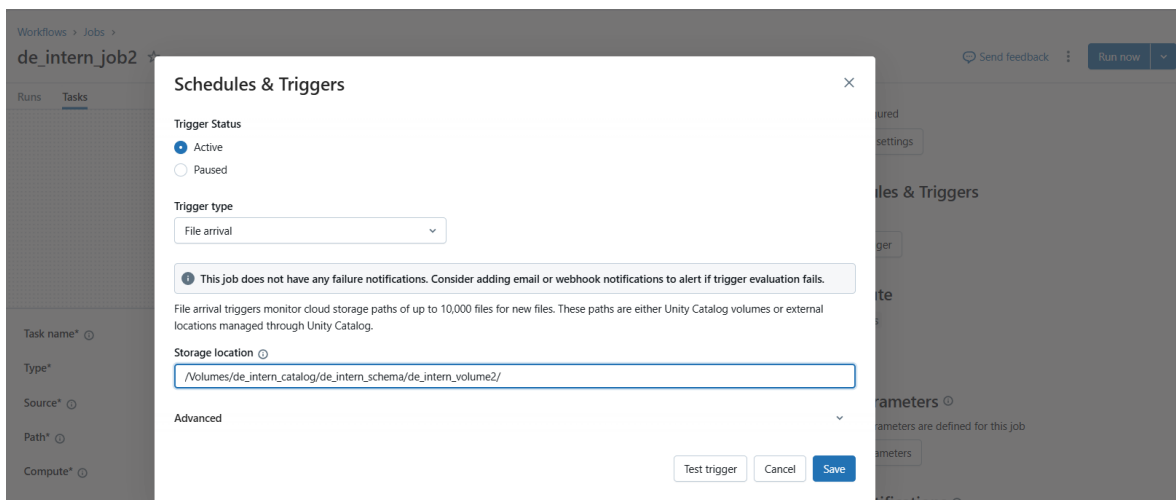
### 3.2.2 Job tự động bằng trigger file arrival (job sẽ tự động chạy mỗi khi volume được chỉ định trong storage location nhận file mới)

#### Mô tả

- **Đầu vào:** Như 3.2.1 File employee.csv được cập nhật theo schedule (giả lập bằng cách random dữ liệu, đổ vào 2 phút một lần).
- **Đầu ra:** Bảng DeltaTable employee\_income gồm: id, fullname, department\_name, salary, income\_level.

#### Quy trình

- Bước 1: Làm các bước tương tự như 3.1 tạo ra jobde\_intern\_job2.
- Bước 2: Tạo trigger file arrival, copy tên đường dẫn volume muốn gắn trigger, save.



- Bước 3: Tạo notebook de\_intern\_notebook\_script để sinh dữ liệu 75 dòng ngẫu nhiên cho employee.csv, sau khi tạo xong sẽ sinh ra một file timestamp.txt vào volume gắn trigger, kích hoạt job de\_intern\_job2 chạy.

```
de_intern_notebook_script Python ☆
File Edit View Run Help Last edit was now

import os
import shutil
import time
from datetime import datetime

# Nhận params từ Databricks Job
department_path = dbutils.widgets.get("department_path")
dest_folder = dbutils.widgets.get("dest_folder")

output_path = f"{dest_folder}/employee.csv"
temp_path = f"{dest_folder}/temp_employee"

# Tạo timestamp để đặt tên file trigger
timestamp_str = datetime.now().strftime("%Y%m%d_%H%M%S")
trigger_file = f"{dest_folder}/{timestamp_str}.txt"

# Khởi tạo Spark Session
spark = SparkSession.builder.appName("Generate Employee Data").getOrCreate()
```

- Bước 4: Tạo job de\_intern\_script\_job2, cấu hình path là notebook de\_intern\_notebook\_script,

Workflows > Jobs > de\_intern\_script\_job2 ☆

Runs Tasks

de\_intern\_script\_job2

Path\* /Workspace/Users/huyt39034@gmail.com/de\_intern\_notebook\_script

Compute\* Serverless

Dependent libraries + Add

Parameters UI JSON

Key	Value
department_path	/Volumes/de_intern_catalog/de { }
dest_folder	/Volumes/de_intern_catalog/de { }

Notifications + Add

Retries Immediately, at most 3x (4 total attempts)

Cancel Save task

Job details

Job ID 761835313458092

Creator huyt39034@gmail.com

Run as huyt39034@gmail.com

Tags Add tag

Description Add description

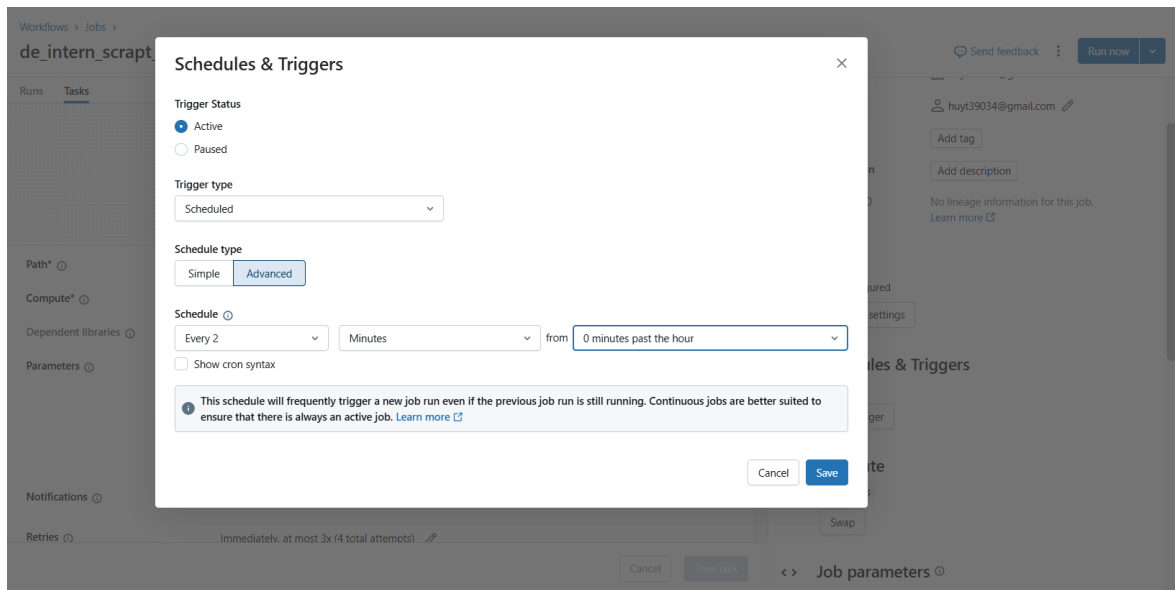
Lineage No lineage information for this job. Learn more

Git Not configured Add Git settings

Schedules & Triggers None Add trigger

Compute Canvaface

- Bước 5: Đặt schedule 2p chạy 1 lần (tại các phút chẵn) cho de\_intern\_script\_job2.



- Bước 6: Theo dõi các job chạy.

**Workflows**

Jobs **Job runs** Pipelines Send feedback

Job: Run as: Start: 02/04/2025 02:00 SA End: 04/04/2025 02:00 SA Run status: Error code: Create job

1

1

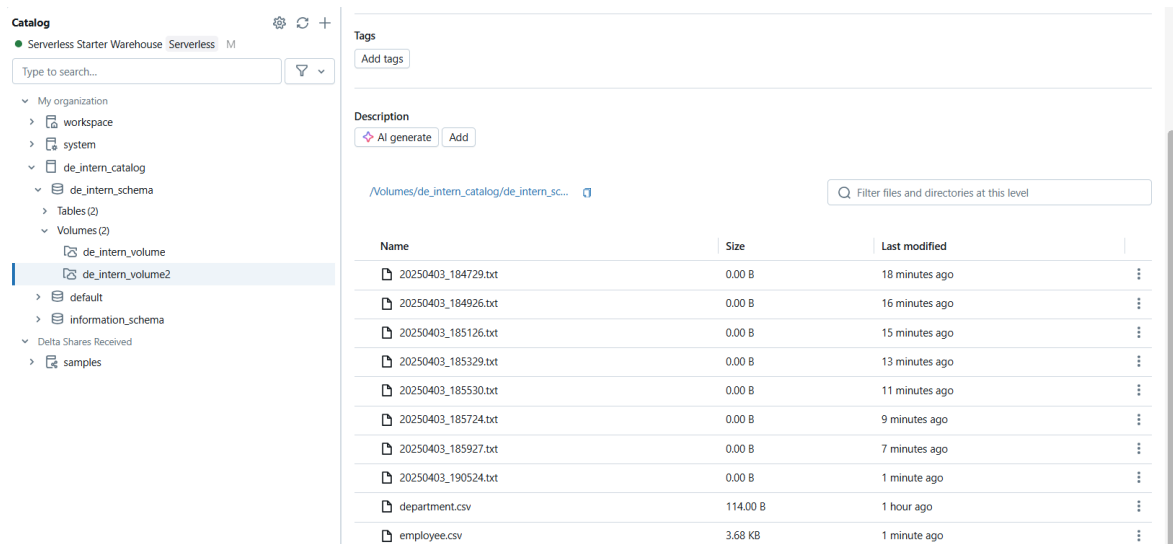
0

2 Apr, 12 PM 3 Apr, 12 AM 3 Apr, 12 PM

Failed Skipped Succeeded

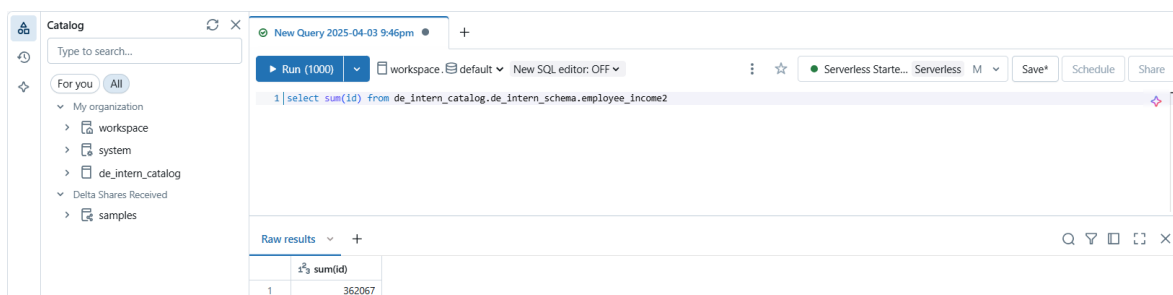
Start time	Job	Run as	Launched	Duration	Status	Error code	Run parameters	
Apr 04, 2025, 01:54 AM	de_intern_job2	huyt39034@gmail.c...	By file arrival	51s	Running			
Apr 04, 2025, 01:52 AM	de_intern_scrapt_job2	huyt39034@gmail.c...	By scheduler	56s	Succeeded			
Apr 04, 2025, 01:52 AM	de_intern_job2	huyt39034@gmail.c...	By file arrival	1m 19s	Succeeded			
Apr 04, 2025, 01:50 AM	de_intern_scrapt_job2	huyt39034@gmail.c...	By scheduler	55s	Succeeded			
Apr 04, 2025, 01:50 AM	de_intern_job2	huyt39034@gmail.c...	By file arrival	1m 9s	Succeeded			
Apr 04, 2025, 01:48 AM	de_intern_scrapt_job2	huyt39034@gmail.c...	By scheduler	1m 12s	Succeeded			
Apr 04, 2025, 01:48 AM	de_intern_job2	huyt39034@gmail.c...	By file arrival	1m 13s	Succeeded			
Apr 04, 2025, 01:46 AM	de_intern_scrapt_job2	huyt39034@gmail.c...	By scheduler	1m 15s	Succeeded			

Theo dõi các file timestamp.txt được tạo ra để kích hoạt job ETL.

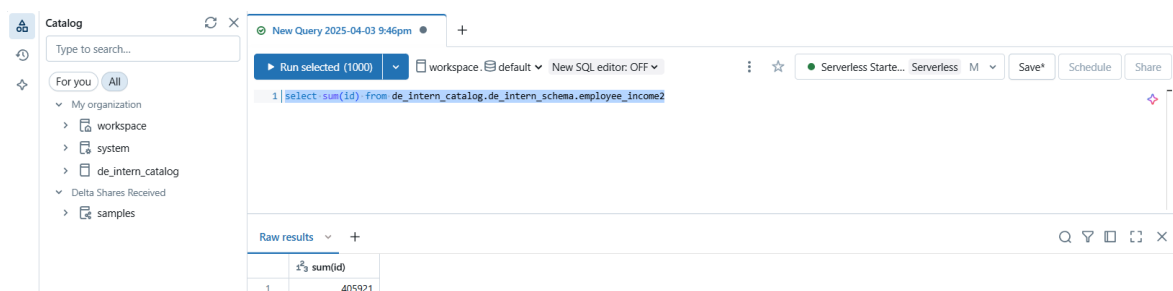


- Bước 7: Truy vấn tổng cột id xem có cập nhật trong bảng DeltaTable sau khi job chạy xong không.

Trước:



Sau:

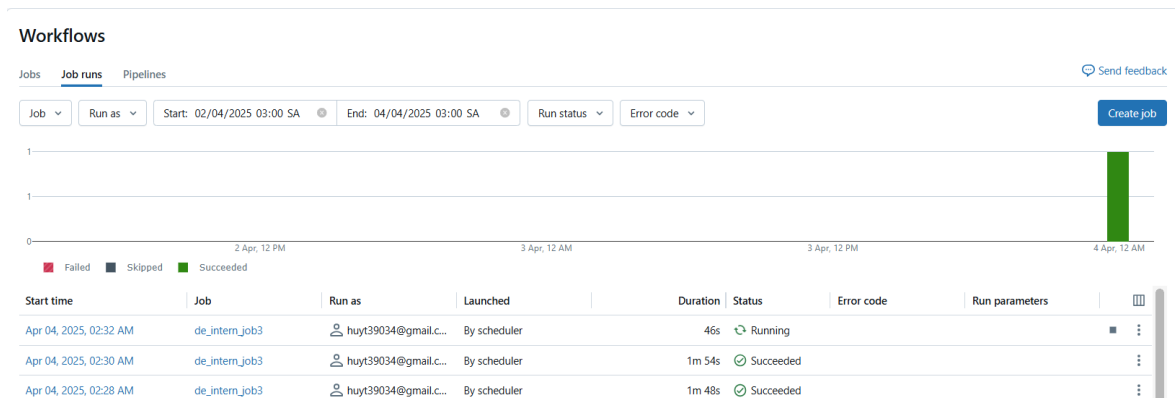


### 3.2.3 Job tự động bằng depend on

- Bước 1: Tạo notebook mới, chỉnh sửa script không cần sinh ra timestamp.txt

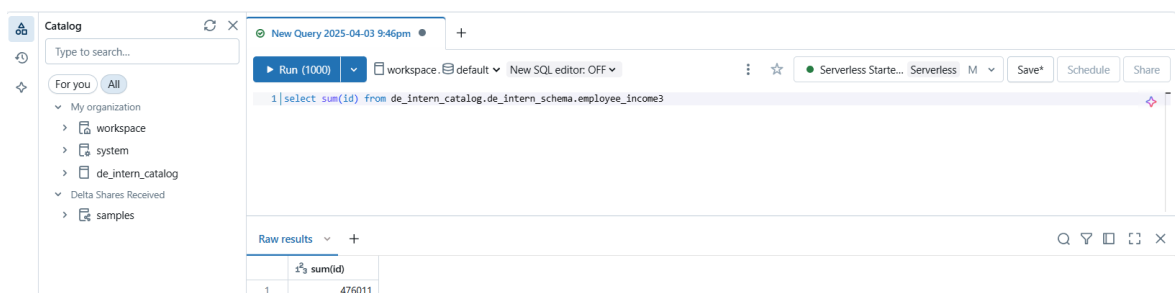


- Bước 4: Đặt schedules 2p và theo dõi job chạy

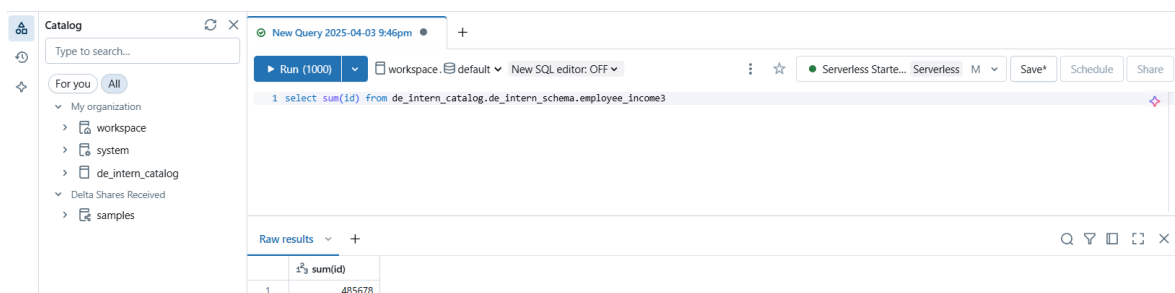


- Bước 5: Kiểm tra xem job có cập nhật vào DeltaTable không.

Trước:



Sau:



# Kết luận

## Tổng kết quá trình thực tập

Quá trình thực tập tại công ty là cơ hội giúp em tiếp cận và làm việc thực tế với các công nghệ hiện đại trong lĩnh vực xử lý và phân tích dữ liệu. Em không chỉ được củng cố kiến thức nền tảng, mà còn được trực tiếp triển khai các quy trình xử lý dữ liệu trên nền tảng AWS và Databricks – hai công cụ phổ biến và mạnh mẽ trong ngành công nghệ hiện nay.

## Mục tiêu và định hướng thực tập

Trong suốt quá trình thực tập, em hướng đến ba mục tiêu chính:

- Nắm bắt tổng quan và cơ chế hoạt động của các dịch vụ thuộc hệ sinh thái AWS và Databricks.
- Xây dựng quy trình xử lý dữ liệu: từ thu thập, lưu trữ, đến xử lý và phân tích dữ liệu.
- Tham gia trực tiếp vào dự án thực tế, áp dụng kiến thức để giải quyết bài toán cụ thể của doanh nghiệp.

## Kết quả đạt được

Sau thời gian làm việc, em đã hoàn thành được nhiều nội dung thực tiễn với các kết quả đáng ghi nhận:



1. **Hiểu và sử dụng thành thạo các dịch vụ AWS:** Em đã tiếp cận và triển khai thành công các dịch vụ như Amazon S3, Athena, Glue, DynamoDB, Lambda, Step Functions, CodePipeline và CloudWatch để phục vụ cho việc xử lý và quản lý dữ liệu.
2. **Xây dựng quy trình xử lý dữ liệu với AWS Glue:** Em đã thiết lập thành công các Glue Job để trích xuất – chuyển đổi – nạp dữ liệu (ETL), khởi chạy thông qua hàm Lambda, và kiểm soát luồng xử lý bằng Step Functions. Quy trình được đánh giá rõ ràng, có khả năng mở rộng và tự động hóa cao.
3. **Triển khai workflow trên nền tảng Databricks:** Em đã thử nghiệm các hình thức job khác nhau bao gồm: job chạy thủ công, job tự động khi có file mới, và job theo chuỗi phụ thuộc. Em cũng sử dụng thành thạo các thành phần như Workspace, SQL Editor và Data Engineering environment trong Databricks.

## Khó khăn và hạn chế

Trong quá trình thực hiện, em cũng gặp một số khó khăn và hạn chế:

1. **Khó khăn trong việc làm quen với hệ thống cloud:** Các dịch vụ của AWS và Databricks có cấu trúc phức tạp, đòi hỏi thời gian để tìm hiểu cách hoạt động cũng như cách kết nối giữa các thành phần.
2. **Chưa tối ưu hoá được toàn bộ quy trình:** Do thời gian thực tập có giới hạn, em chưa có cơ hội tối ưu hiệu suất và chi phí vận hành hệ thống ở mức cao nhất, cũng như chưa đi sâu vào các bài toán nâng cao như xử lý dữ liệu lớn ở quy mô phân tán.

## Định hướng phát triển

Từ kinh nghiệm thực tế tích lũy được, em mong muốn tiếp tục theo đuổi lĩnh vực Data Engineering. Một số định hướng cụ thể của em bao gồm:

- Tìm hiểu sâu hơn về kiến trúc hệ thống xử lý dữ liệu lớn (Big Data).

- Triển khai các pipeline dữ liệu theo chuẩn CI/CD kết hợp các công cụ như Airflow, dbt.
- Tối ưu hoá chi phí và hiệu suất khi xử lý dữ liệu trên các nền tảng cloud.

Kỳ thực tập không chỉ giúp em tích lũy được kiến thức và kỹ năng thực tế mà còn là bước đệm quan trọng trong hành trình nghề nghiệp. Em tin rằng những gì đã học được trong thời gian này sẽ là nền tảng vững chắc cho các bước phát triển tiếp theo trong tương lai.