Project by Gero Krikawa  
 **End to End Data Engineering Project in AWS using Spark (Pyspark)**

Used Technologies:

* AWS Cloud
* Infrastructure as a Code
* AWS Glue:
  + AWS Glue Data Catalog
  + AWS Glue ETL Datapipelines (CSV File 🡪 AWS Redshift,
    - Apache Spark
    - Jupyter Notebook, Pyspark (Data Cleaning, Transformation)
* AWS Redshift

Inhalt

[2 Overview: Data Engineering Lifecycle 1](#_Toc158560103)

[3 AWS ETL Architektur for Data Engineering 2](#_Toc158560104)

[3.1 Project Workflow 2](#_Toc158560105)

[4 Screenshots and explanation during development 3](#_Toc158560106)

[4.1 Create Data Engineering System in AWS using „Infrastructure as Code“ 3](#_Toc158560107)

[4.2 Upload csv in S3 6](#_Toc158560108)

[4.3 Create database 7](#_Toc158560109)

[4.4 Create Crawler for tables with data souce sales\_record.csv 7](#_Toc158560110)

[4.5 Run the crawler 12](#_Toc158560111)

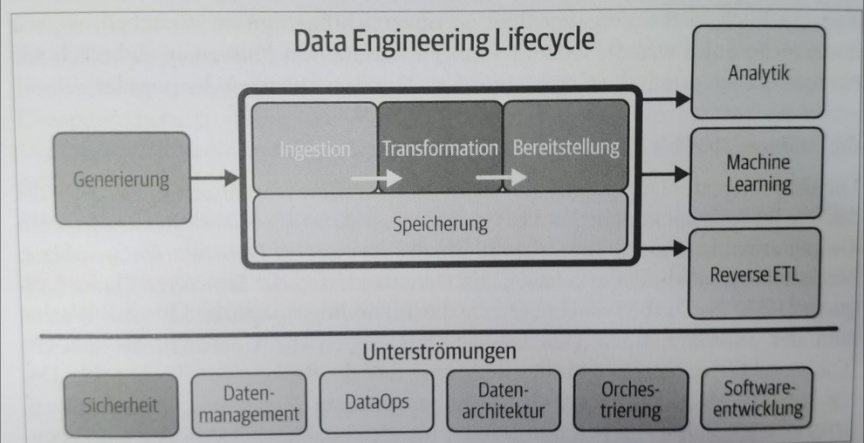
[4.6 ETL Job as interactive jupyter Notebook using pyspark 13](#_Toc158560112)

[4.6.1 Jupyter Notebook for ETL Process using Spark / Pyspark – Step by Step as interactive Notebook 14](#_Toc158560113)

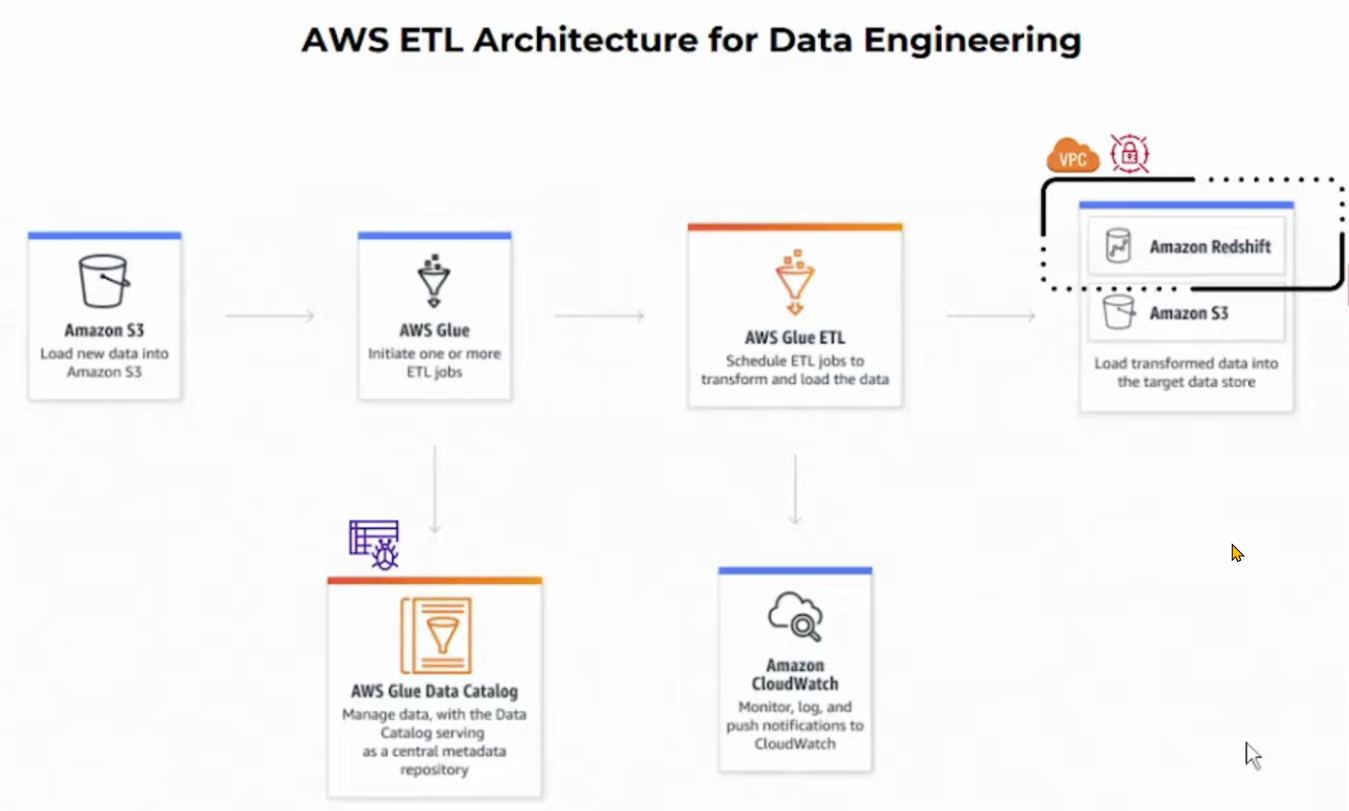
[4.6.2 Jupyter Notebook as Pyspark Script that could be sheduled 20](#_Toc158560114)

[5 AWS Redshift 21](#_Toc158560116)

# Overview: Data Engineering Lifecycle



# AWS ETL architecture for Data Engineering

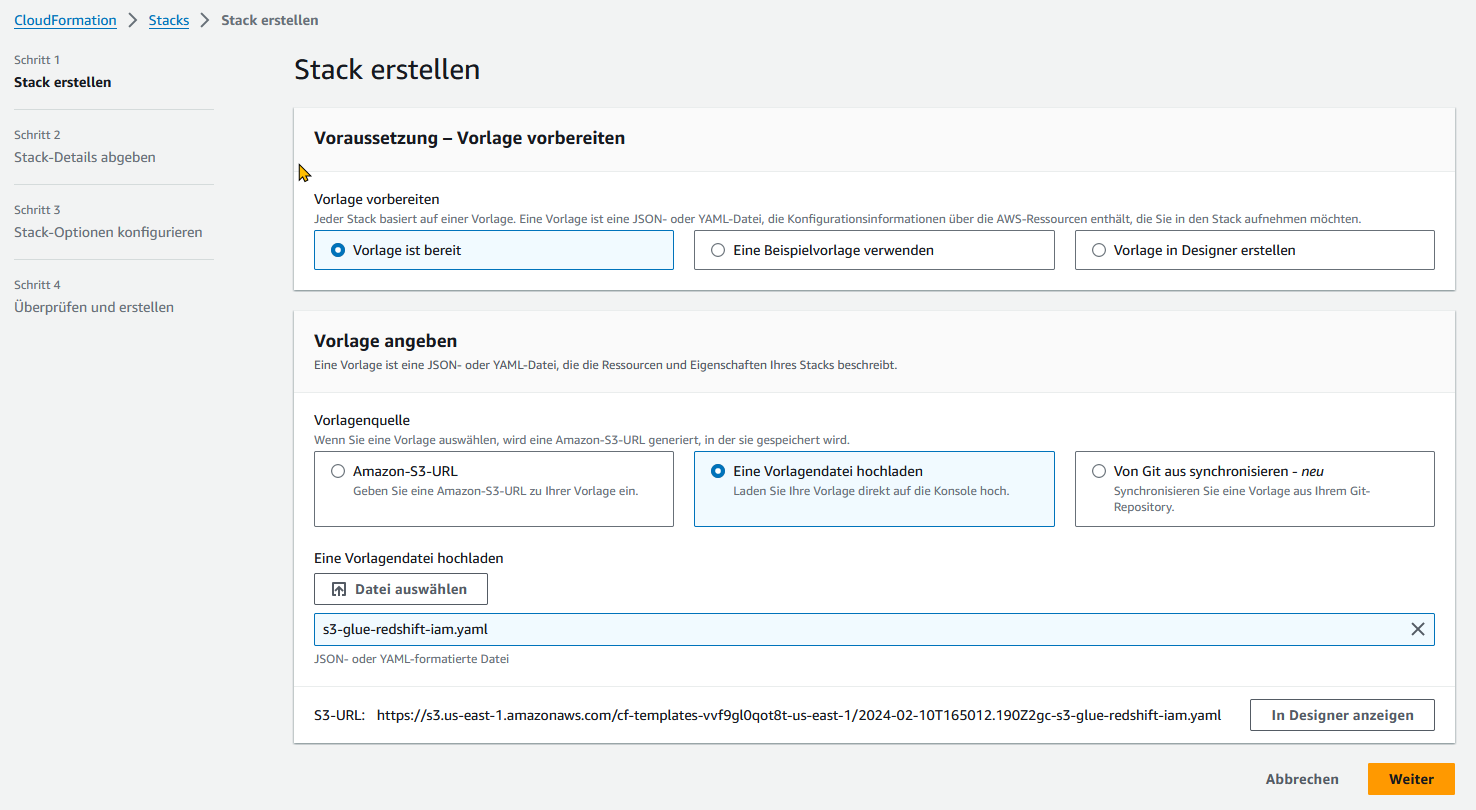


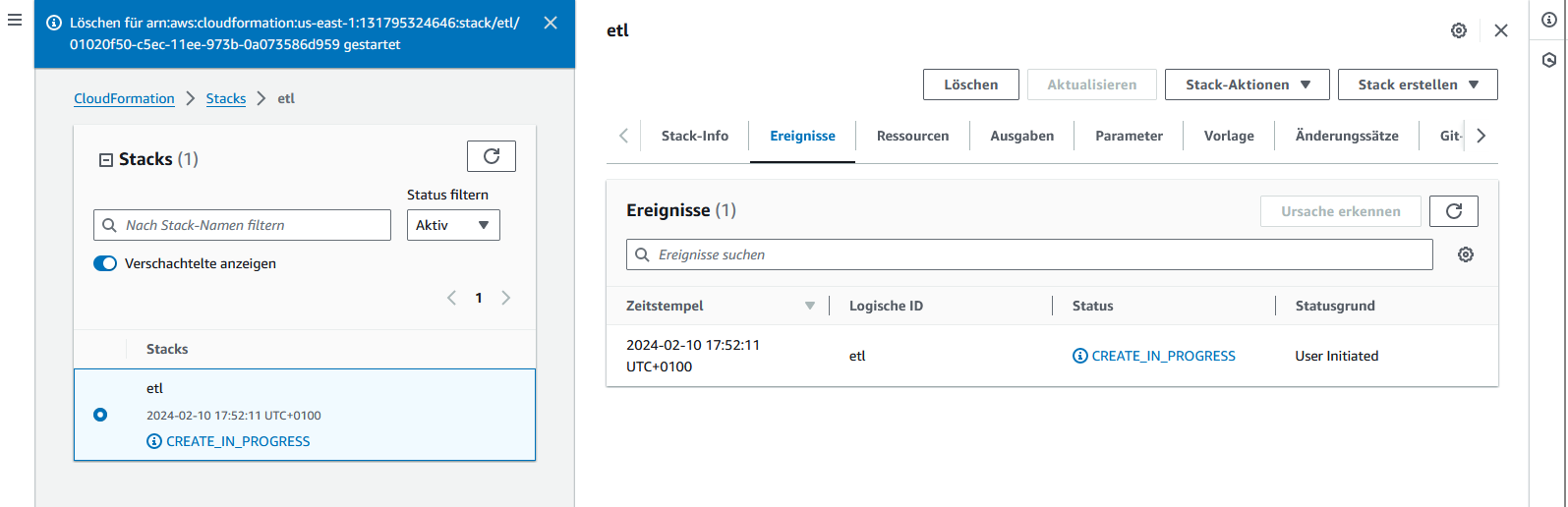
## Project Workflow

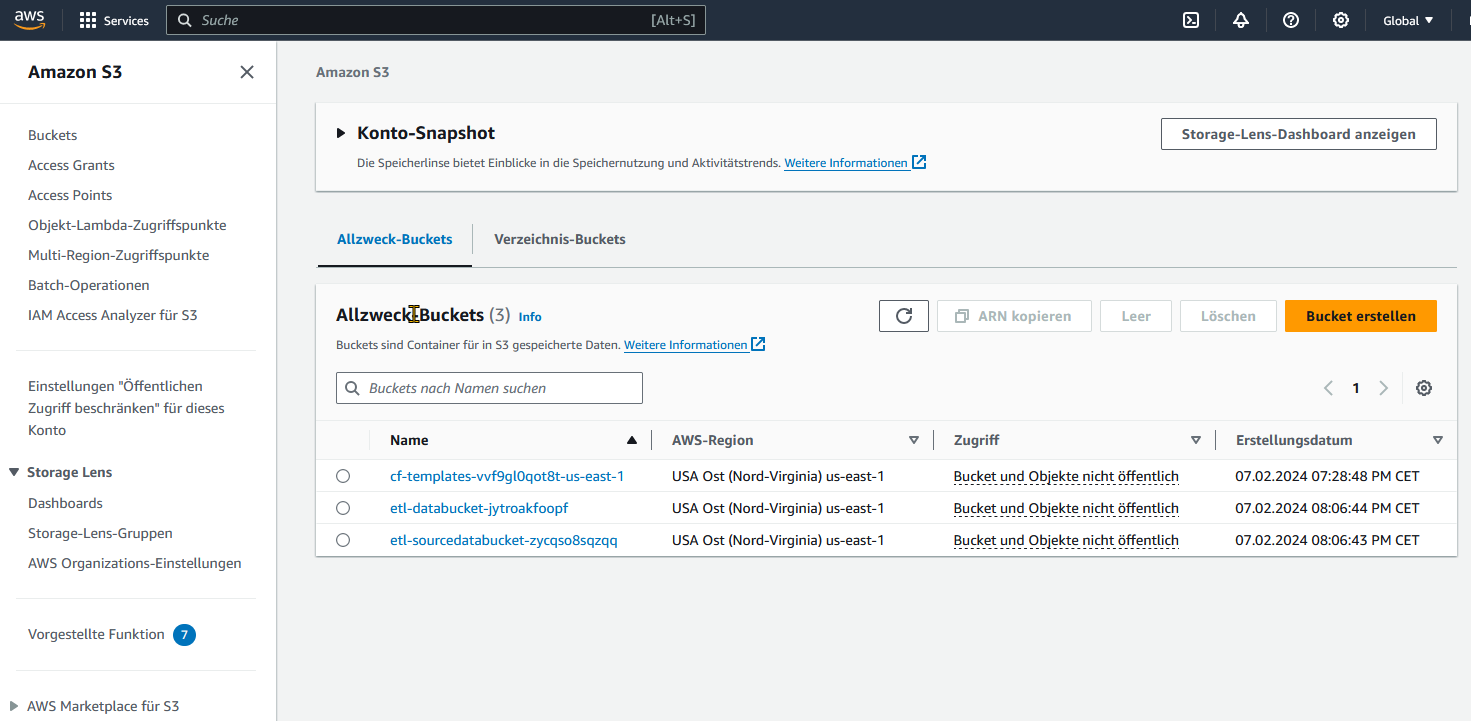
* Create Data Engineering System in AWS using „Infrastructure as Code“
* End to End Data Pipeline
  + Data Source File: sales\_records.csv
  + Destination: AWS Redshift
  + S3 Storage & Source of Data Pipeline
  + AWS Clue – using crawler to catalog data
  + Processing data using Pyspark within interactive Jupyter Notebook in Glue
    - Build Data Pipeline in Pyspark using Glue Jupyter interactive Notebook
    - Reading data from s3 storage, processing it in Spark and then loading it into Redshift (using dynamic frames and spark data frames

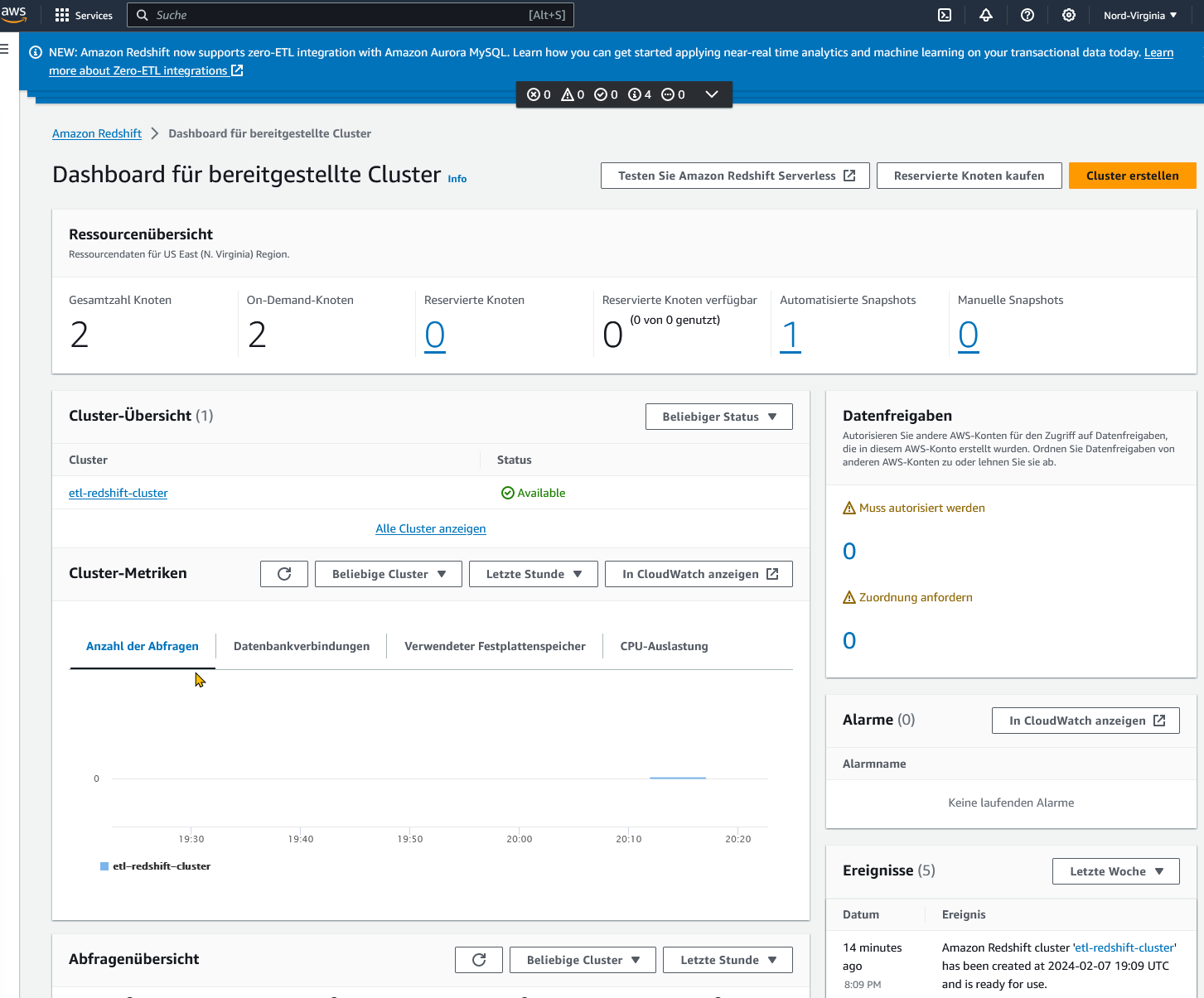
# Screenshots and explanation during development

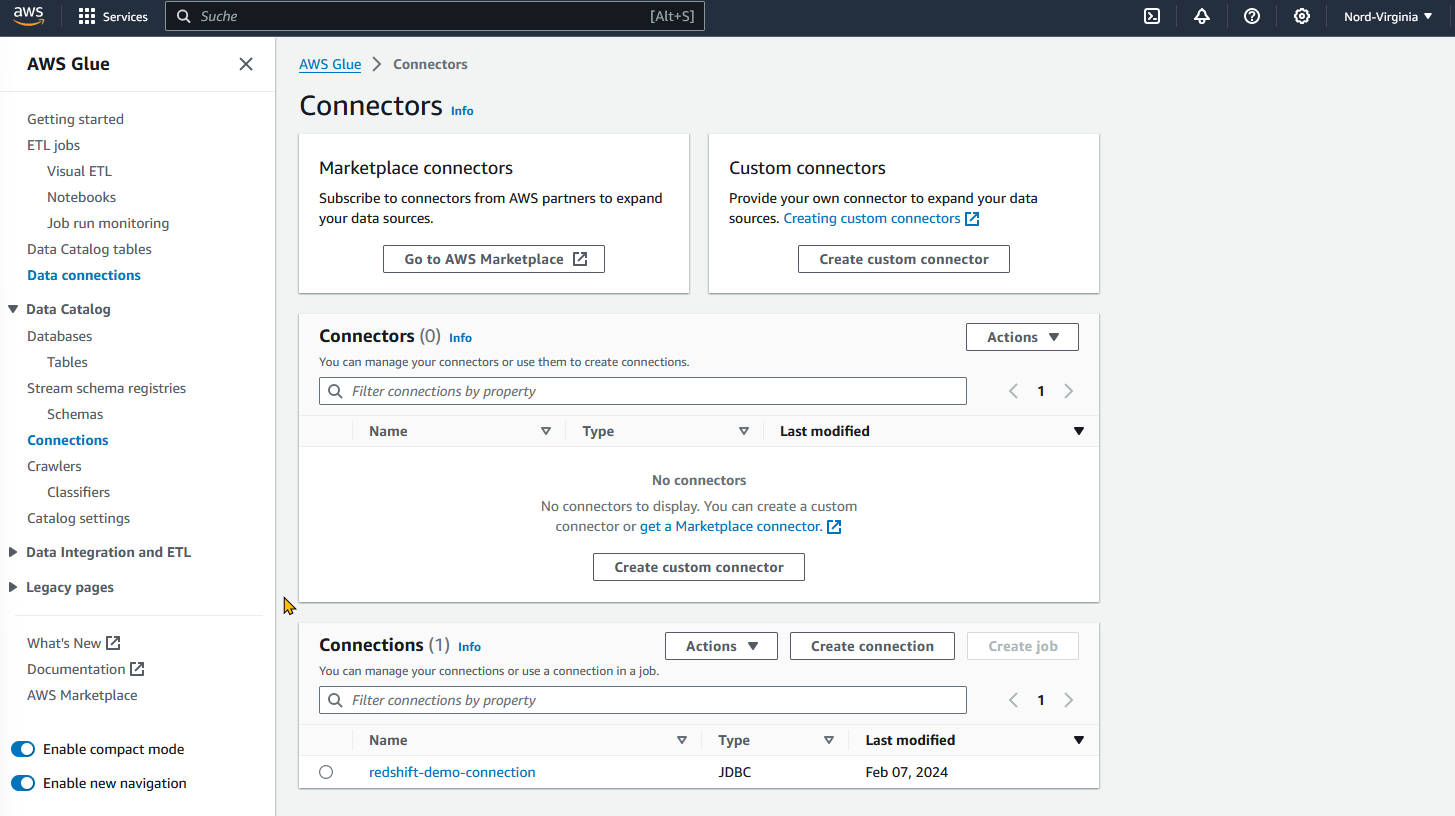
## Create Data Engineering System in AWS using „Infrastructure as Code“

****

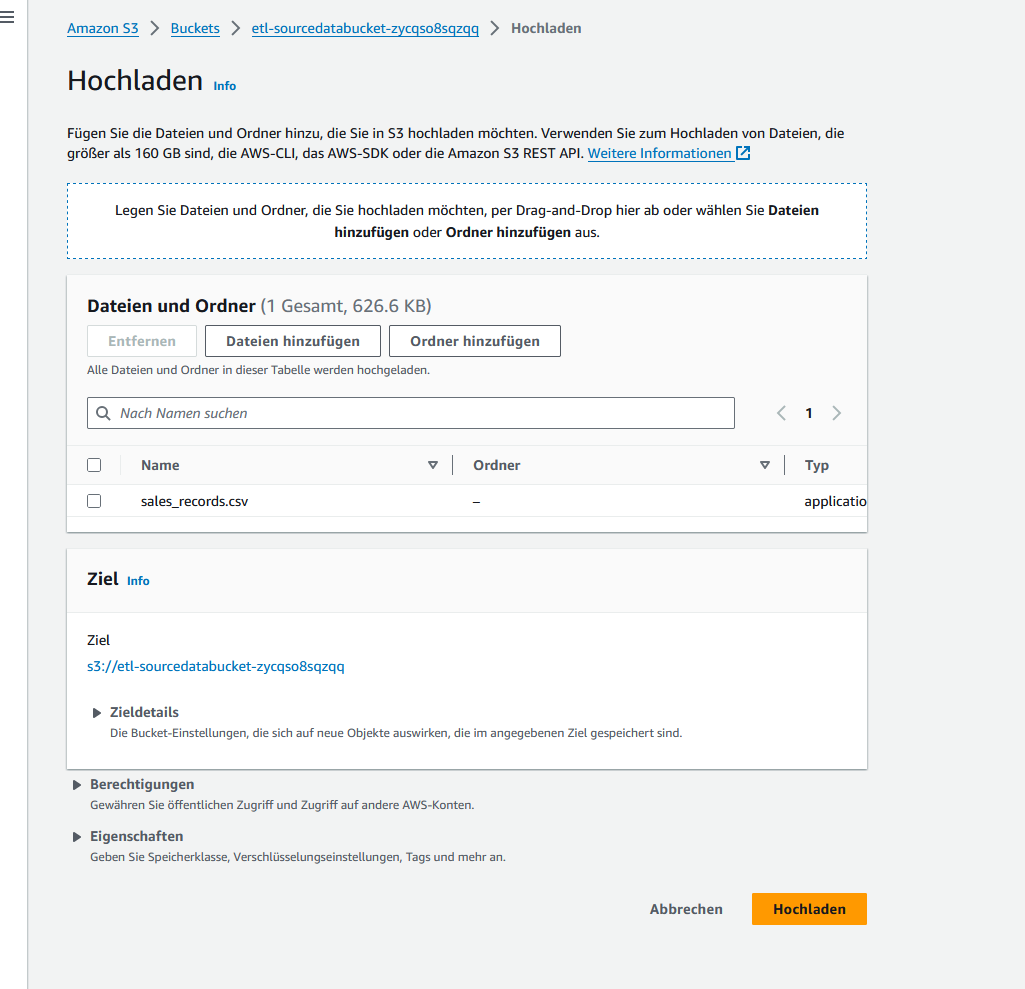
****

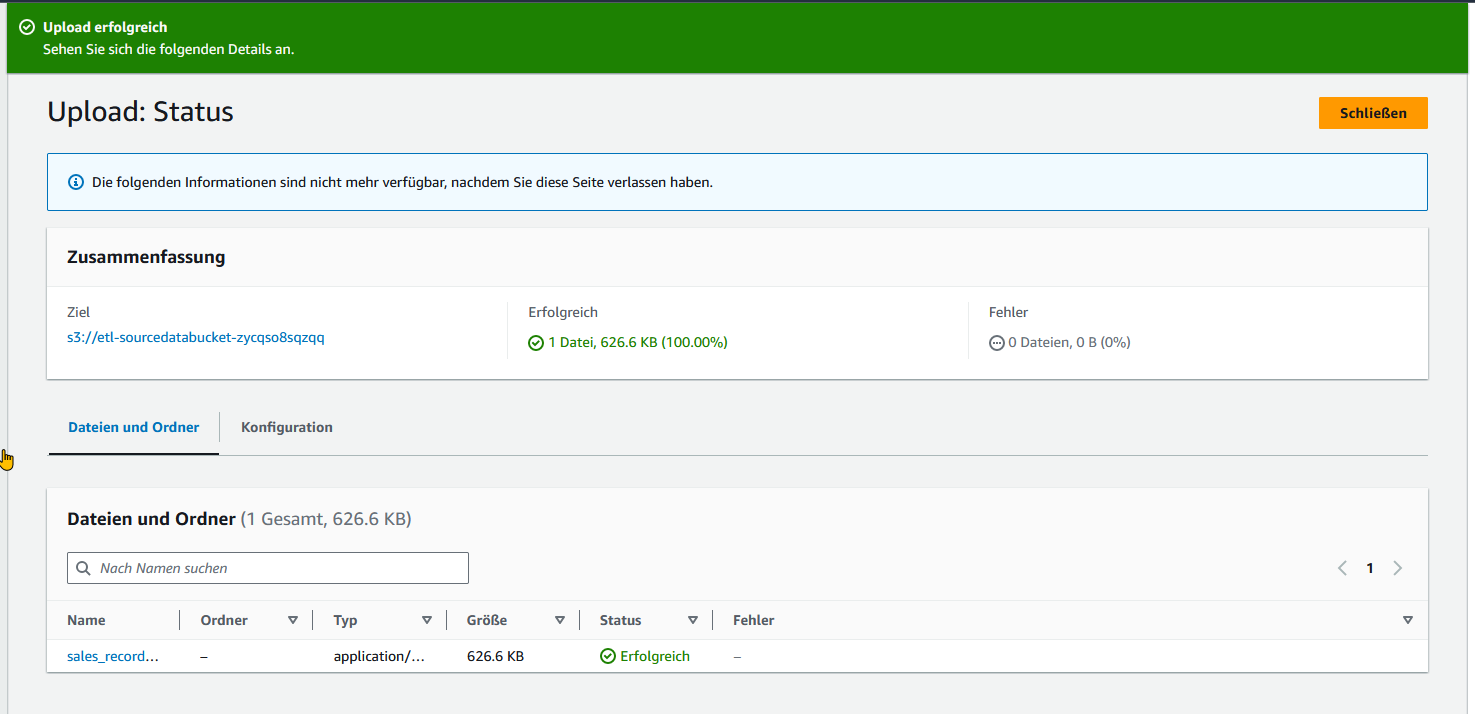




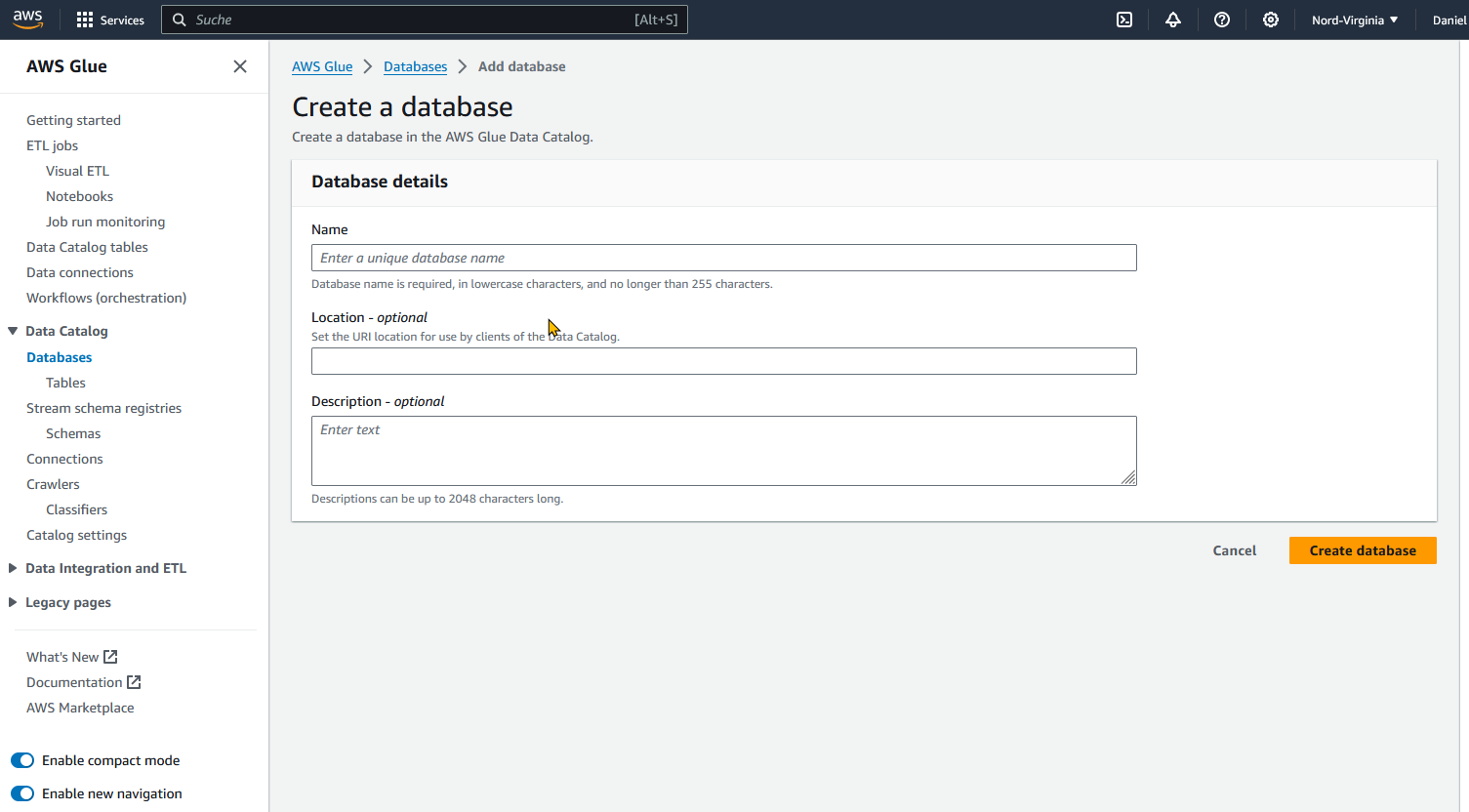


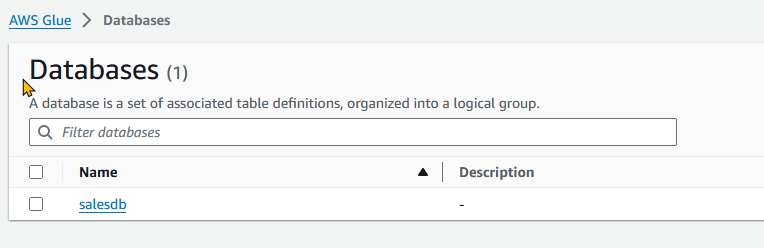
## Upload csv in S3



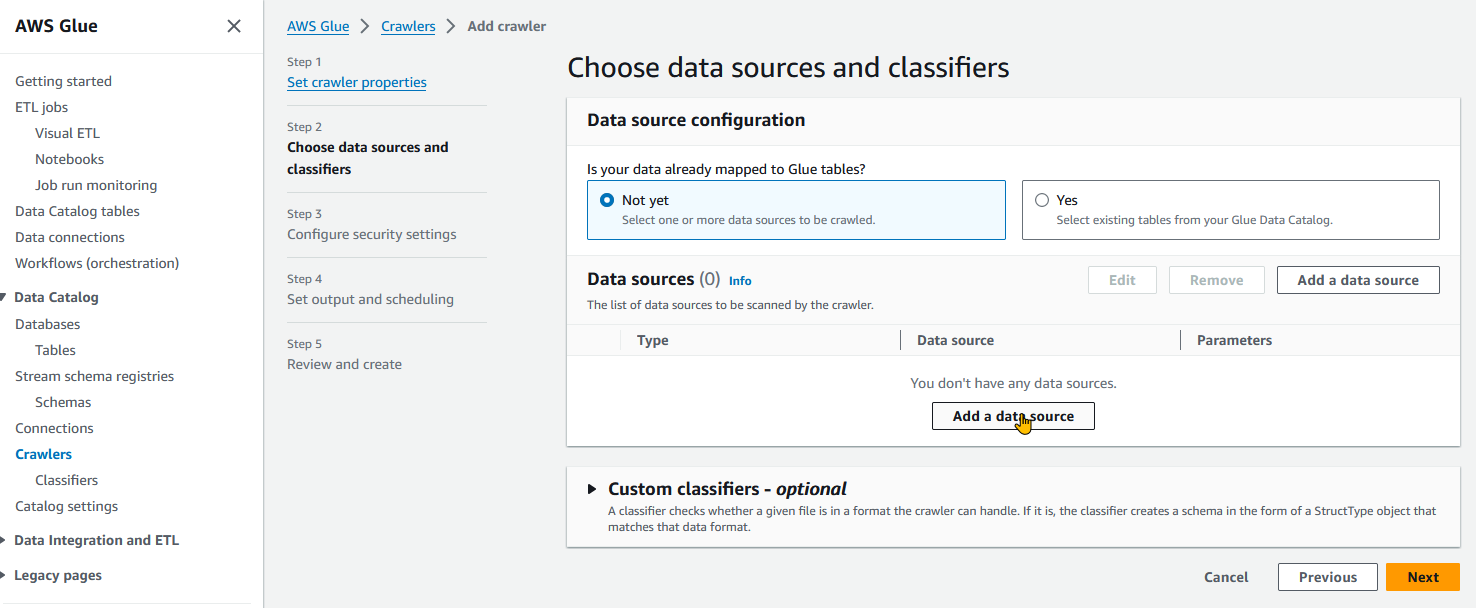


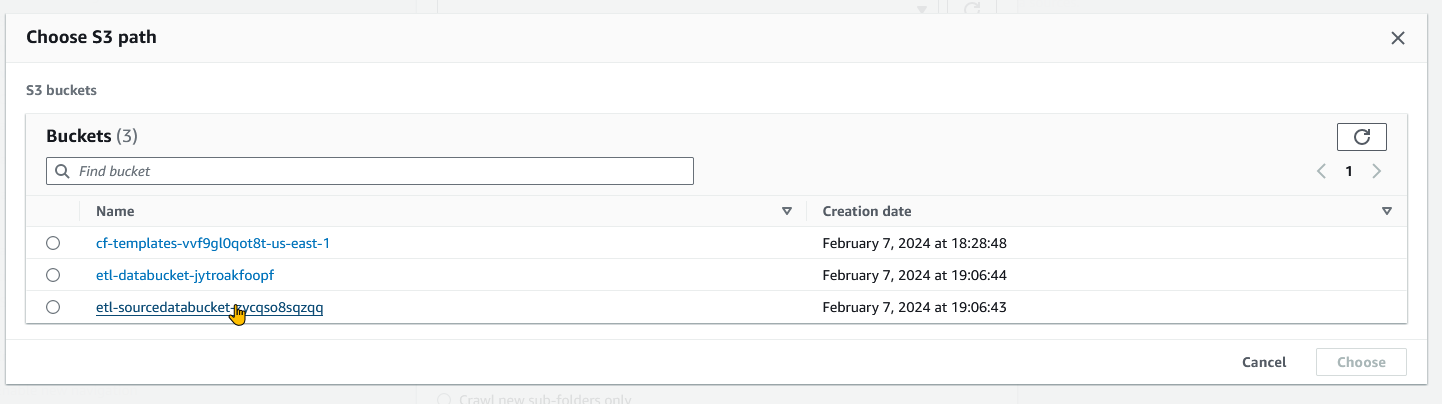
## Create database

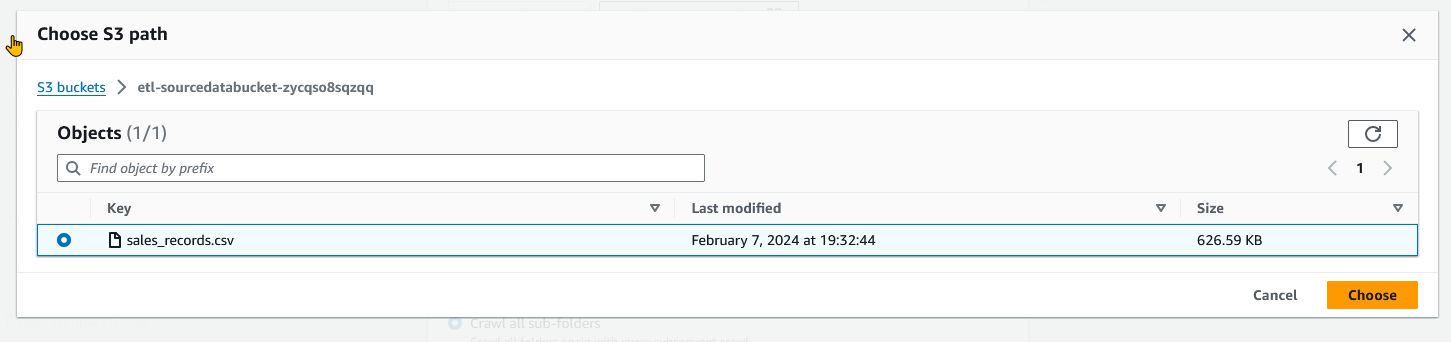


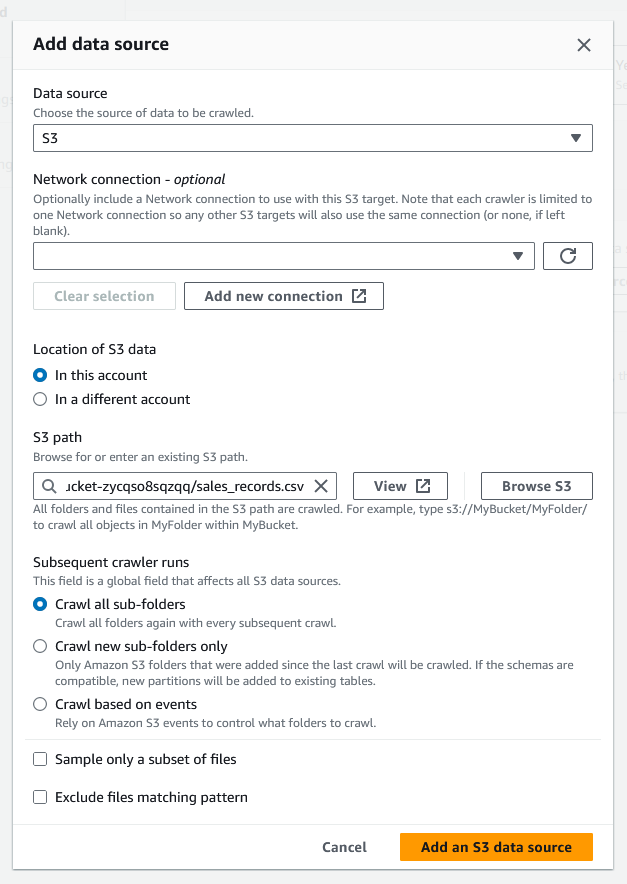


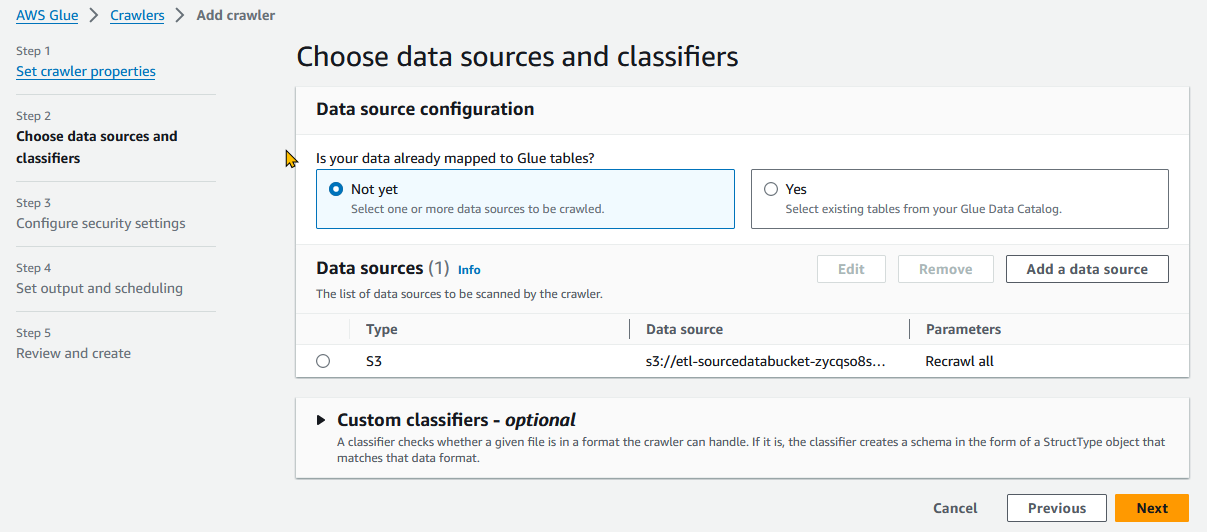
## Create Crawler for tables with data souce sales\_record.csv

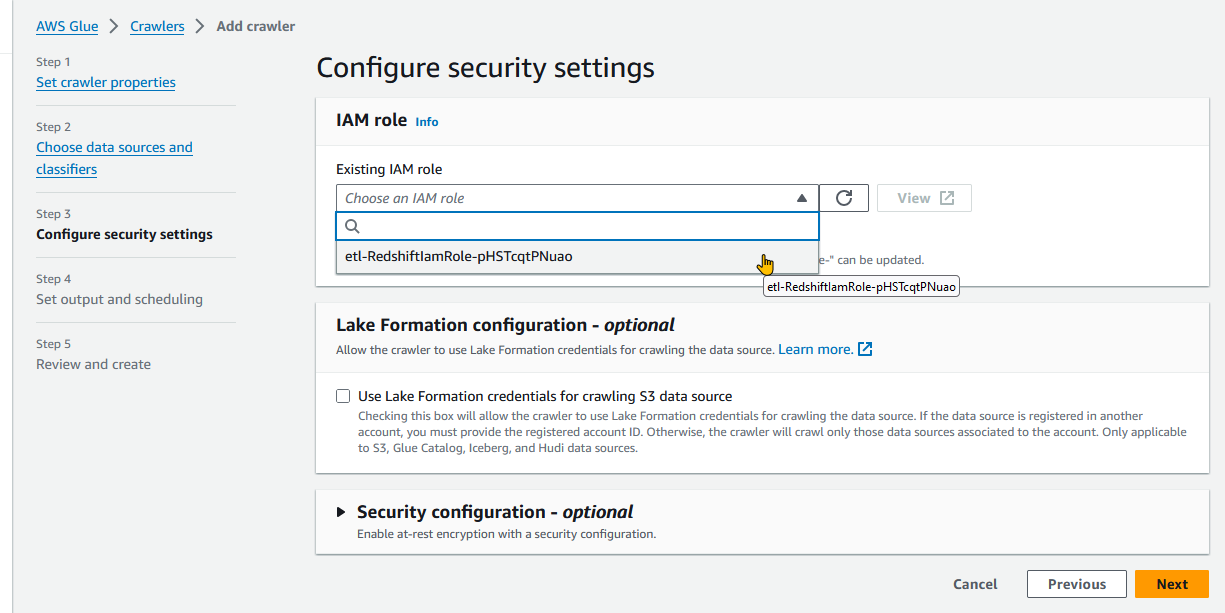


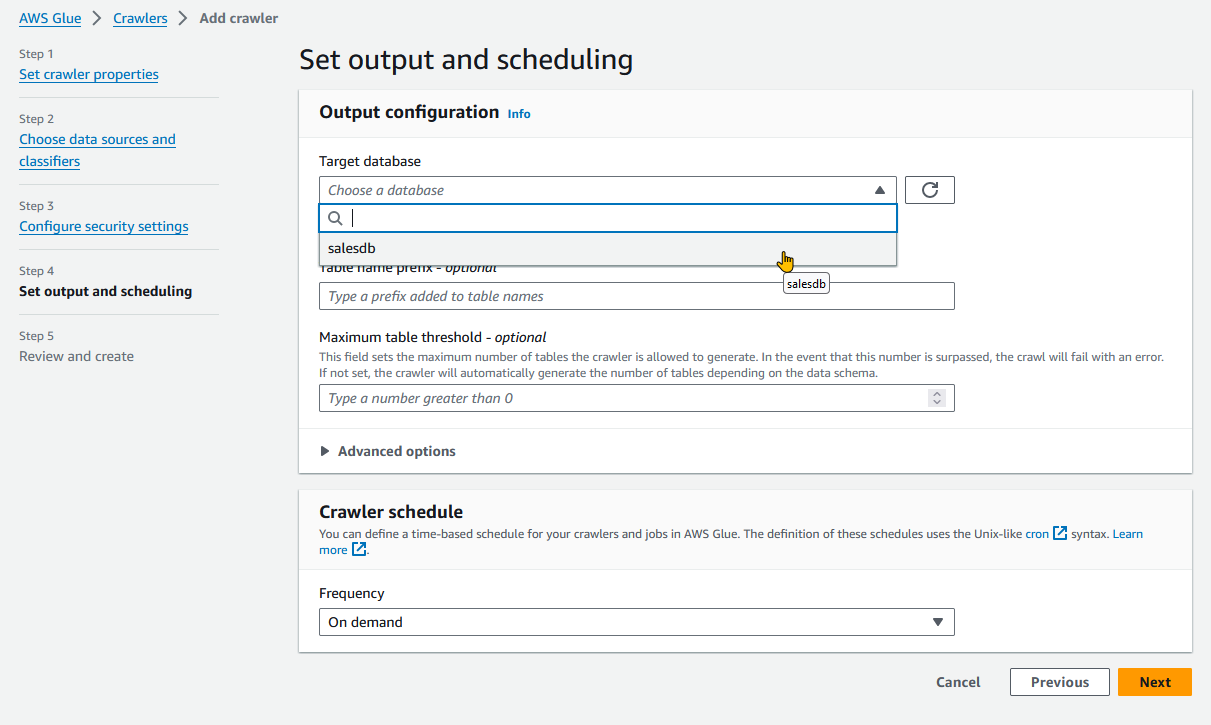


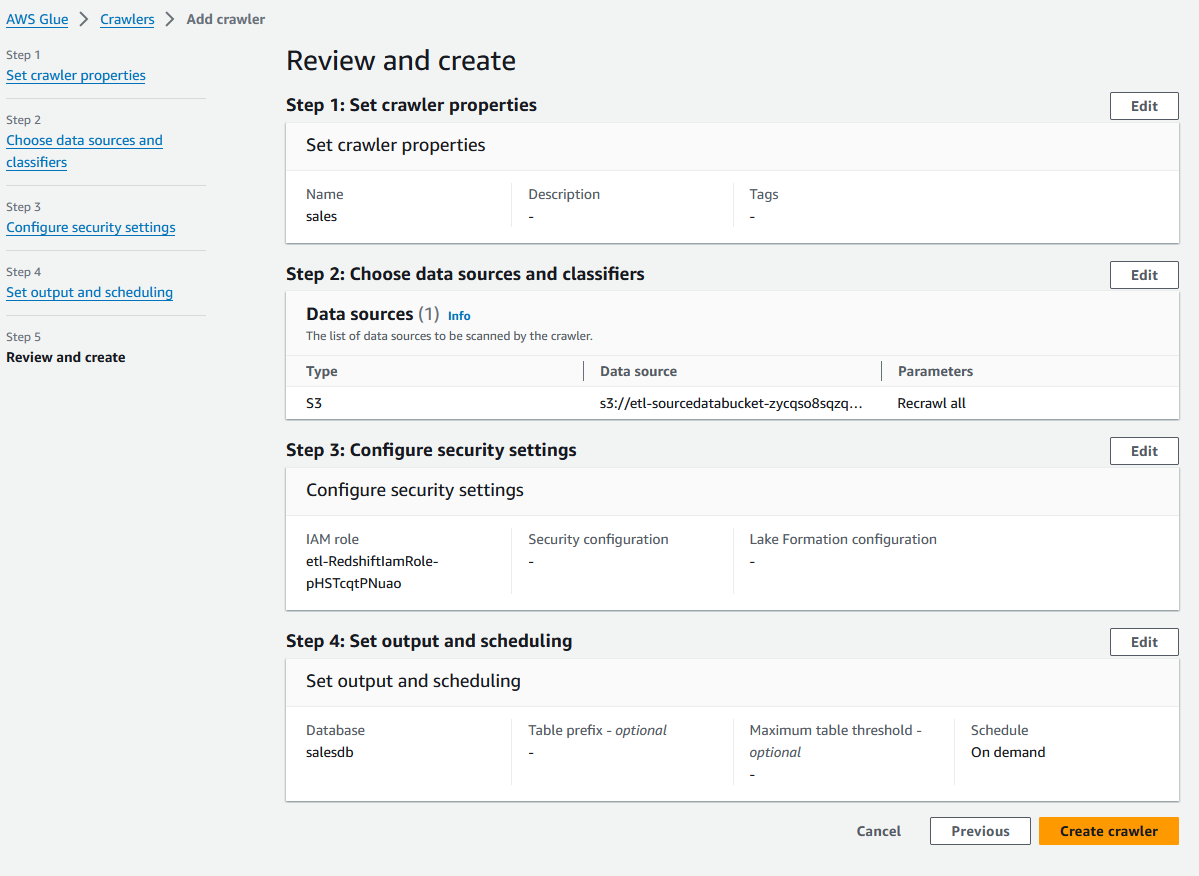


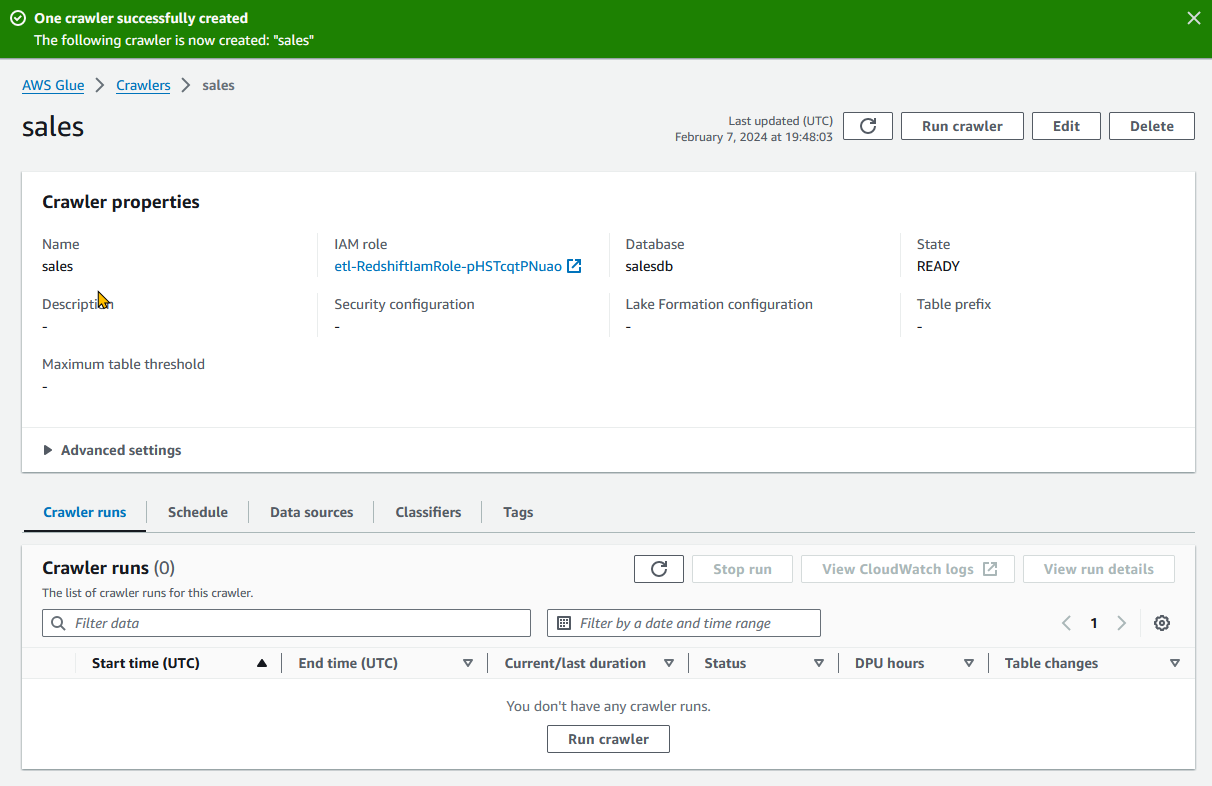


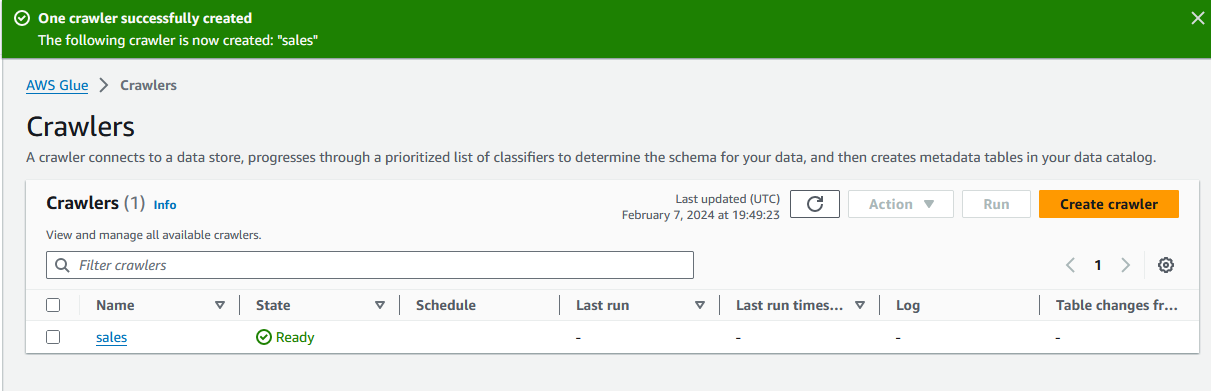




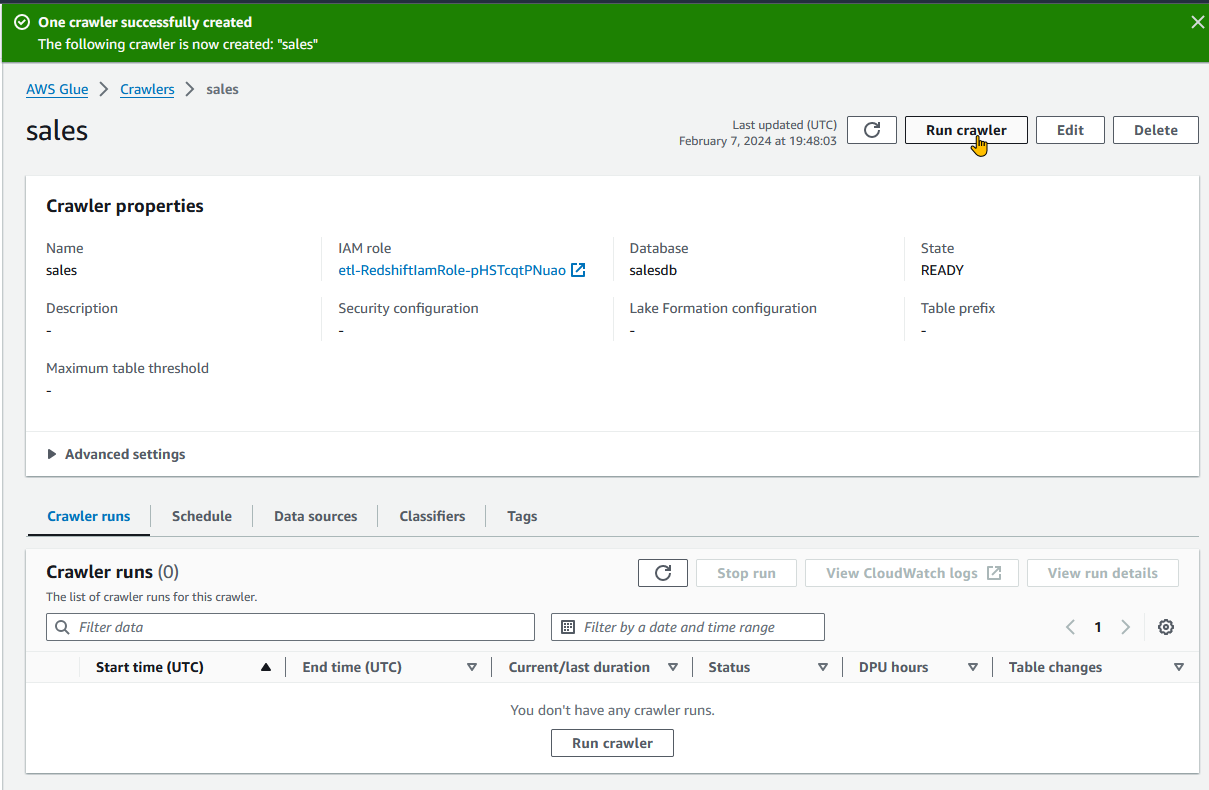


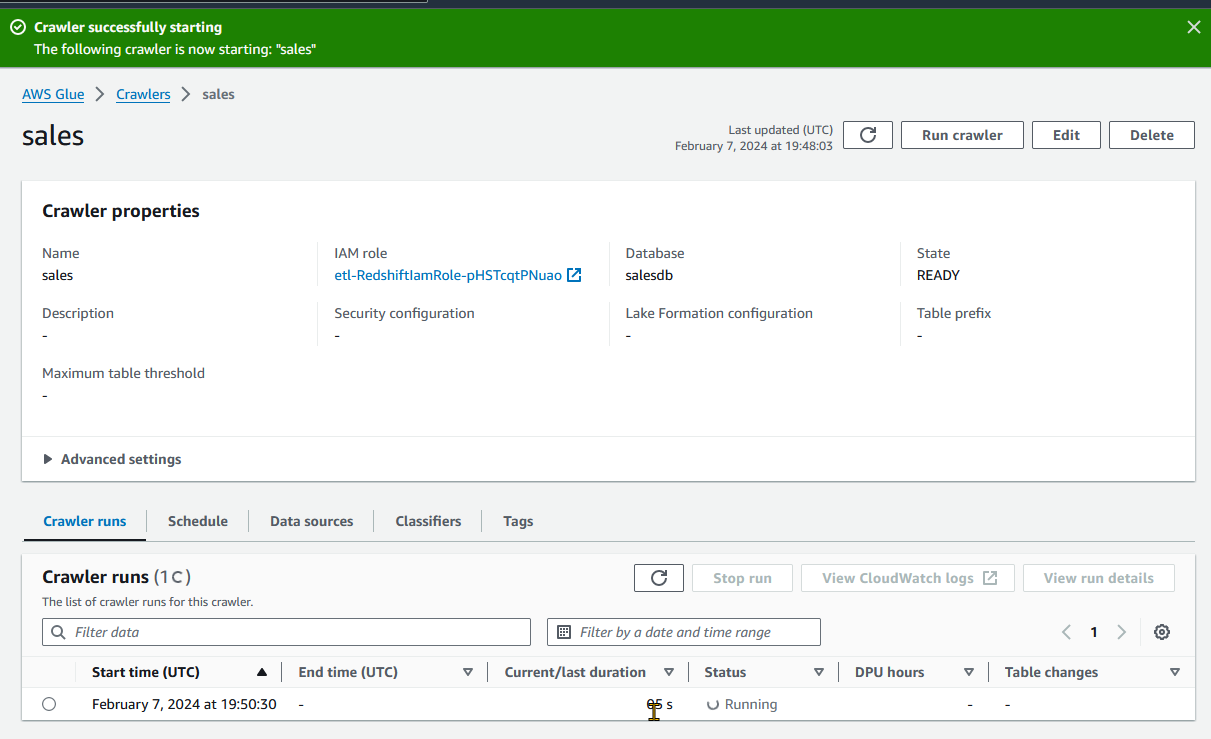


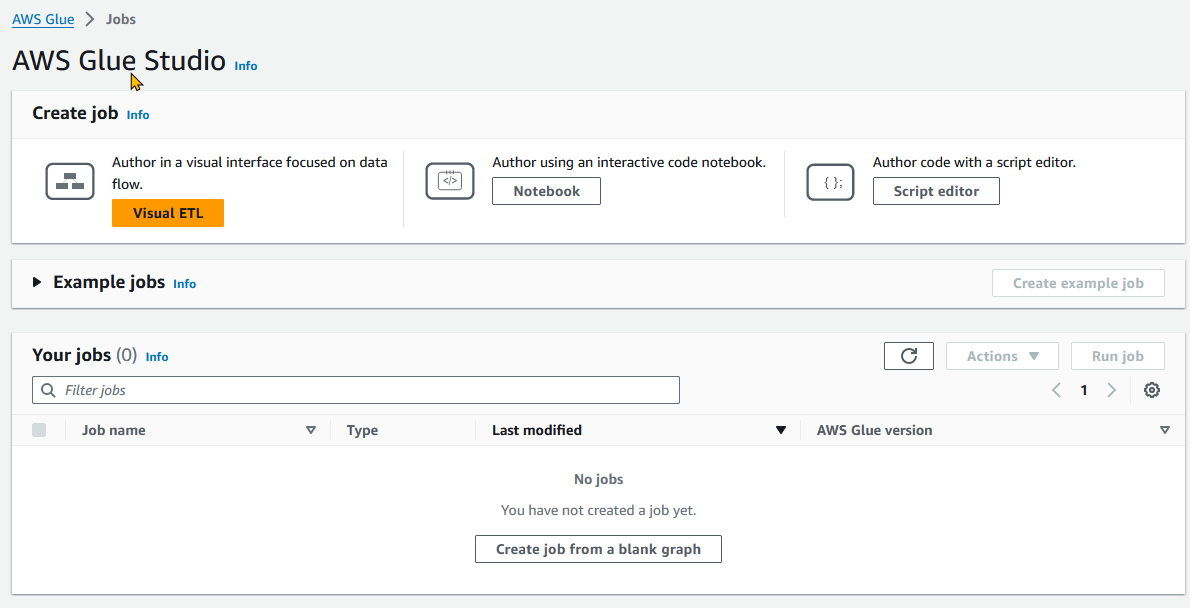




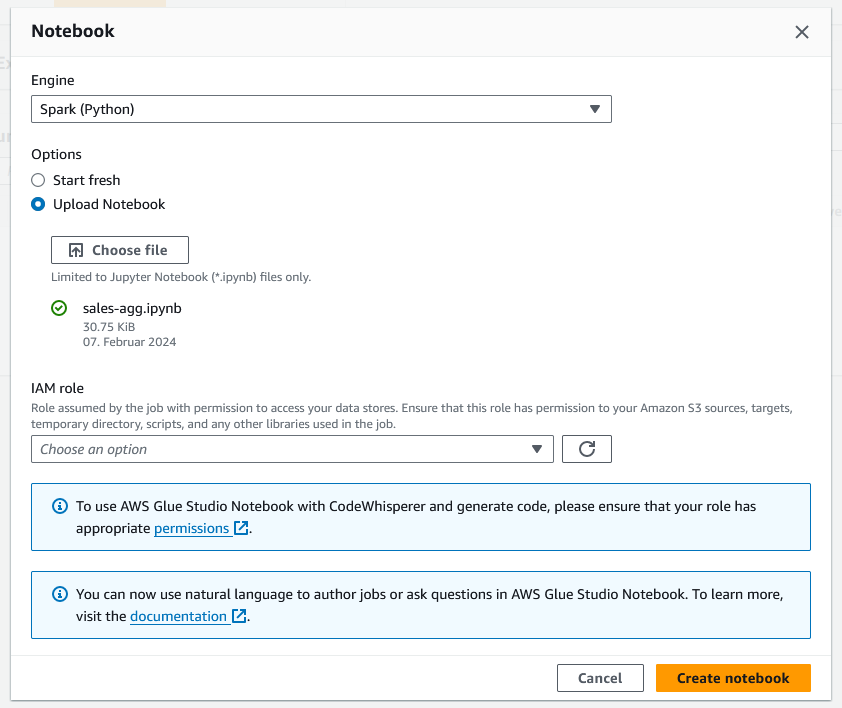
## Run the crawler

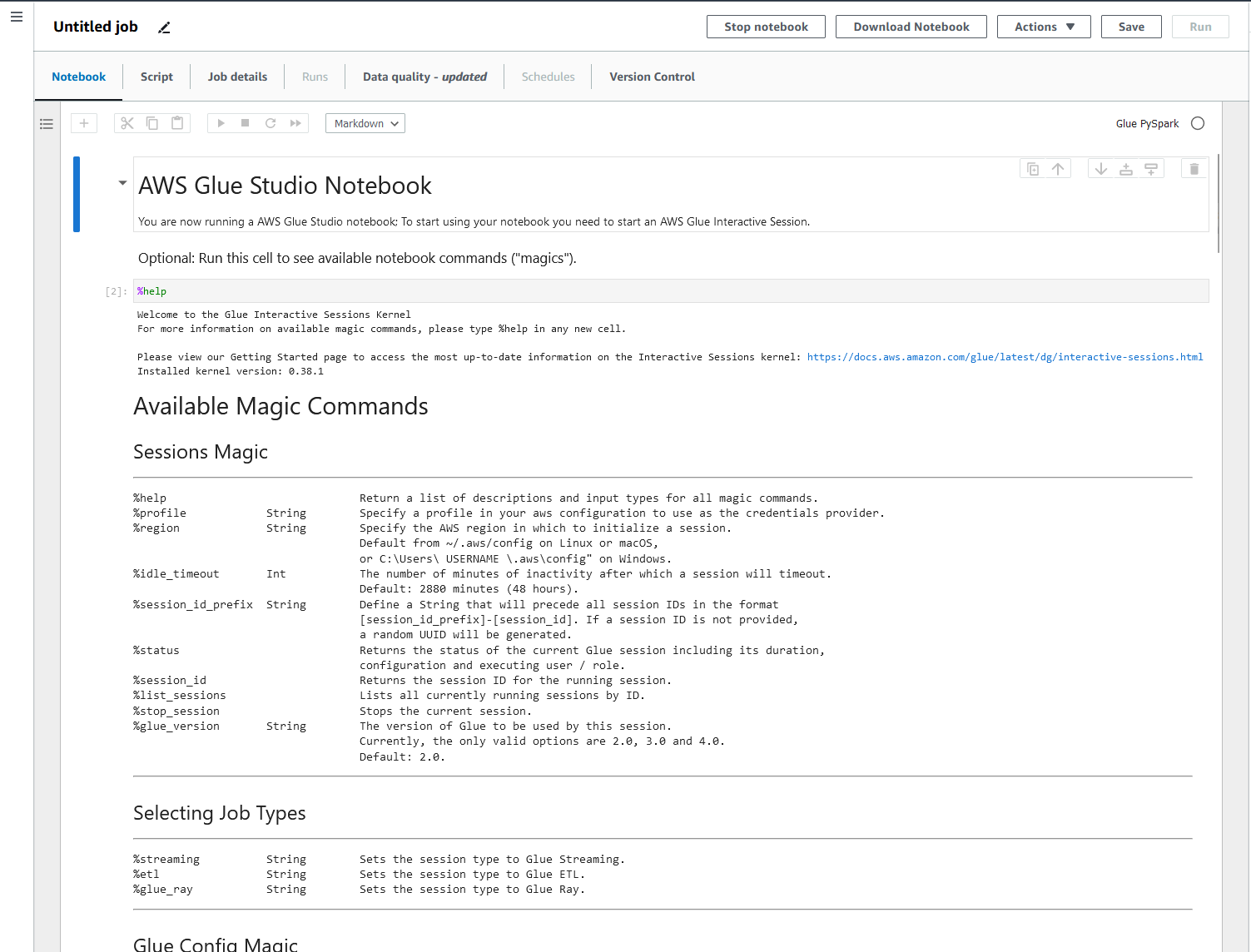




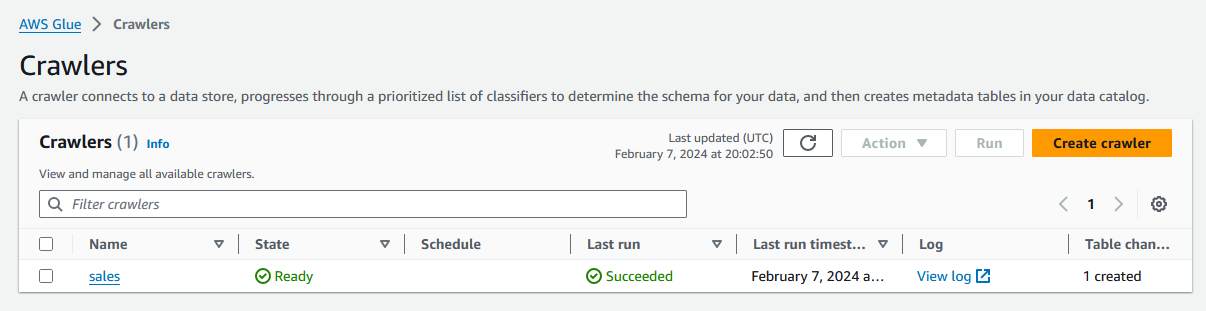


## ETL Job as interactive jupyter Notebook using pyspark





Crawler run succeeded

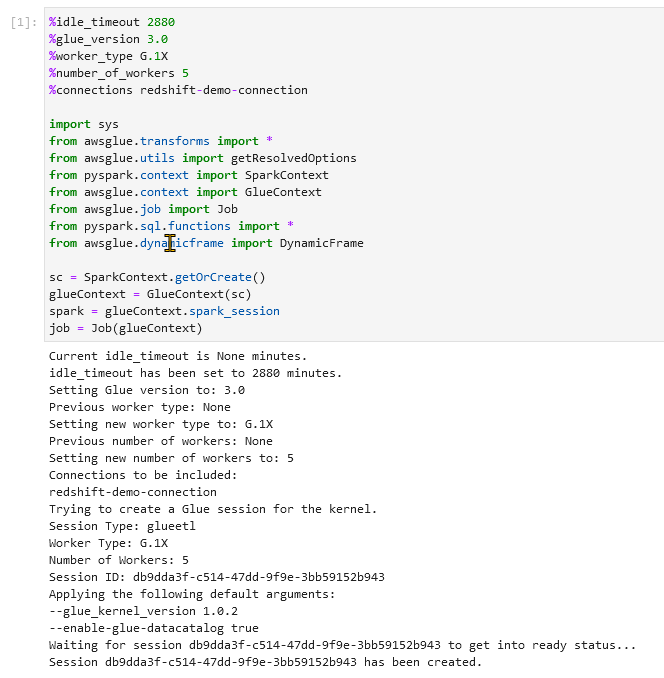


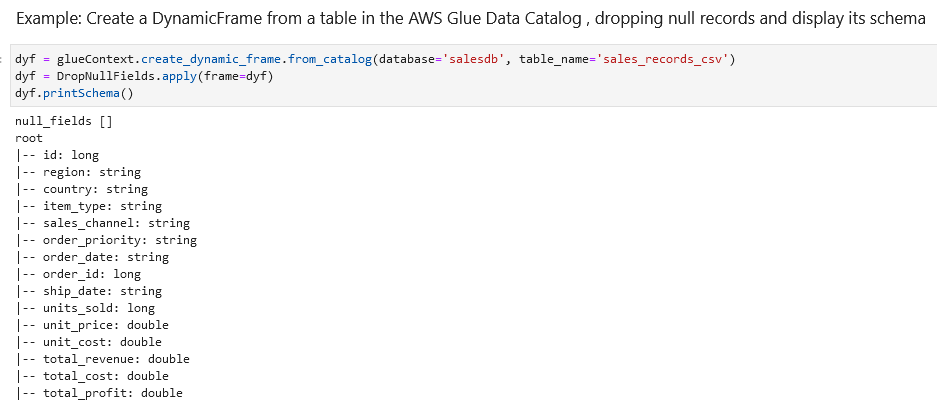
### Jupyter Notebook for ETL Process using Spark / Pyspark – Step by Step as interactive Notebook

**Workflow steps:**

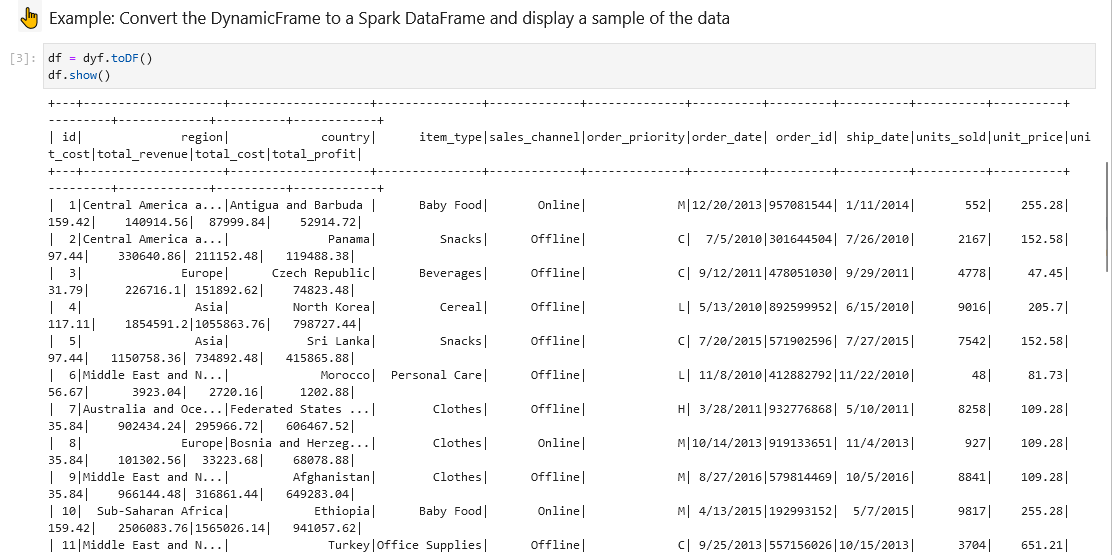
* fetch data from S3 in csv format,
* catalog data,
* clean (dropping null values)
* transform (date)
* perform some aggregation
* write data to redshift db

Import Libaries & initializing Spark and glue context (main entry point for AWS glue ETL)

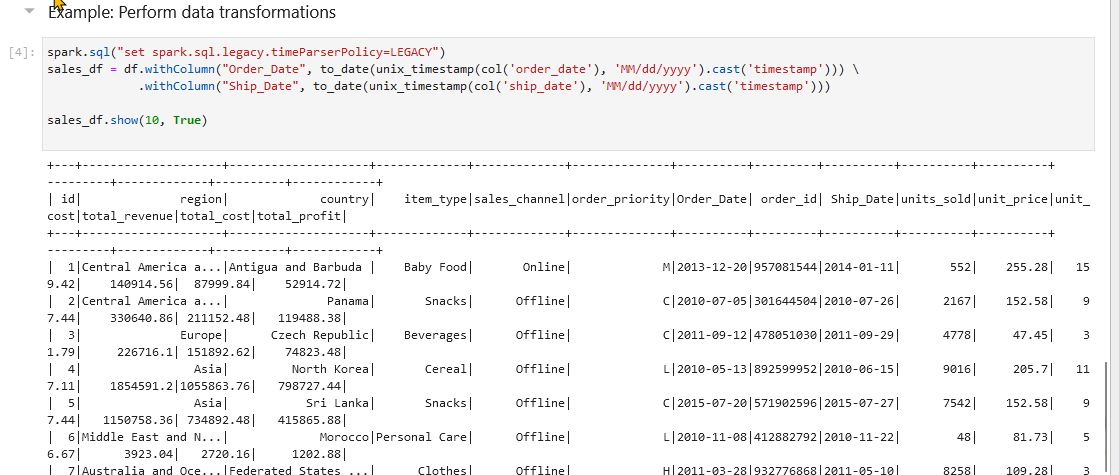




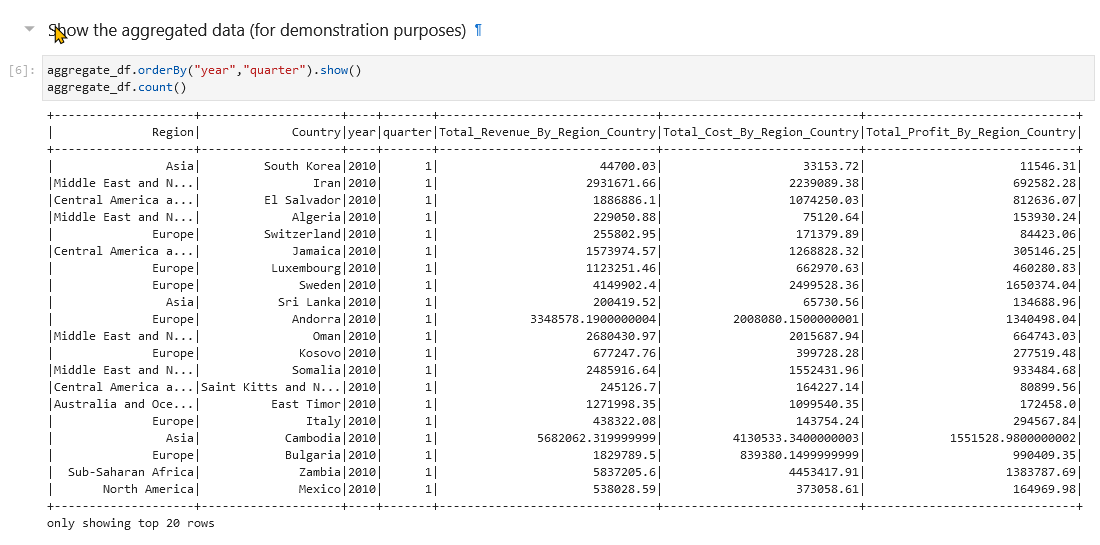
Dynamic DataFrame similar to spark dataframe but different syntax to perfom action

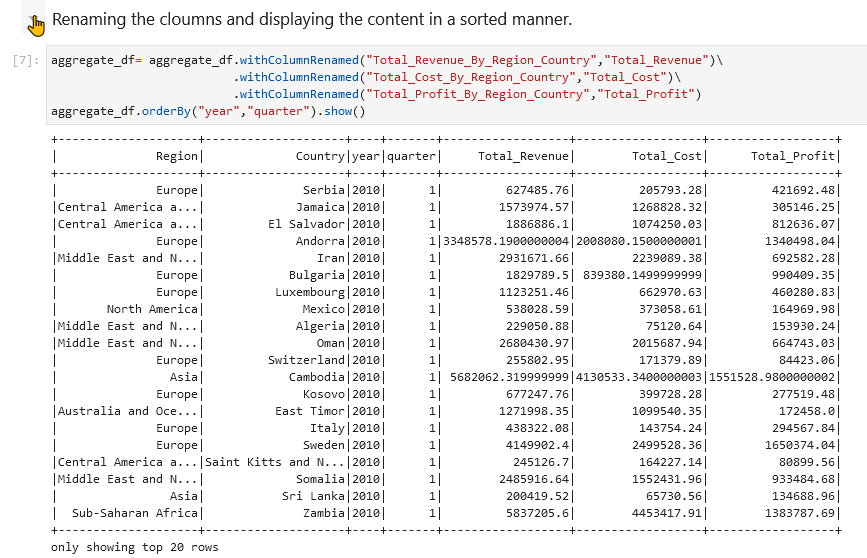


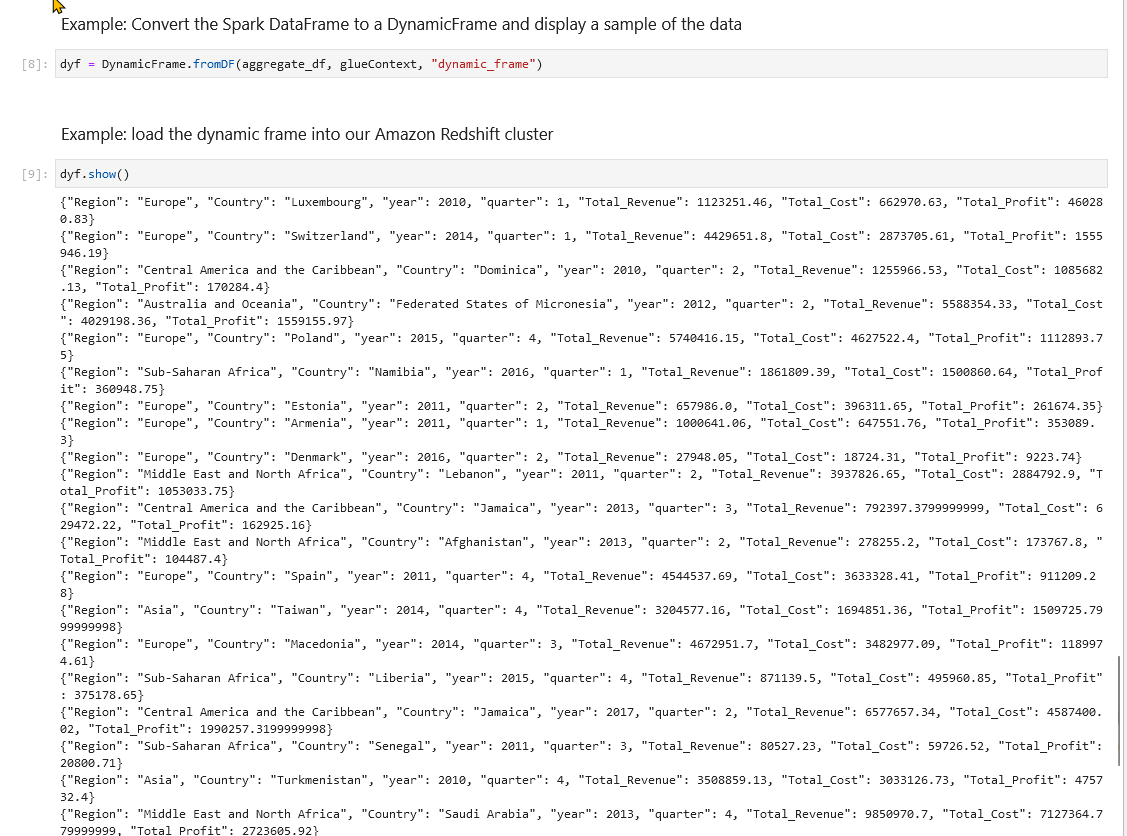
Date Transformation





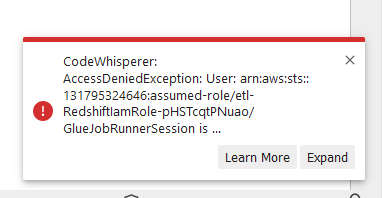


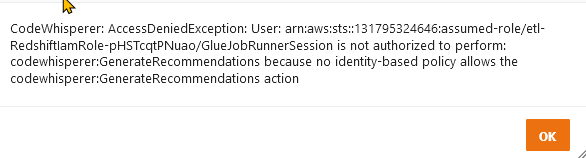




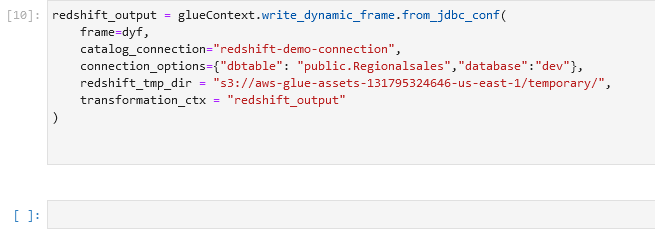
Writing dynamicFrame to Redshift   
Authorization access problem





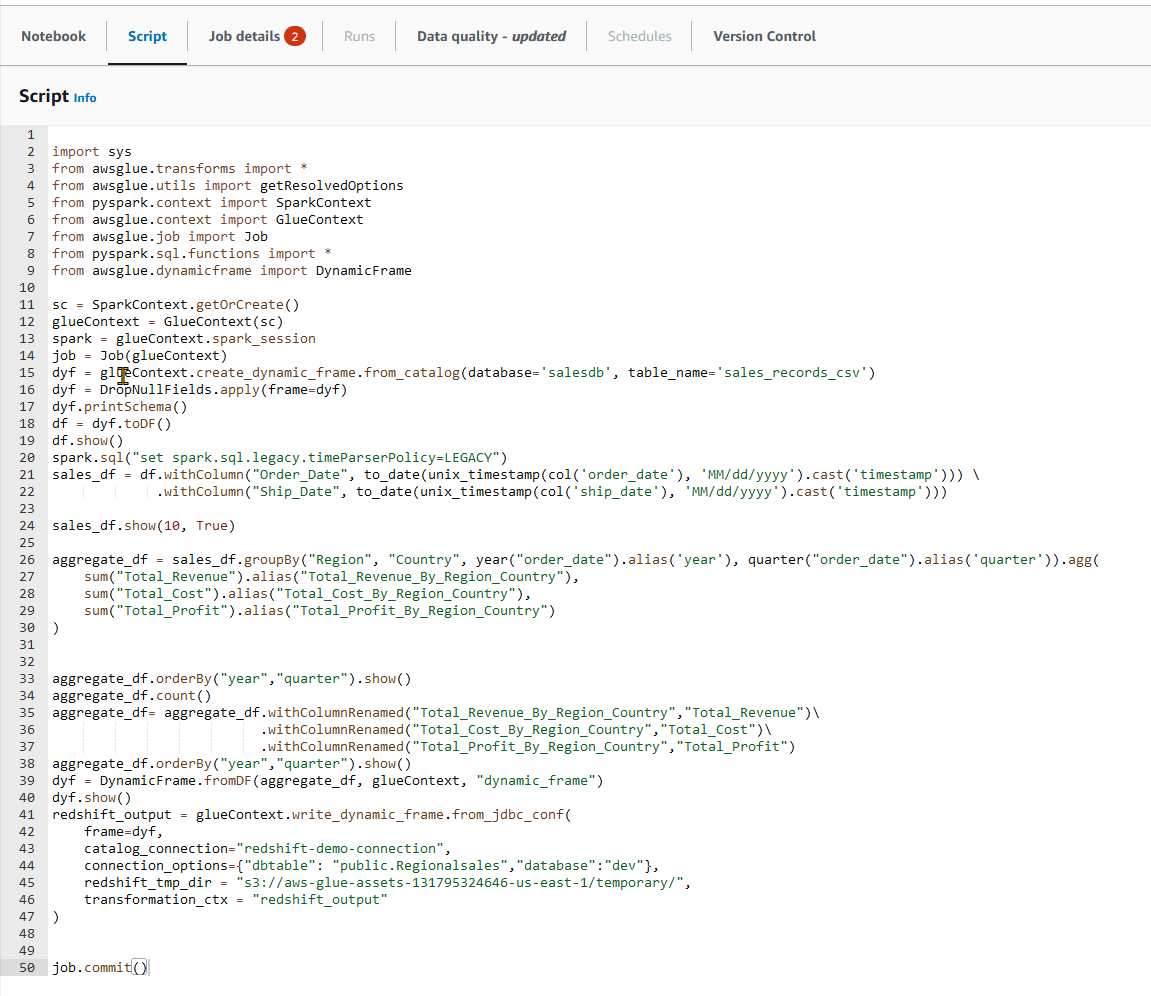


**Solution**:  
First a lot of trial & error concerning role authorizations…but the solution was much simpler…  
The adress for the redshift temp directory was wrong that caused the autohrization problem.



### Jupyter Notebook as Pyspark Script that could be sheduled

### 



# AWS Redshift

