

| Business Template  **IAM and S3 services** |
| --- |
|  |

[**1. VPN Setup**](#_heading=h.5393pv2k275r) **3**

[**2. AWS CLI Setup**](#_heading=h.3tnjr8a3z9x0) **4**

[**3. S3 Bucket Preparation**](#_heading=h.rsihriyidwv5) **5**

[**4. Folder Hierarchy**](#_heading=h.vcnqr1lhvz2u) **7**

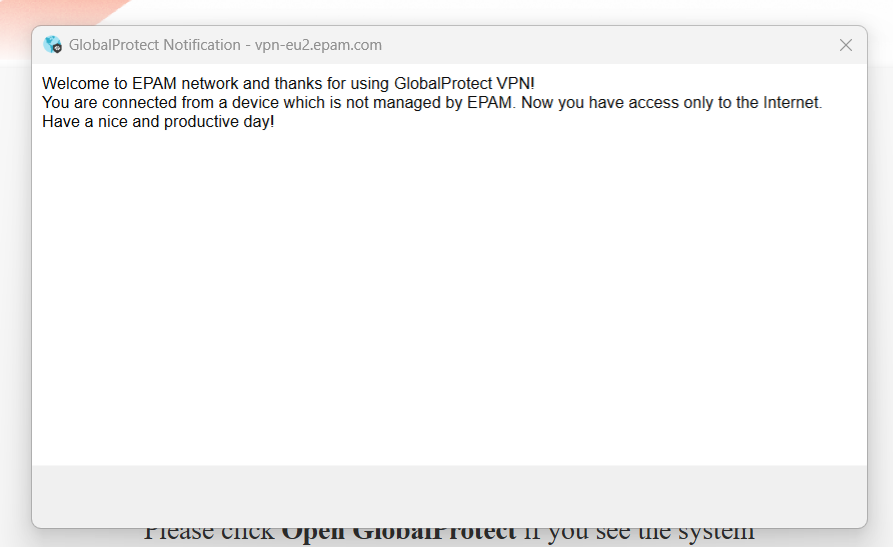
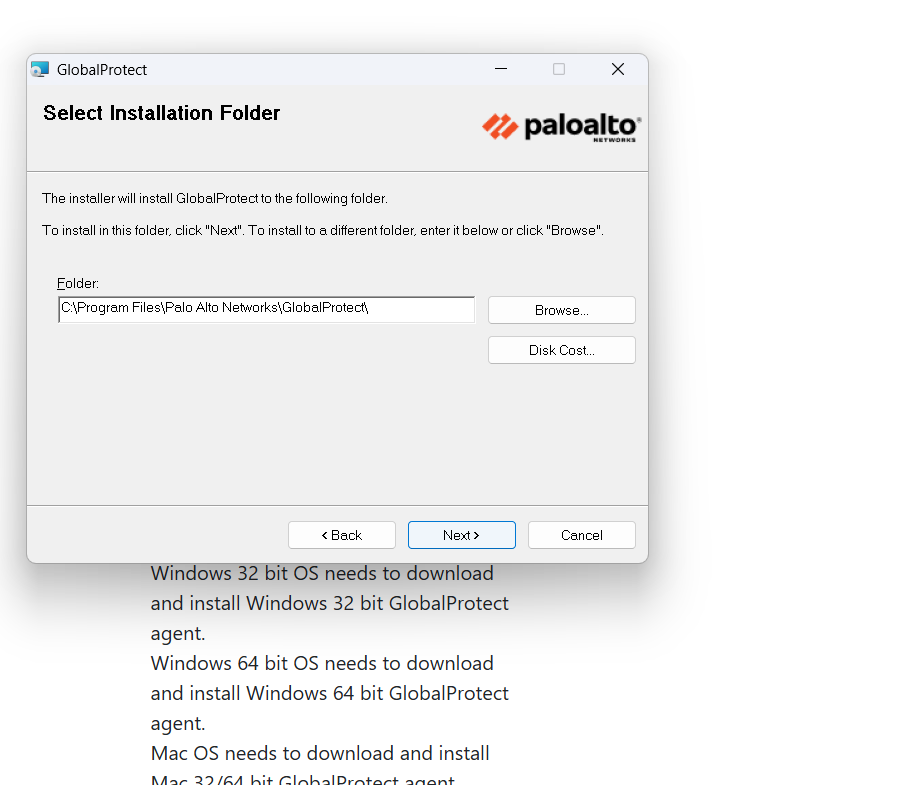
[**5. Data Catalog Setup with AWS Glue**](#_heading=h.1wchkmdehnck) **8**

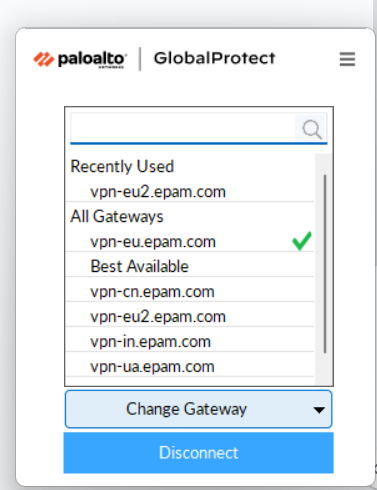
[**6. Data Type Validation and Schema Alignment**](#_heading=h.mkewoeah6xr7) **10**

[**7. Query Execution and Validation**](#_heading=h.qs45trofau8e) **18**

# 1. VPN Setup

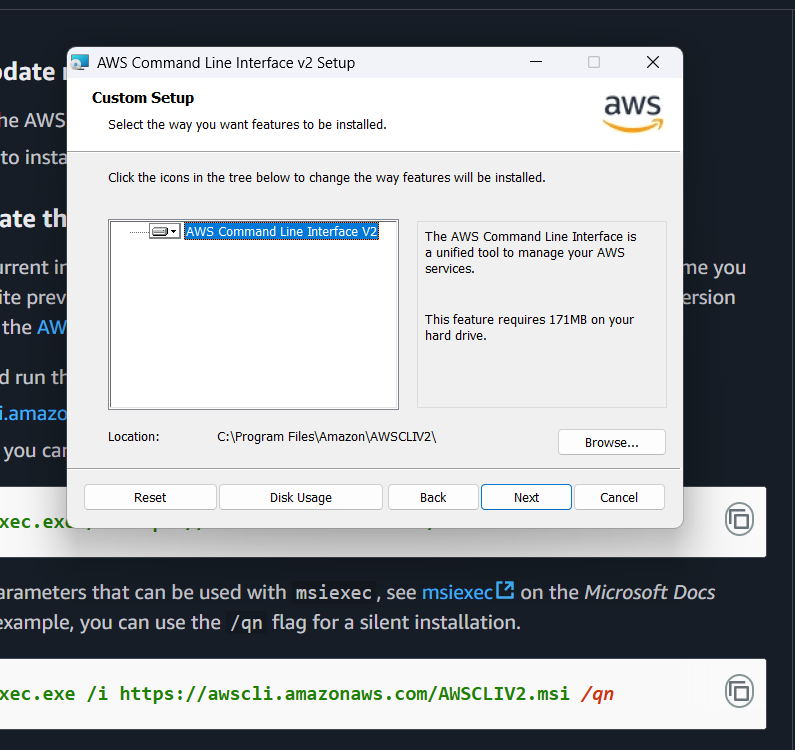
First, I installed the VPN and configured it to connect to vpn-eu.epam.com.



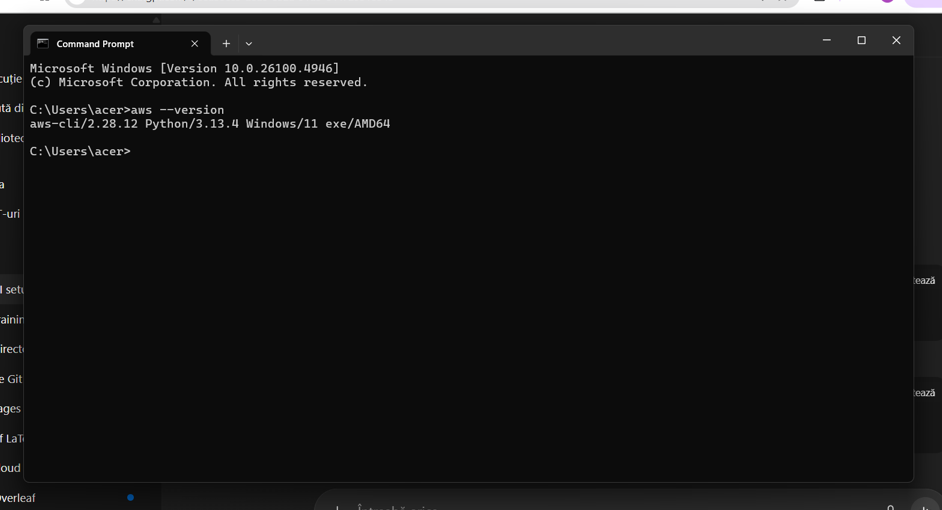


# 2. AWS CLI Setup

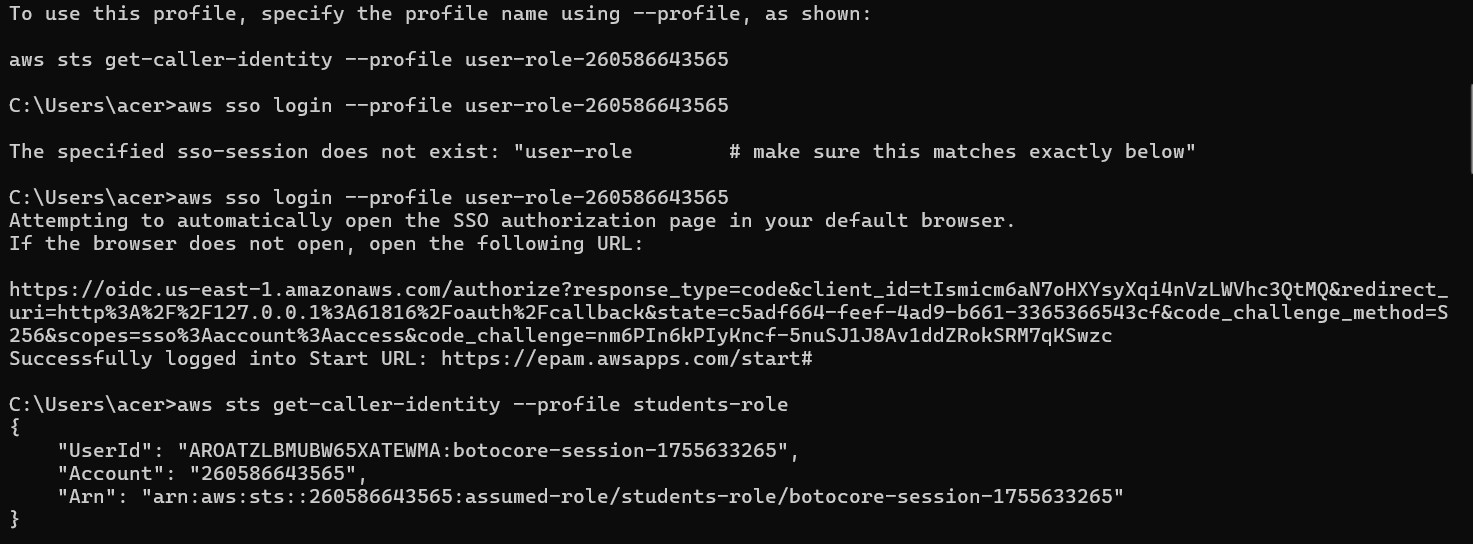
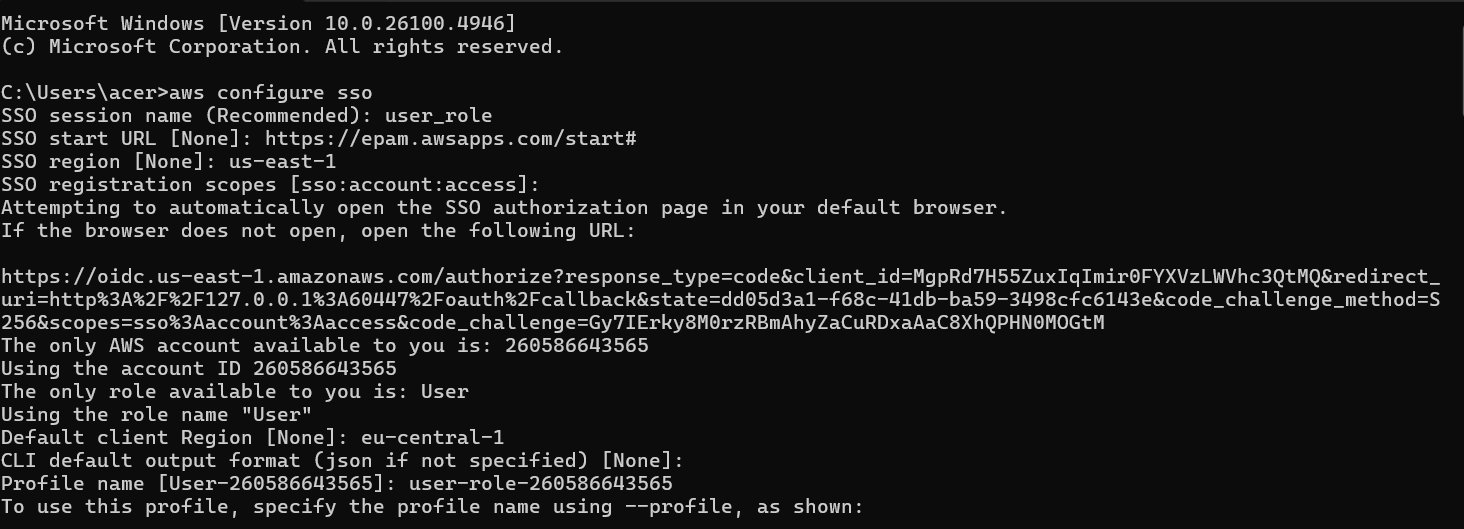
Then, I installed and configured the AWS Command Line Interface (CLI) on my laptop.



Next, I checked the installed AWS CLI version.

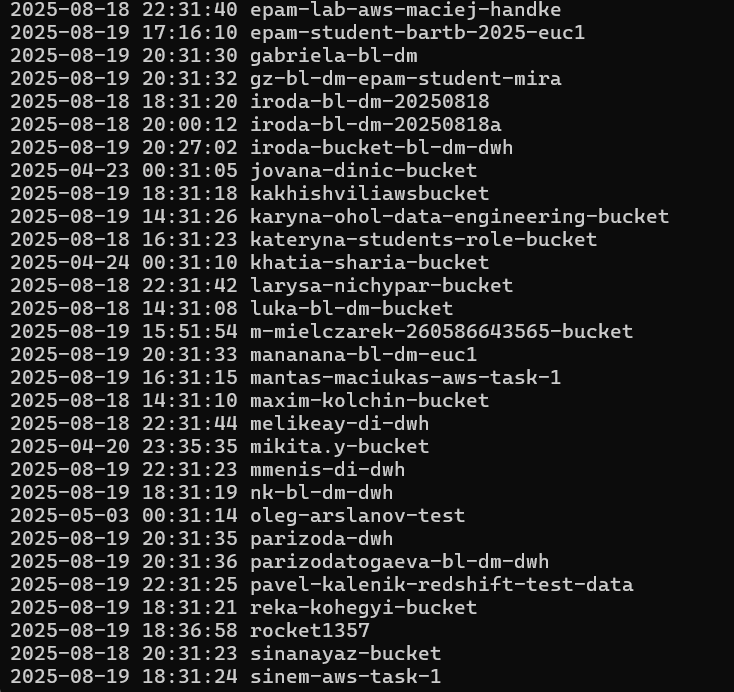
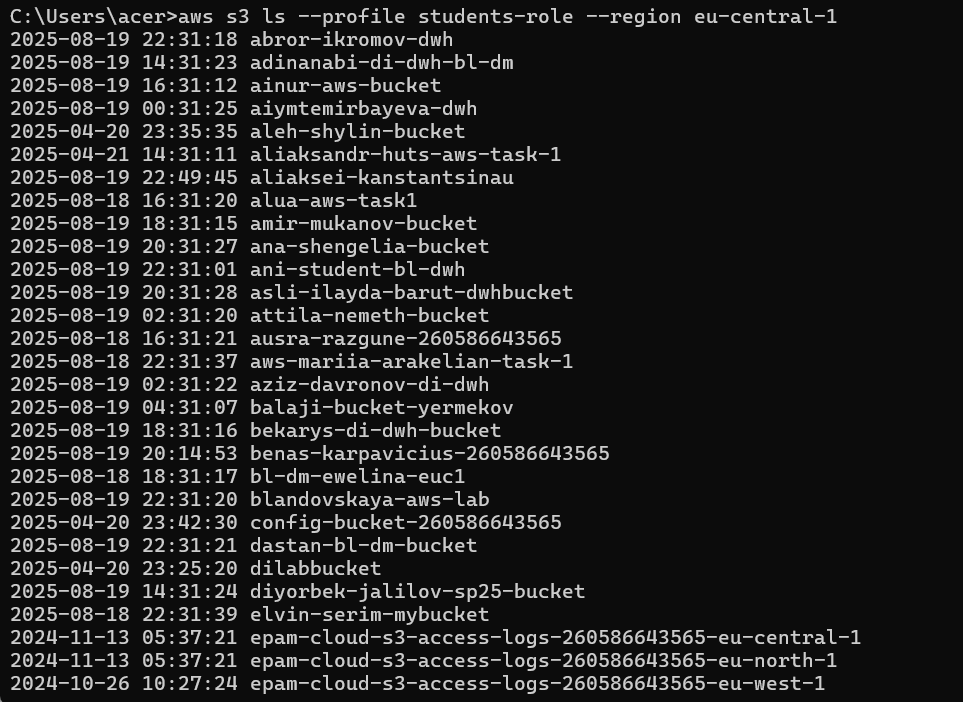


I set up the AWS account using the command prompt.

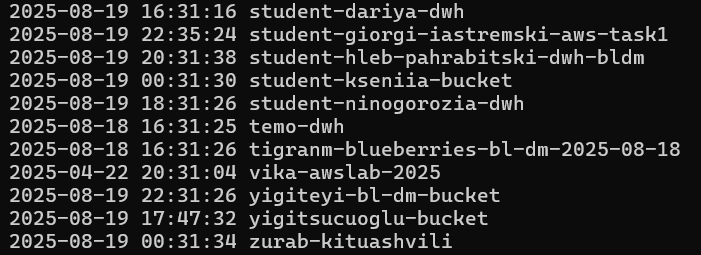


# 3. S3 Bucket Preparation

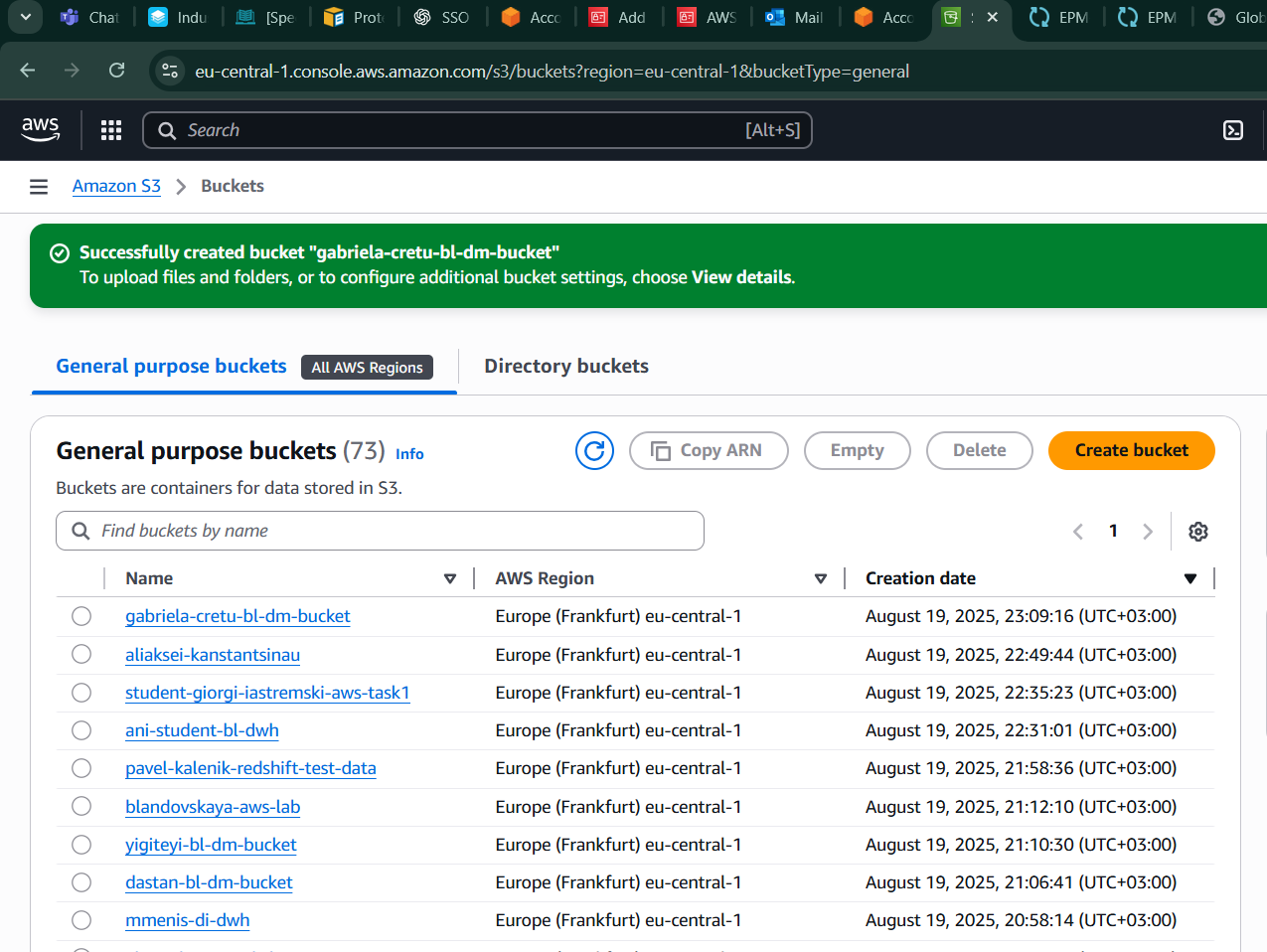
Next, I listed all the existing S3 buckets.



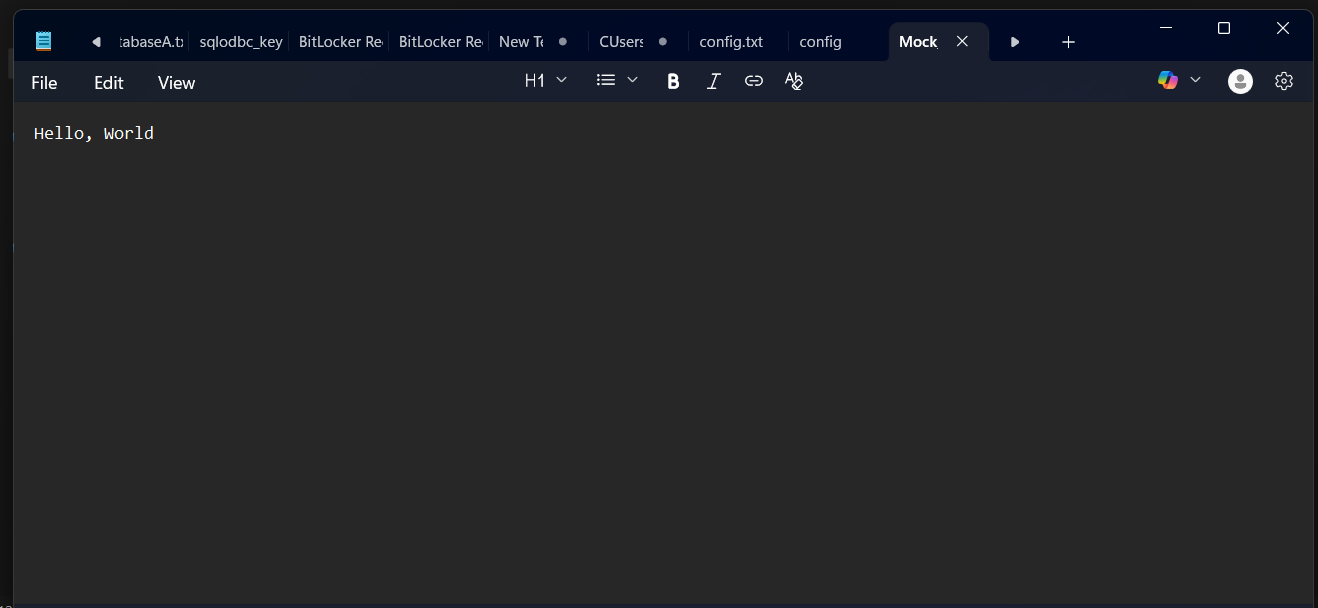
\



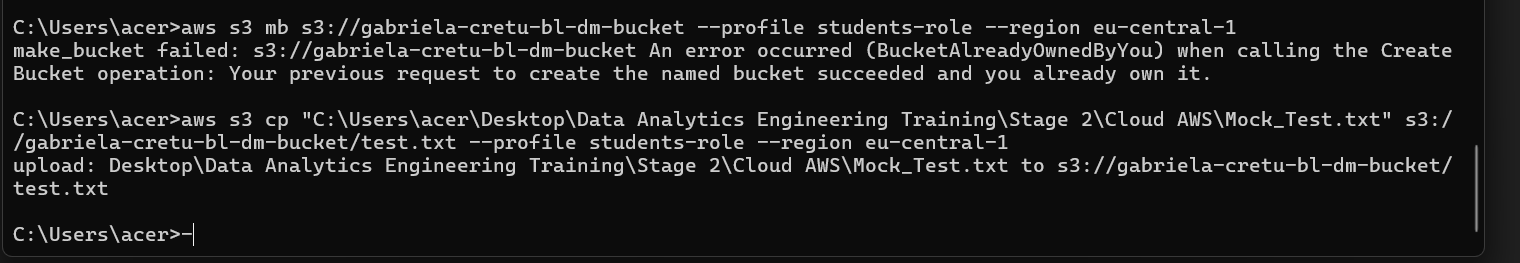
In the next step, I created a new S3 bucket.

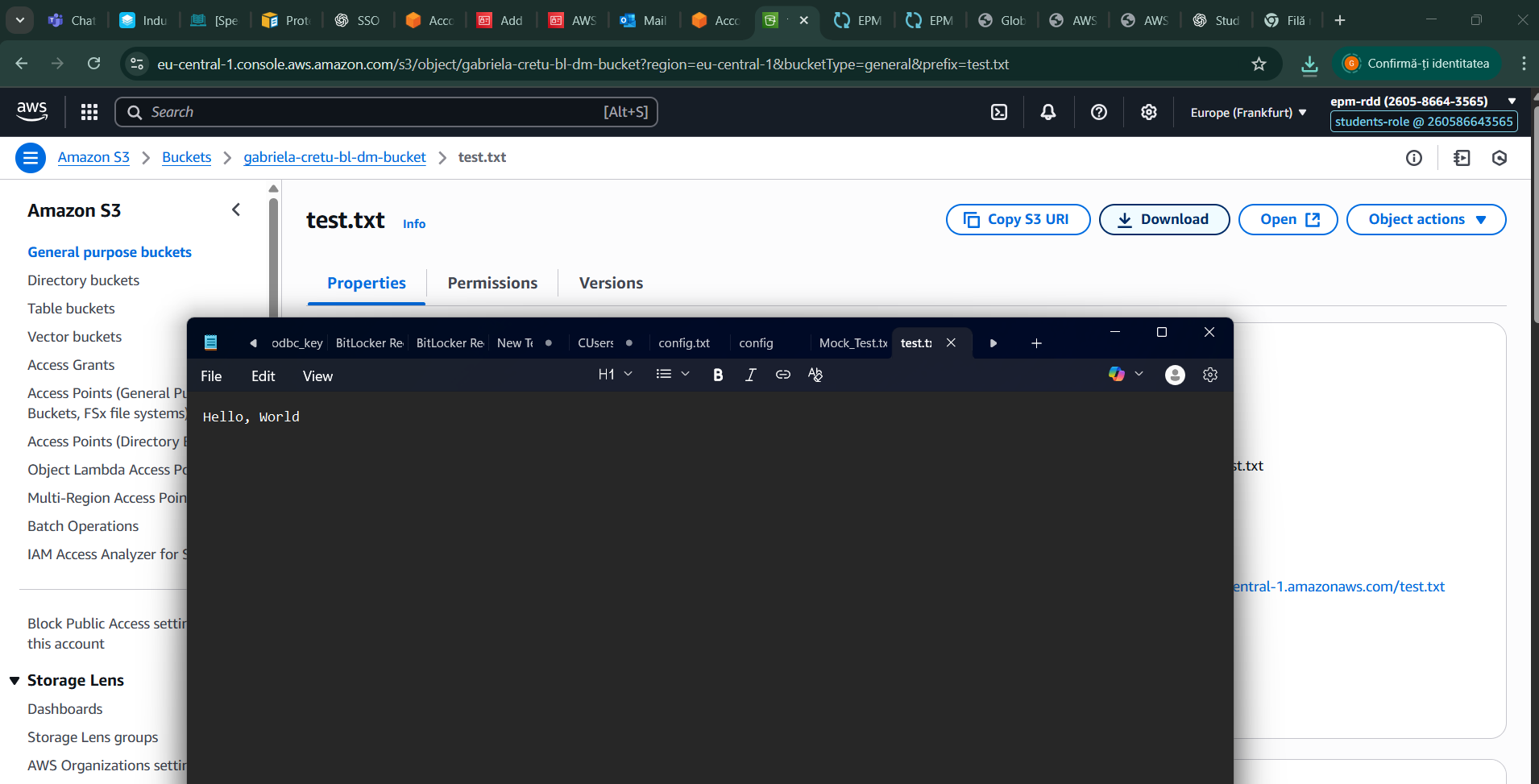
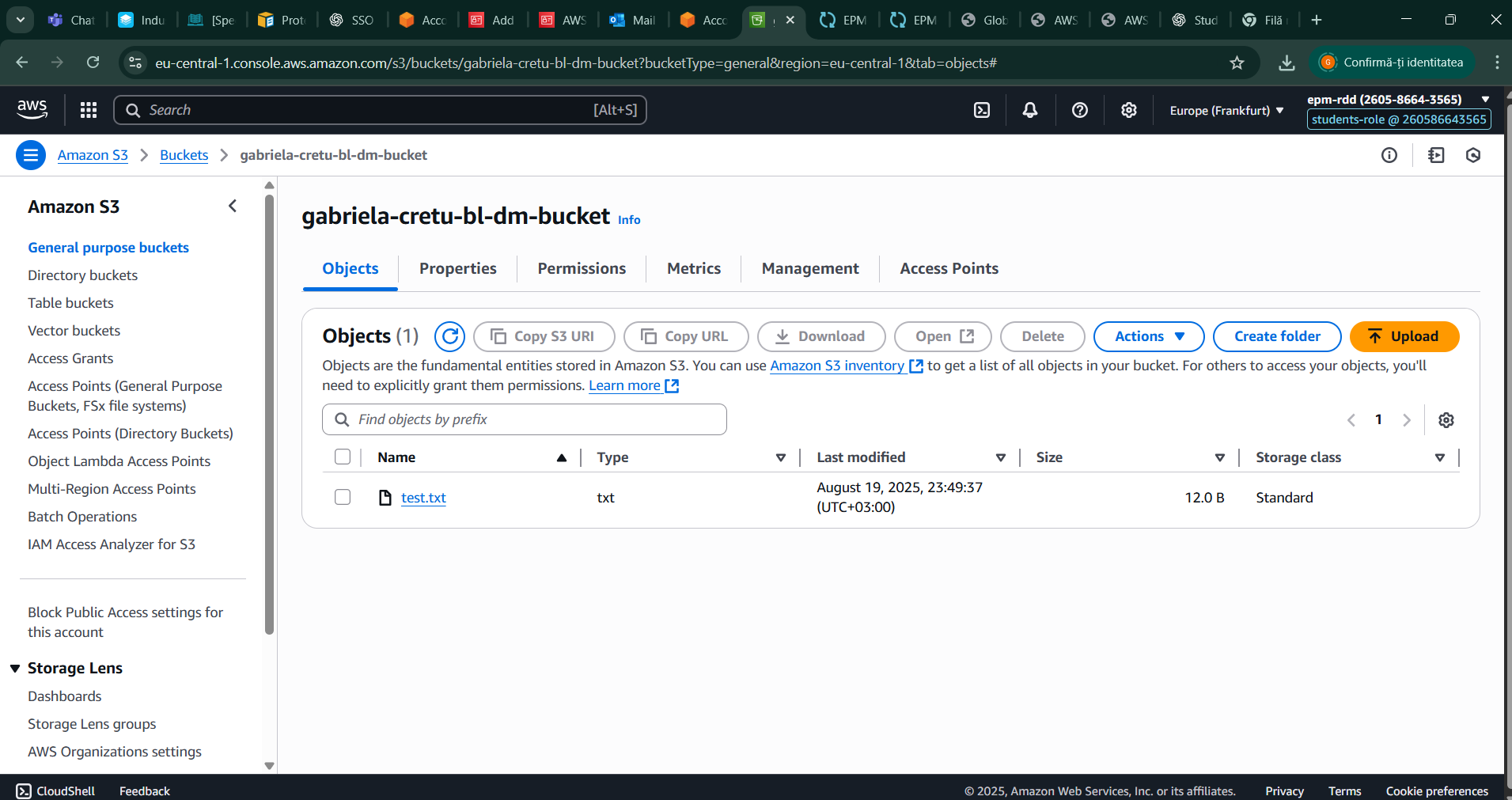


Next, I created a mock test file to verify that I could upload it to my S3 bucket.



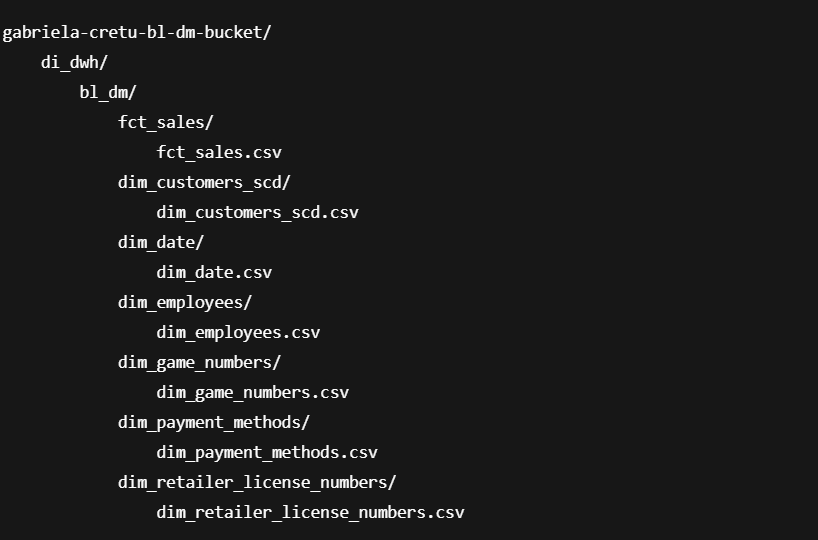
As we can see, the upload was successful, confirming that our connection is working properly.



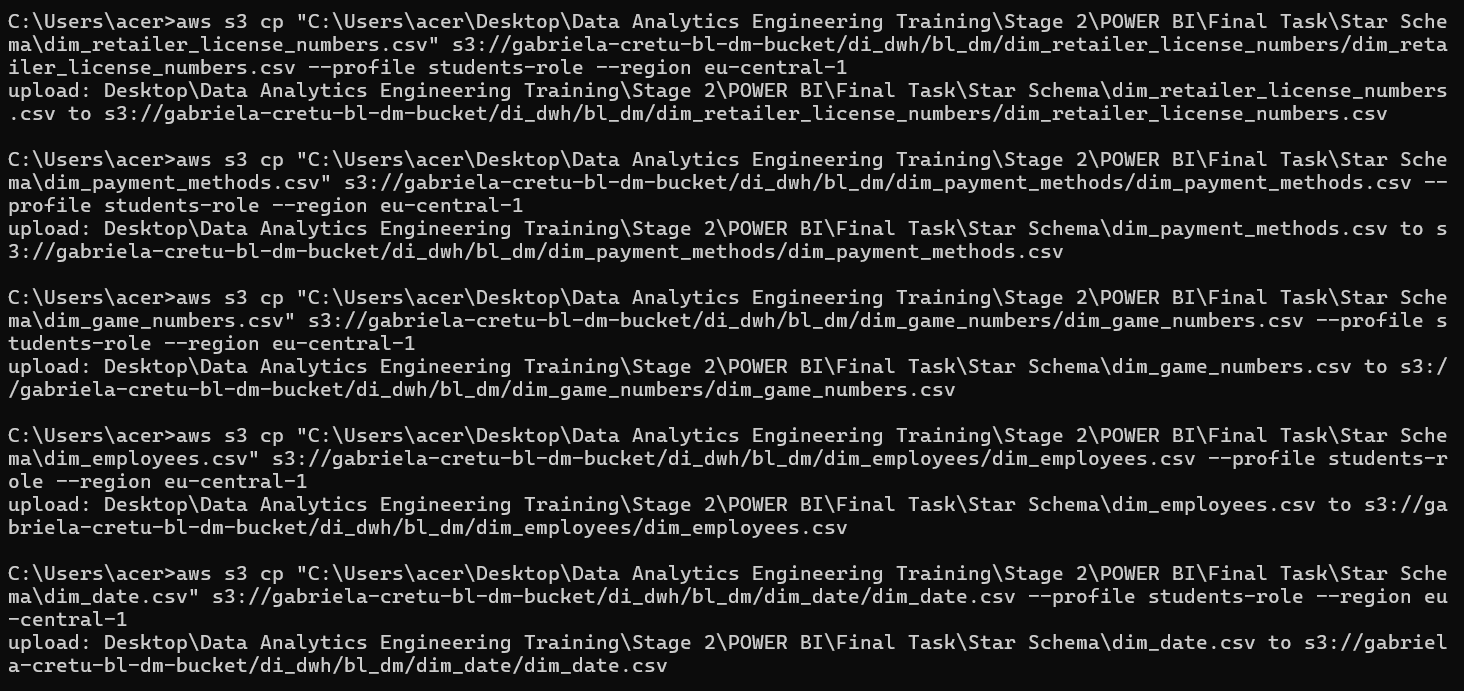


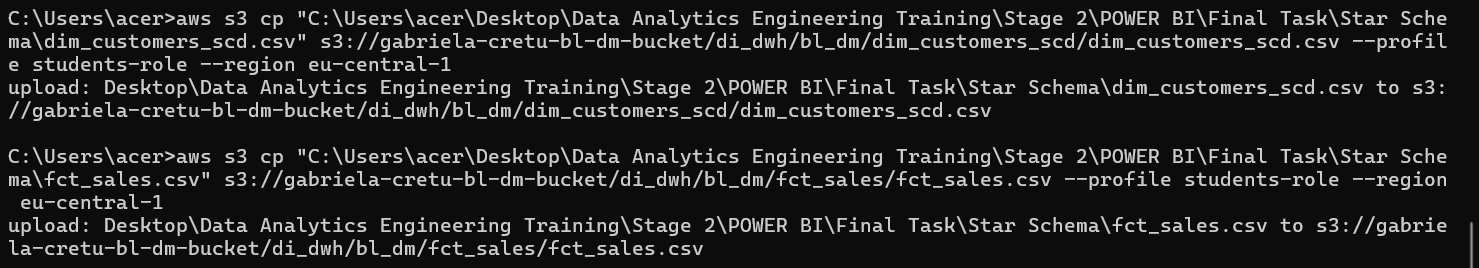
# 4. Folder Hierarchy

Next, I configured a folder hierarchy to correctly define my data warehouse in the cloud, following established conventions to ensure it functions properly.



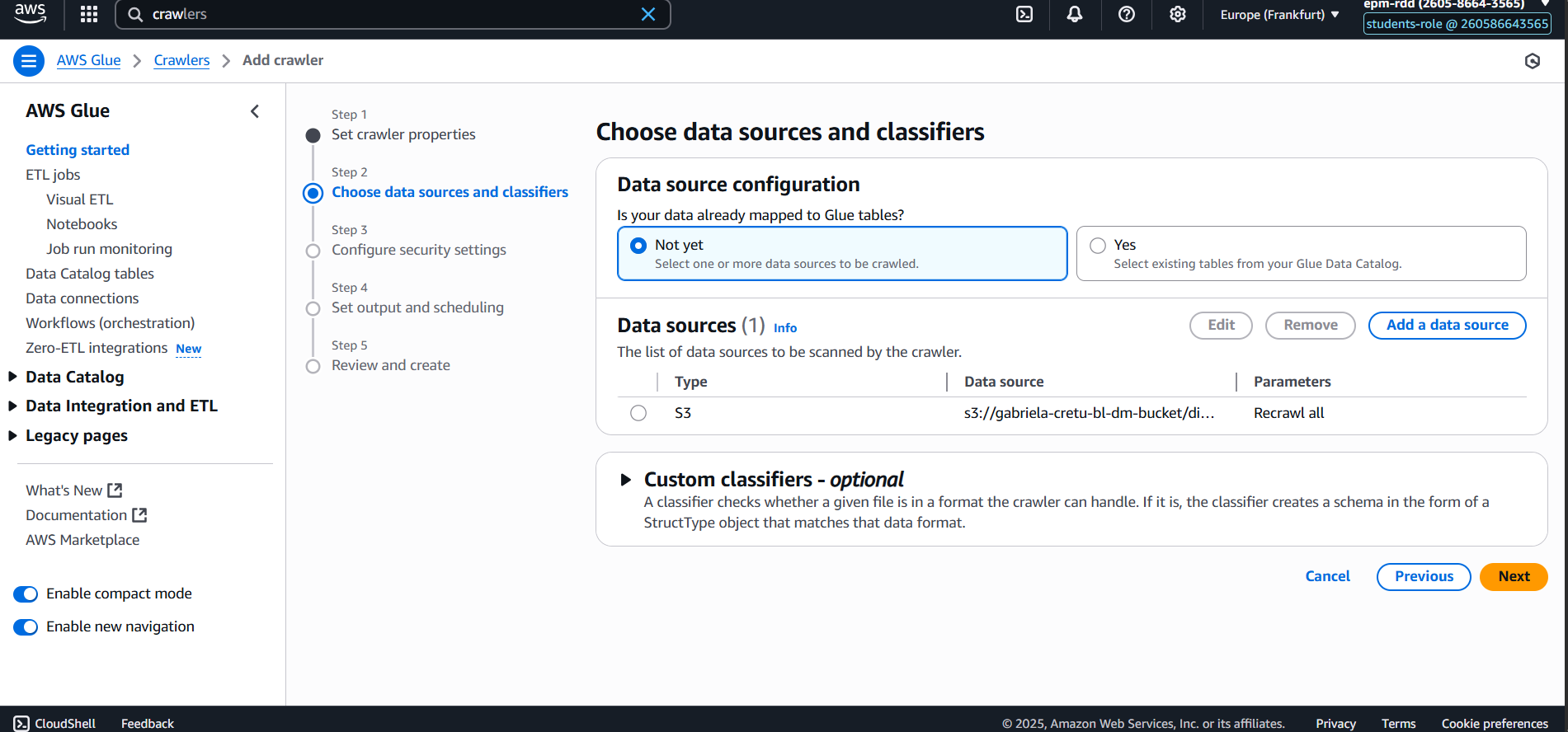
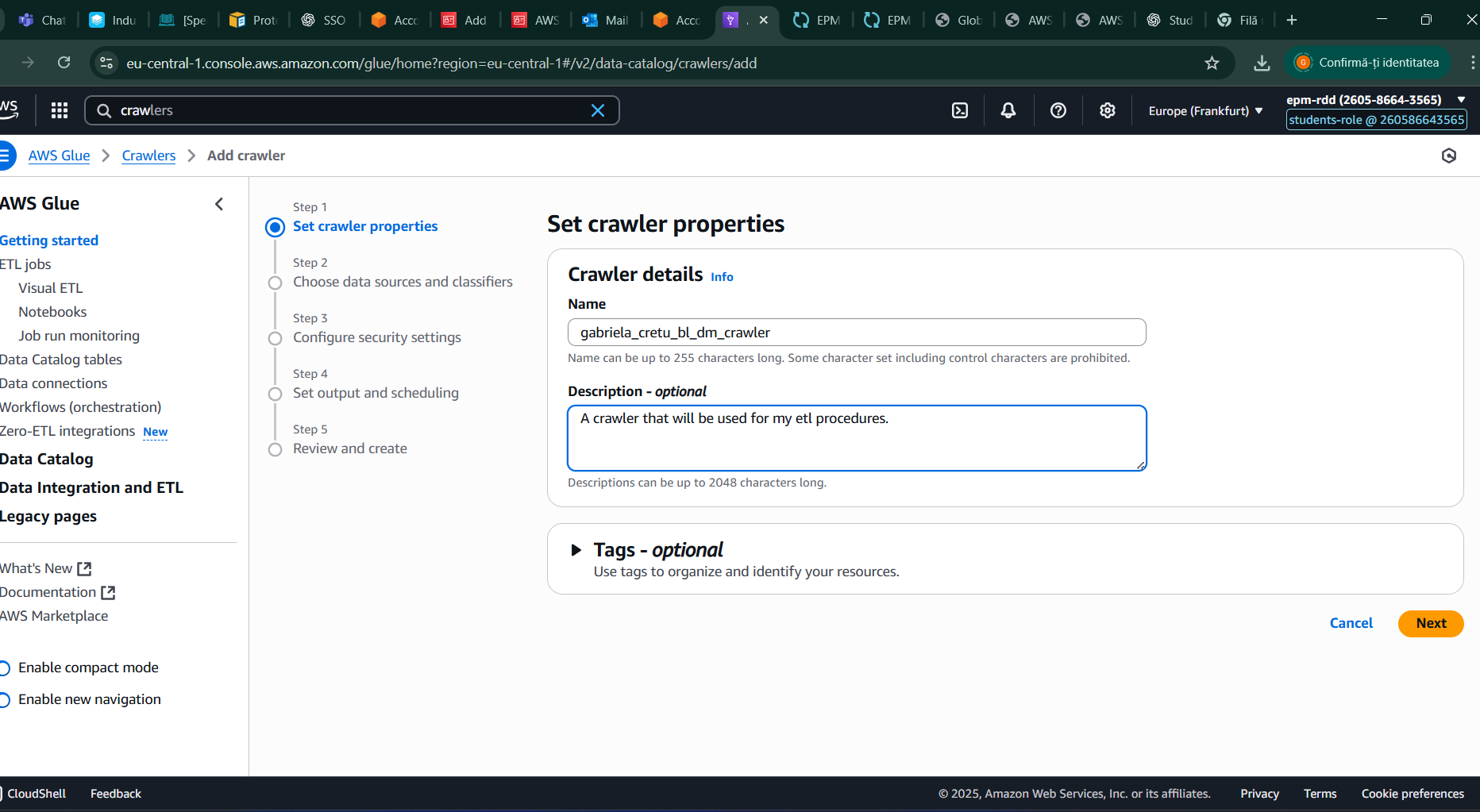
Next, based on the defined hierarchy, I uploaded my tables.

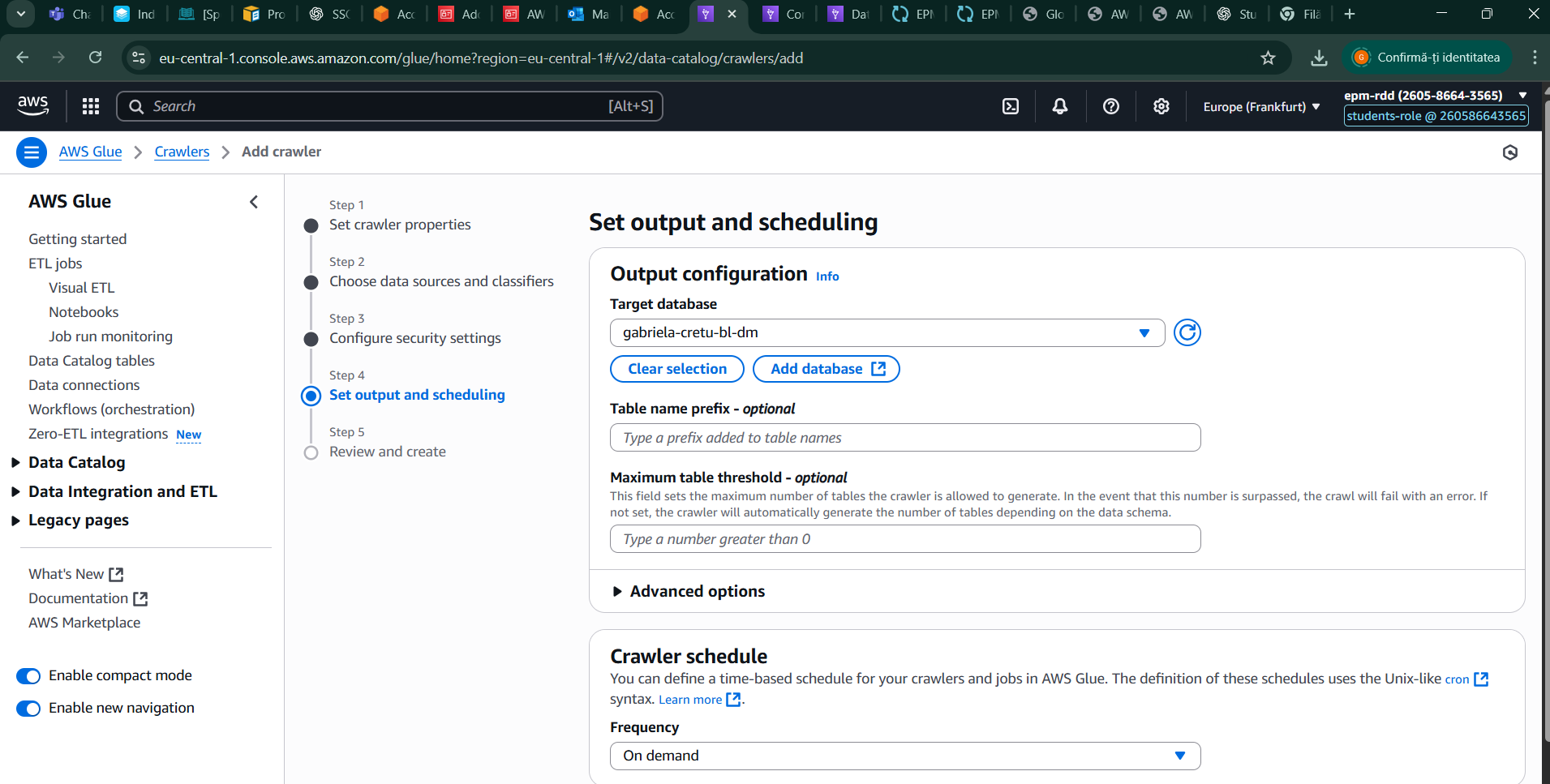
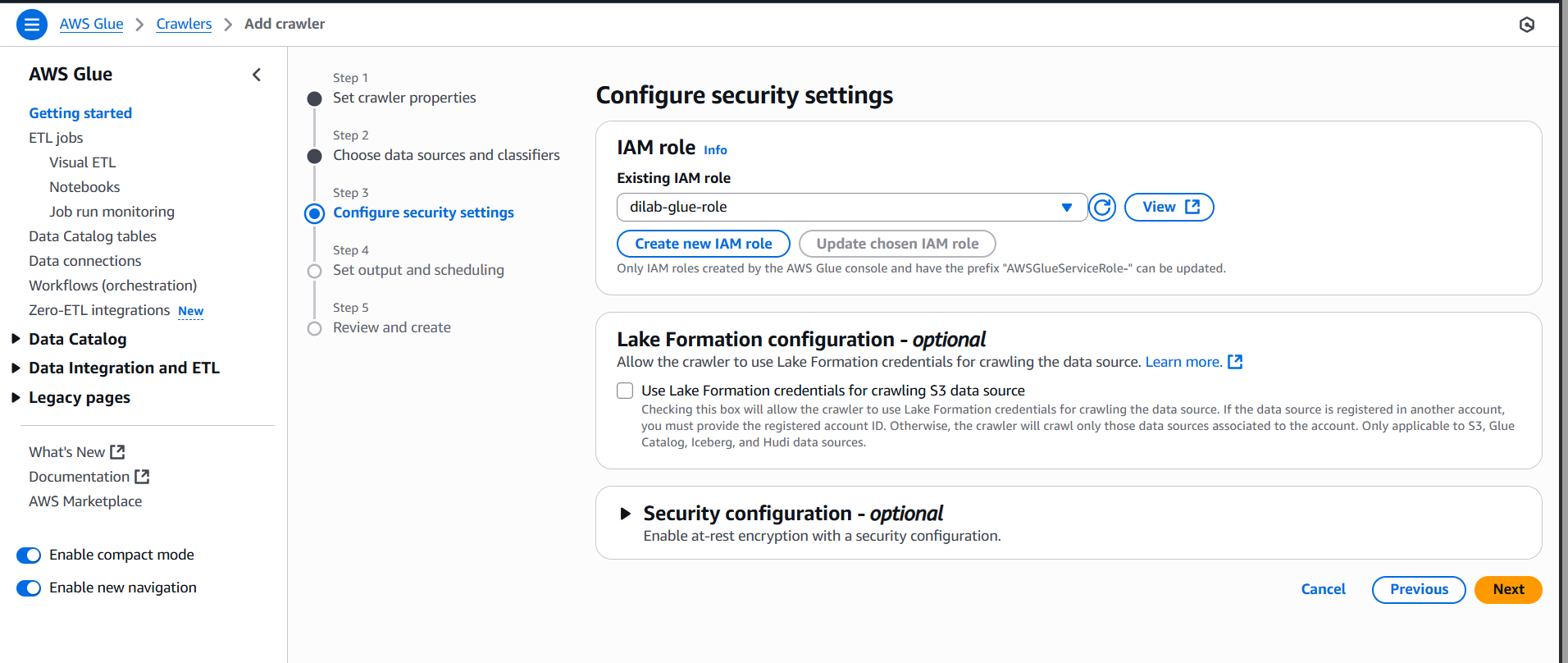


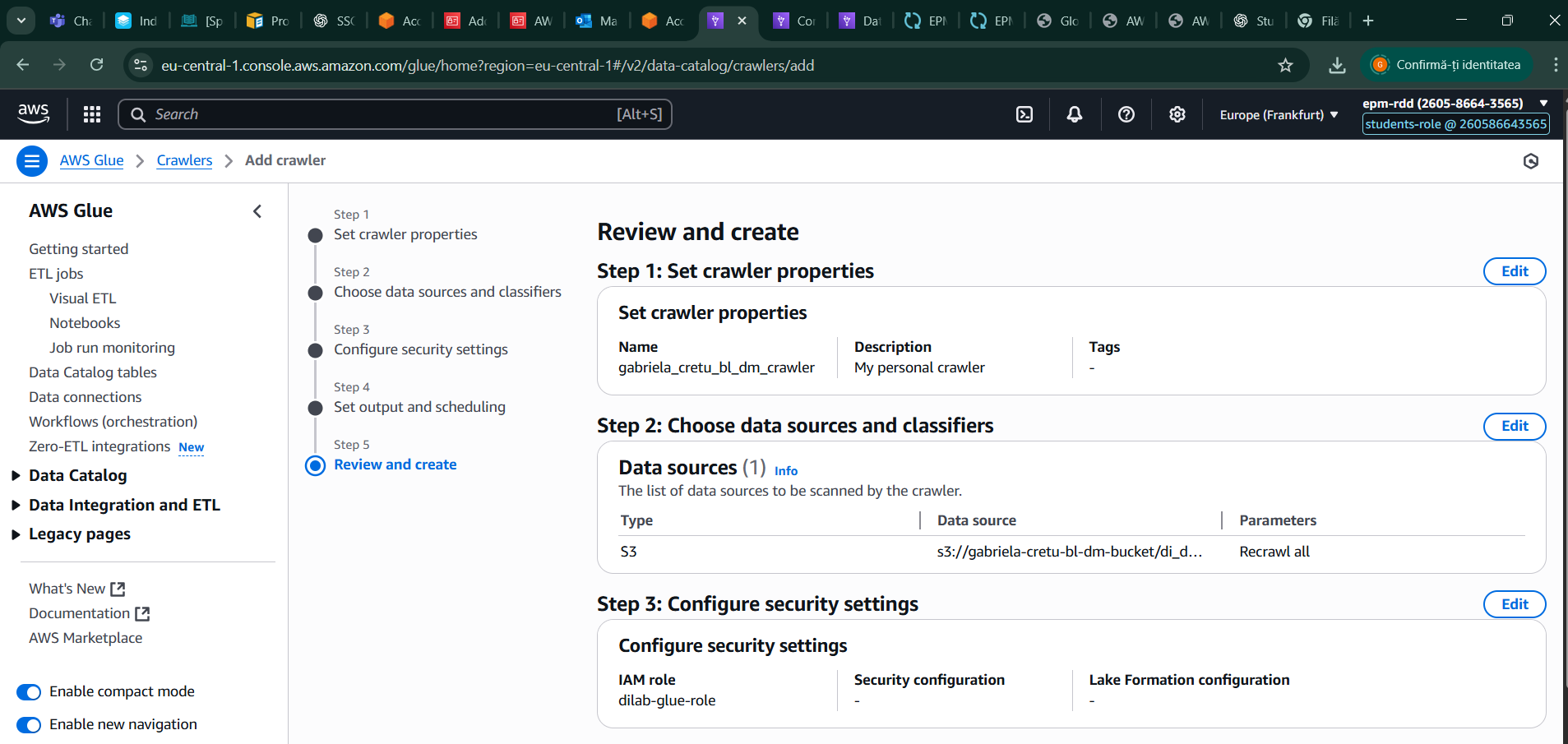


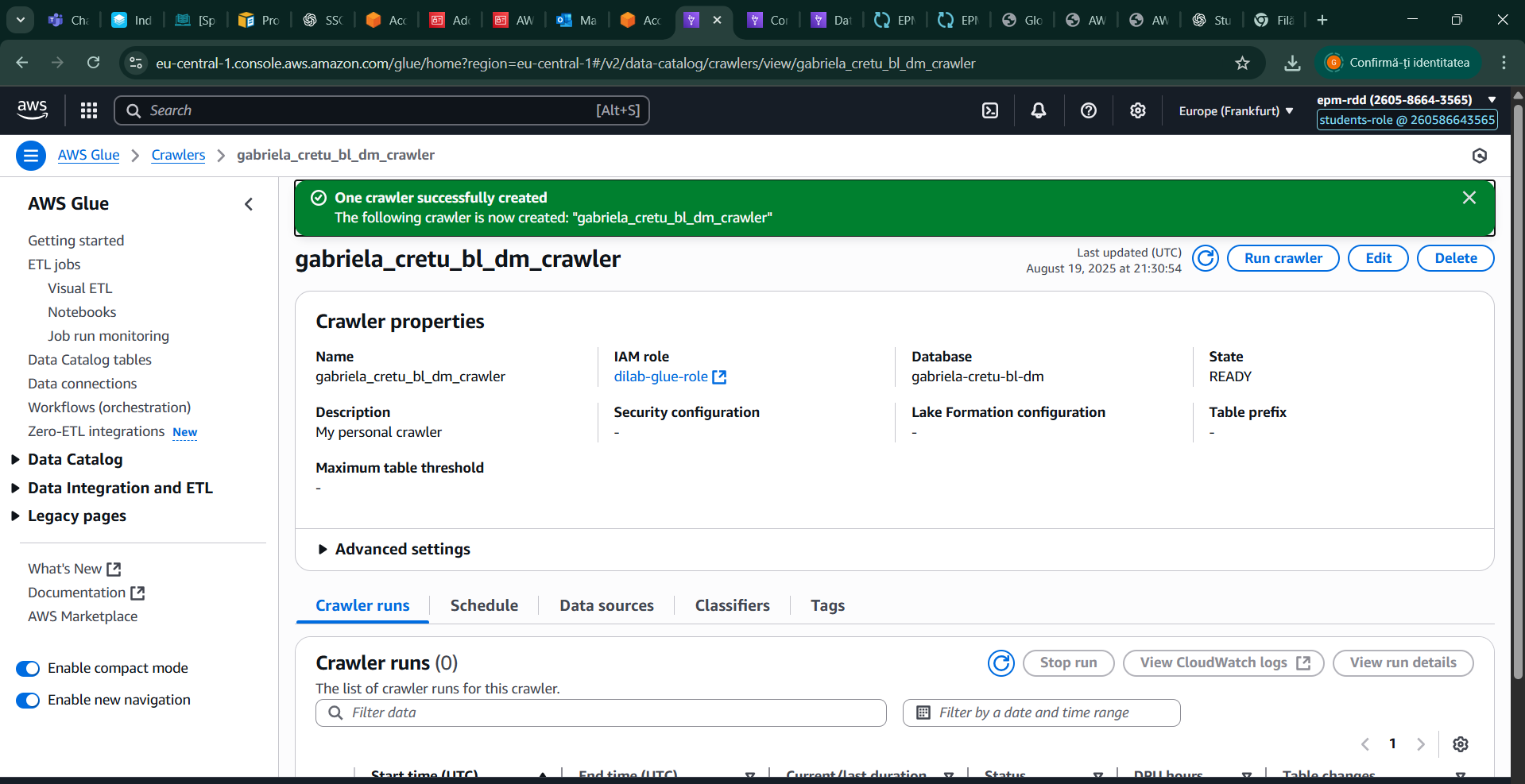
# 5. Data Catalog Setup with AWS Glue

Next, I created my crawler.

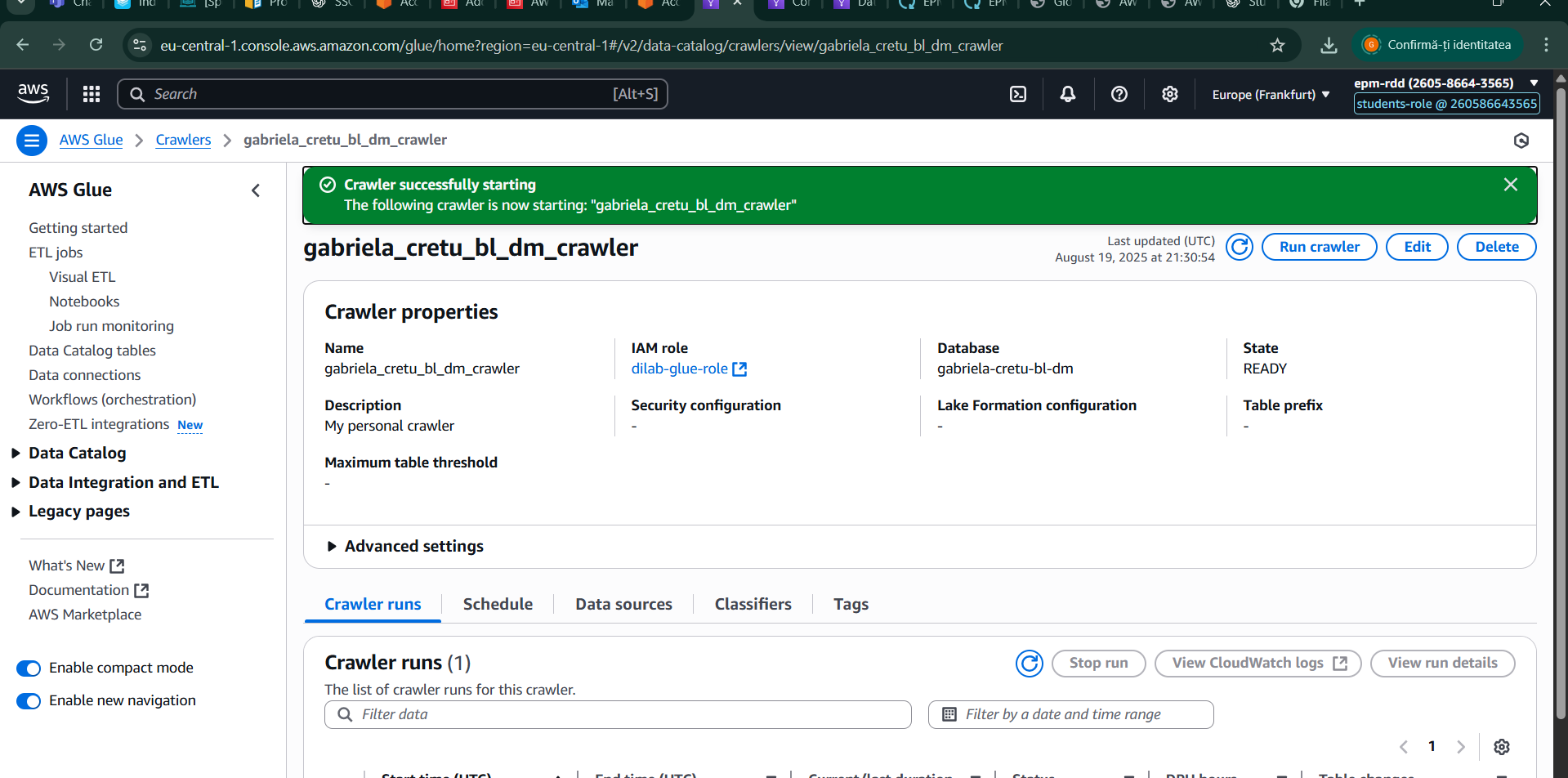


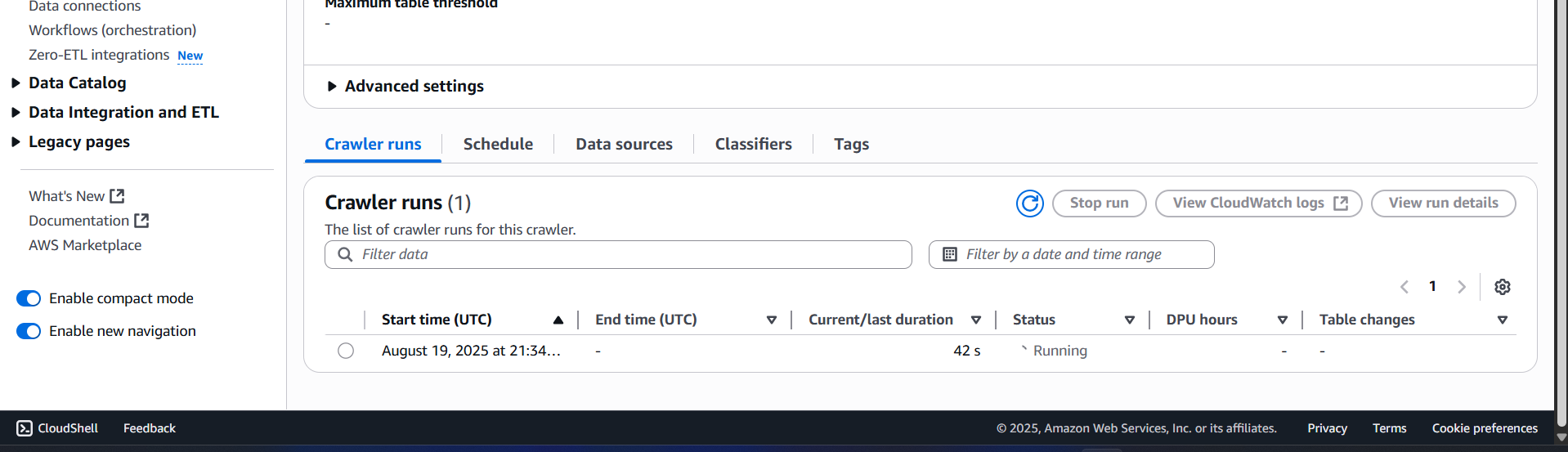






Next, I started the crawler to load my database tables into the cloud.





# 

# 

# 

# 

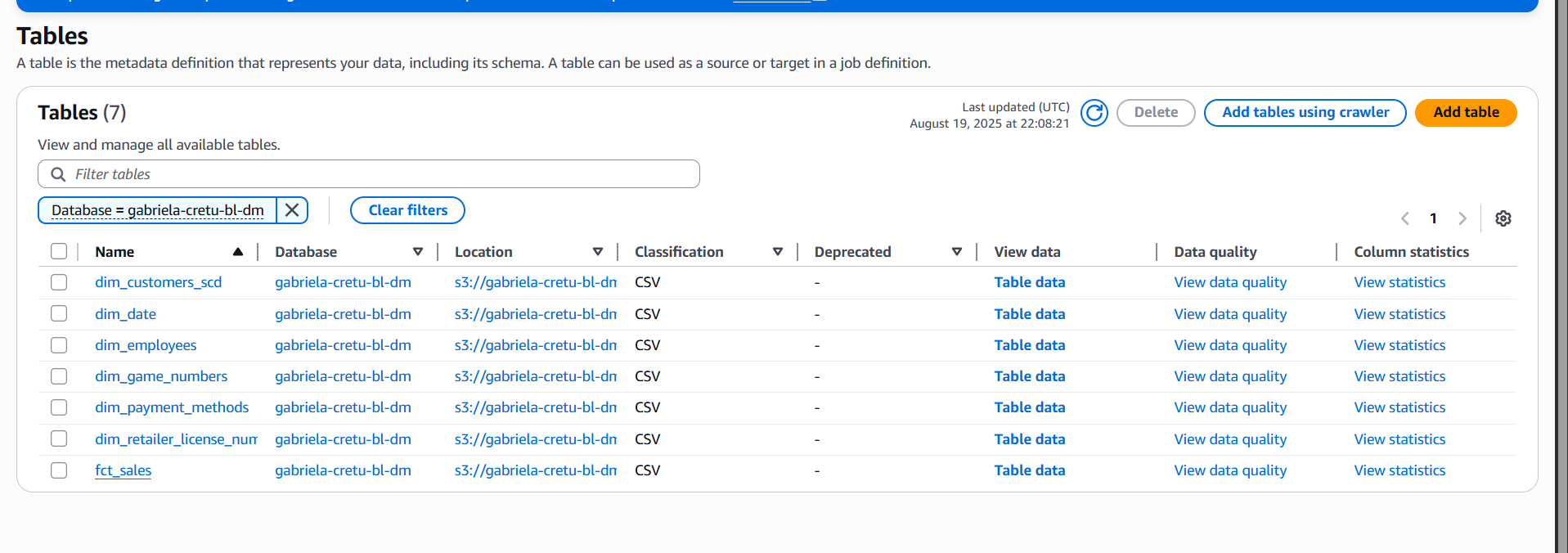
# 

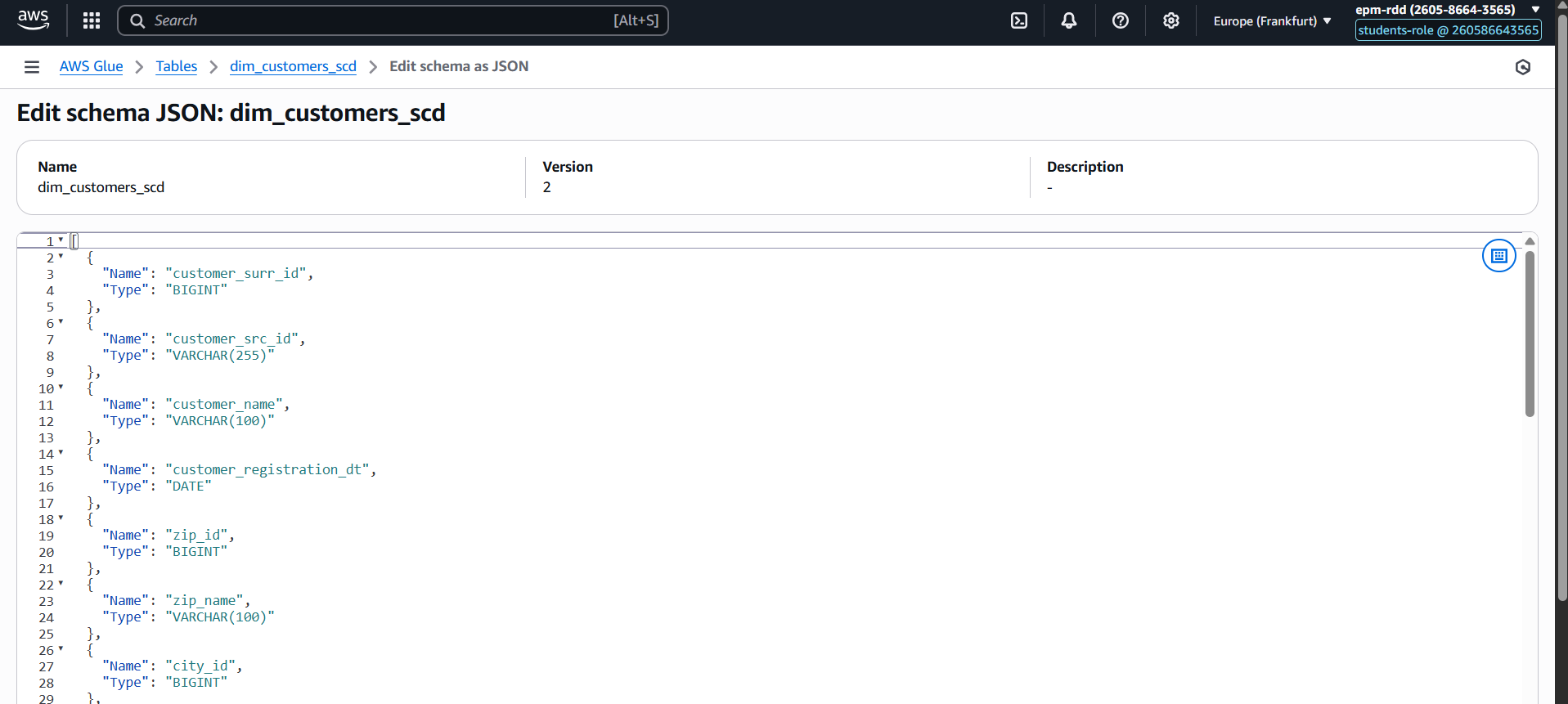
# 

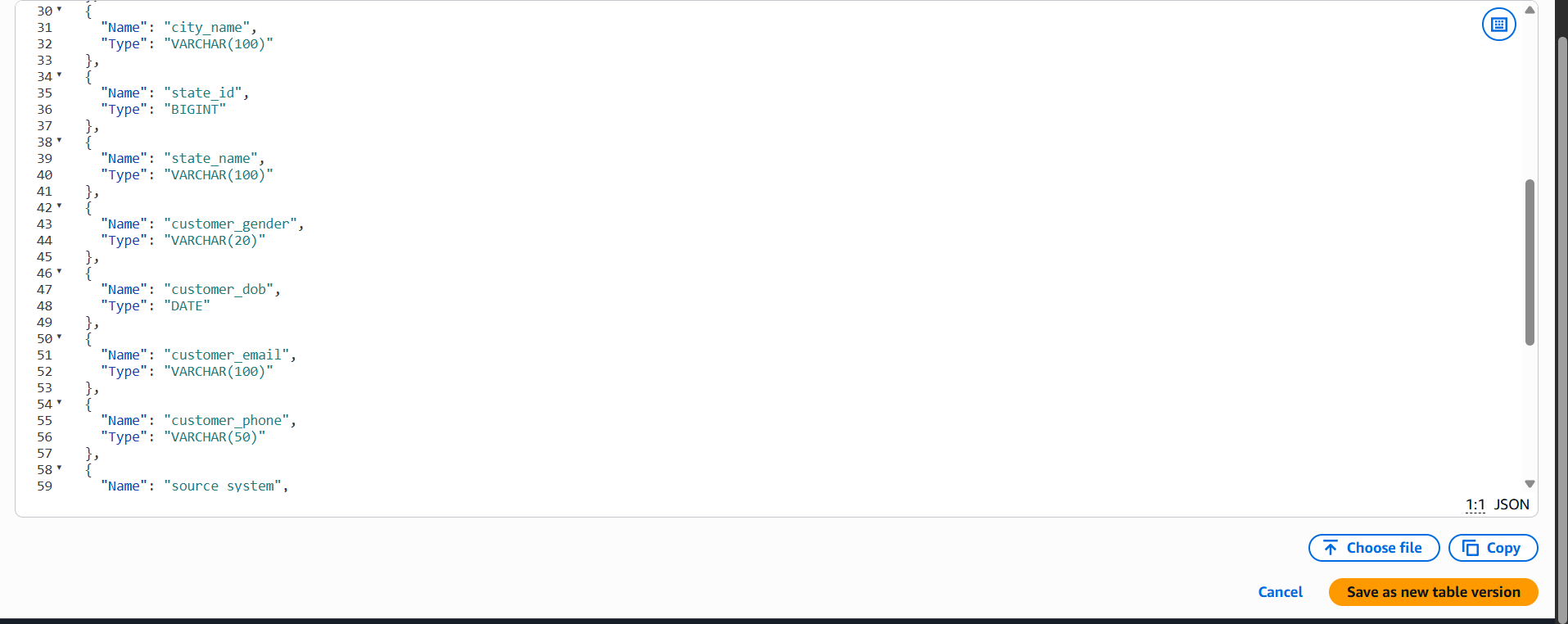
# 

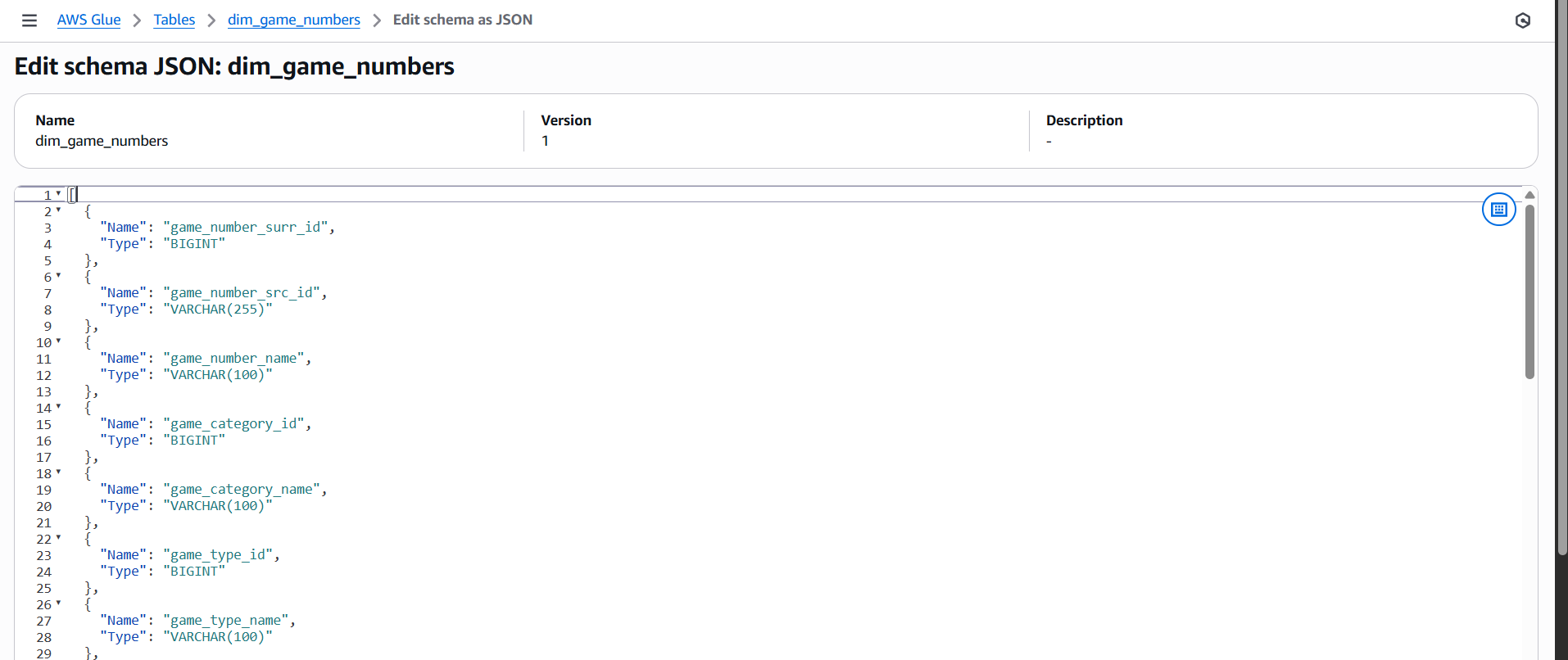
6. Data Type Validation and Schema Alignment

Before querying my tables, I first verified that the data types matched the ones I had defined for each column in the dimension and fact tables in DBeaver. Since the CSV files were not yet parsed, most columns were interpreted as strings, so I had to convert them back to their correct data types to ensure proper functionality during queries. I accomplished this by accessing the tables in AWS Glue, as shown below. Additionally, I updated the JSON code to reflect these correct data types and added partition indexes to the dimension tables based on the primary keys defined in DBeaver, providing more flexibility for querying.

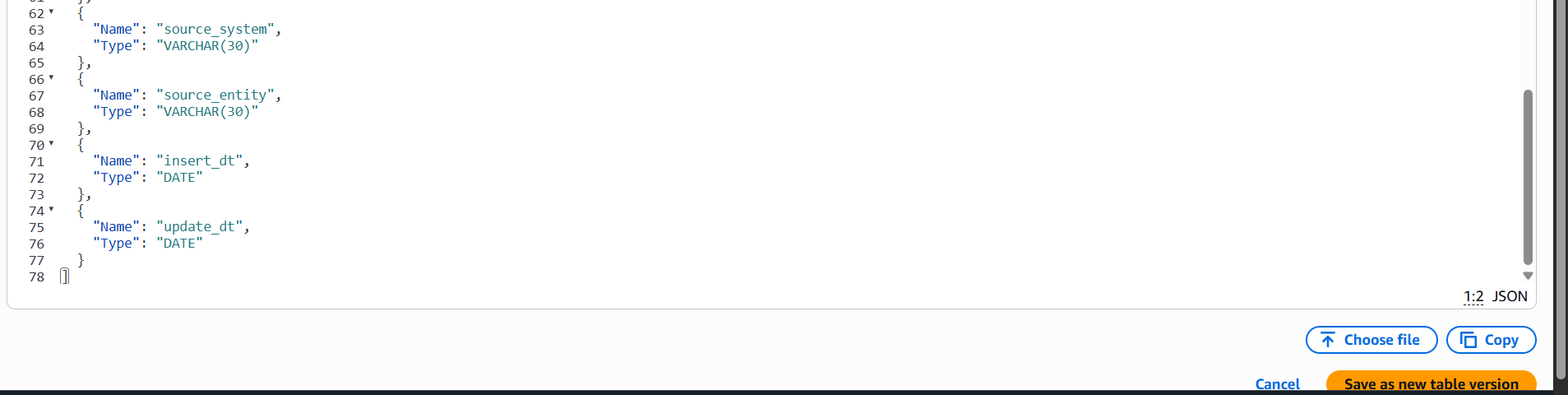


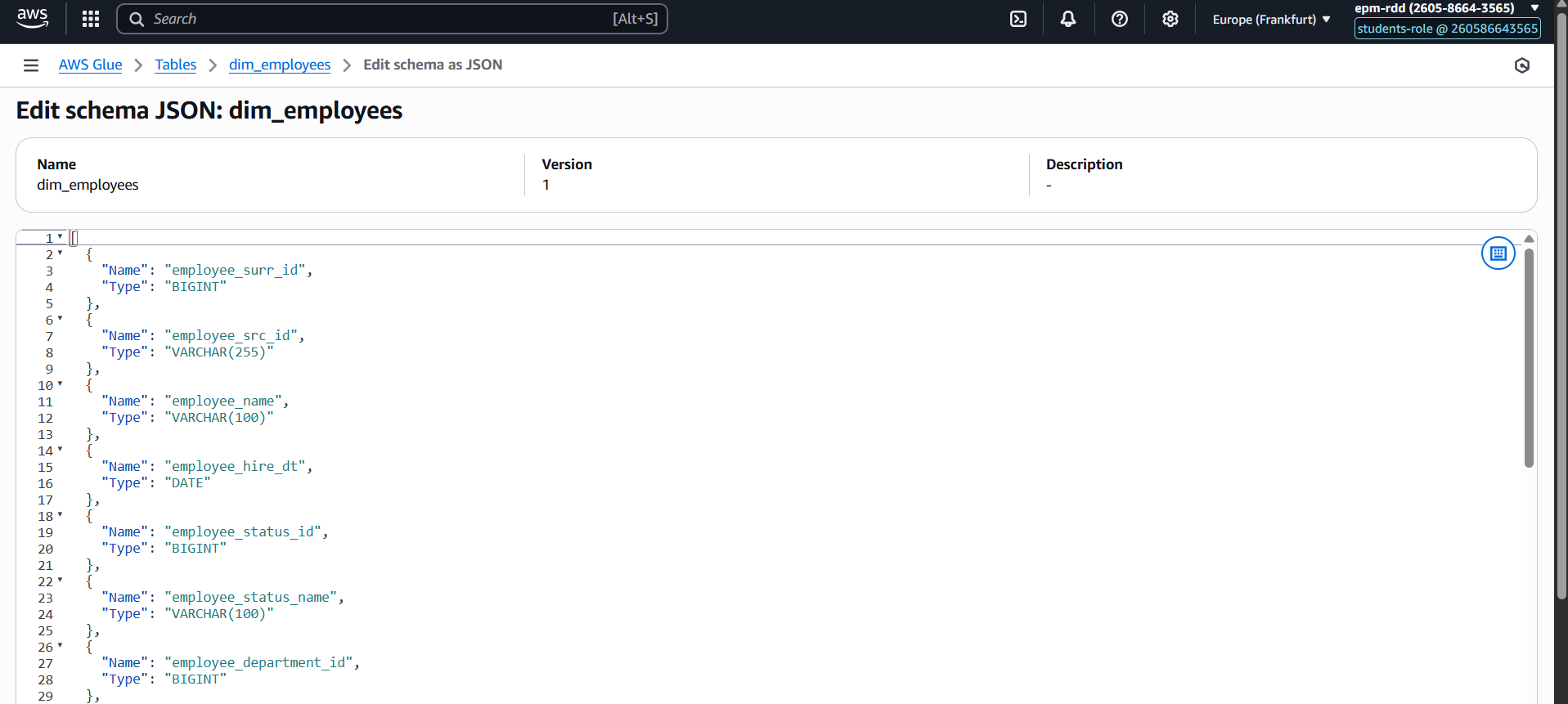


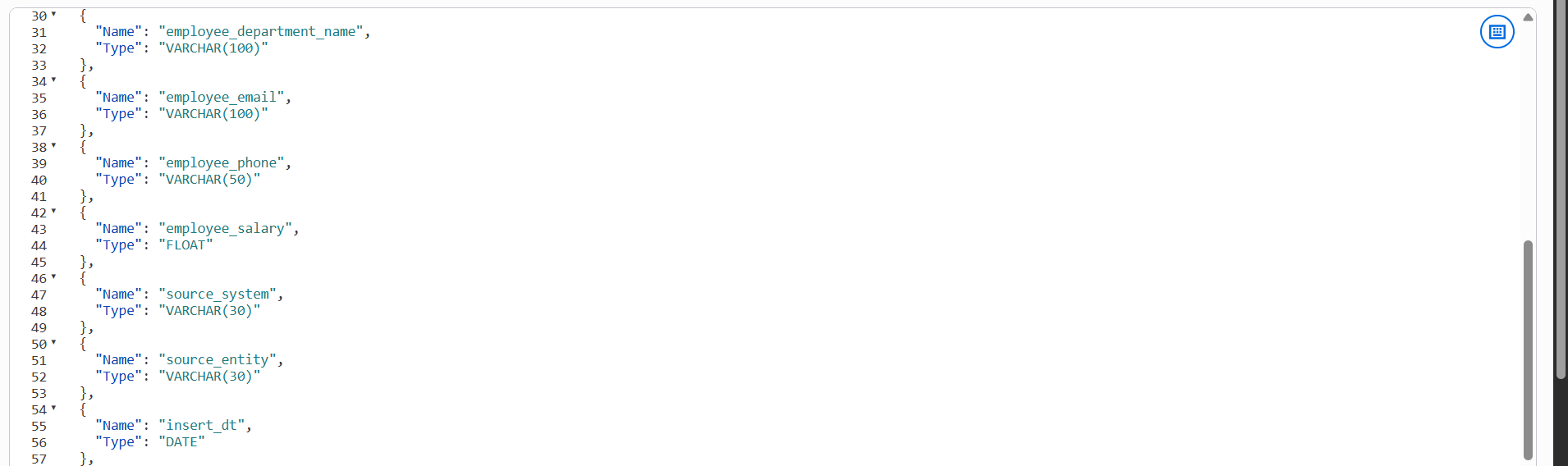


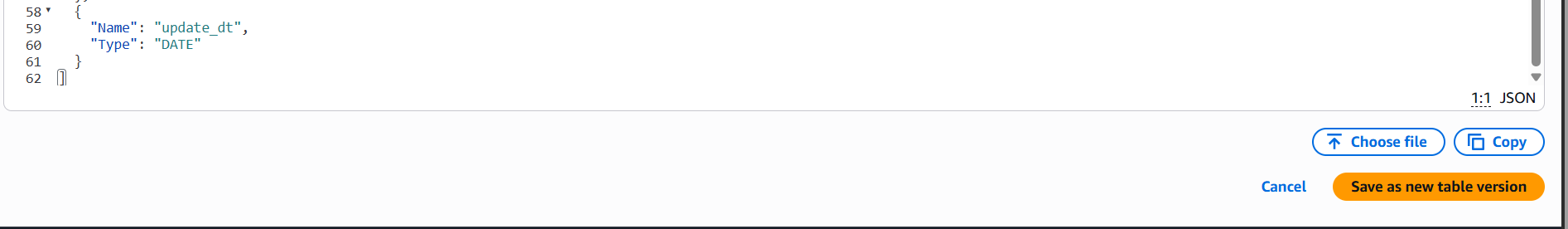


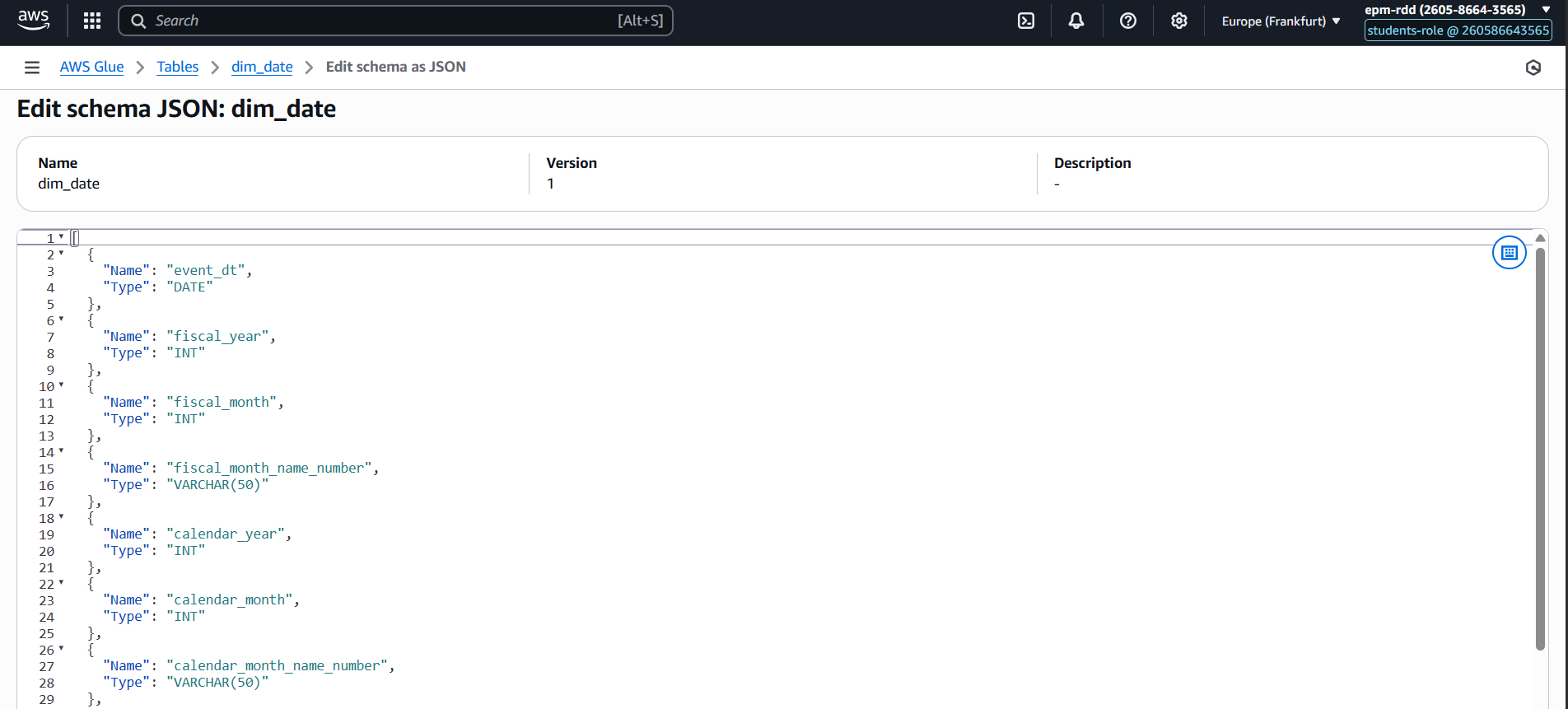


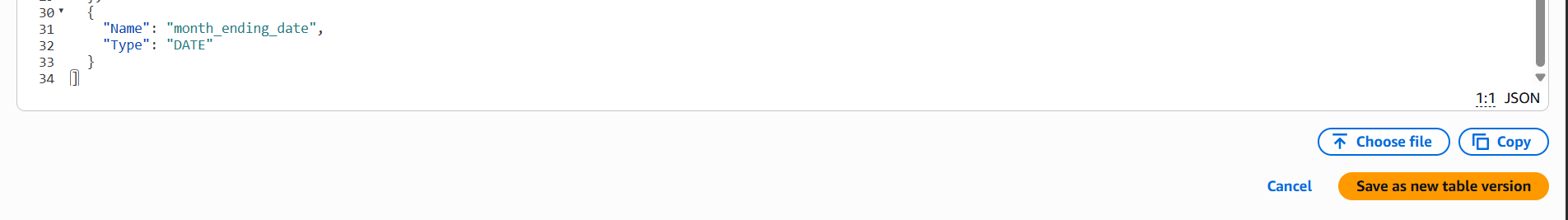






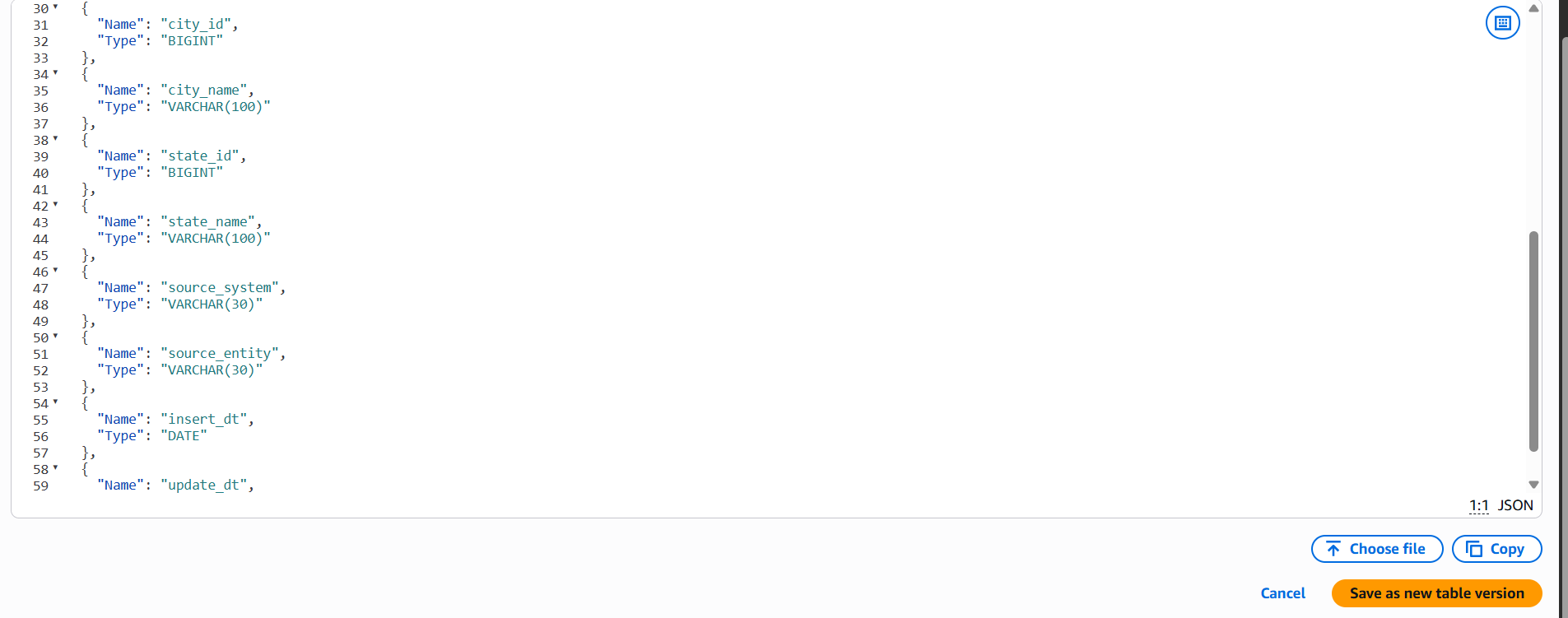


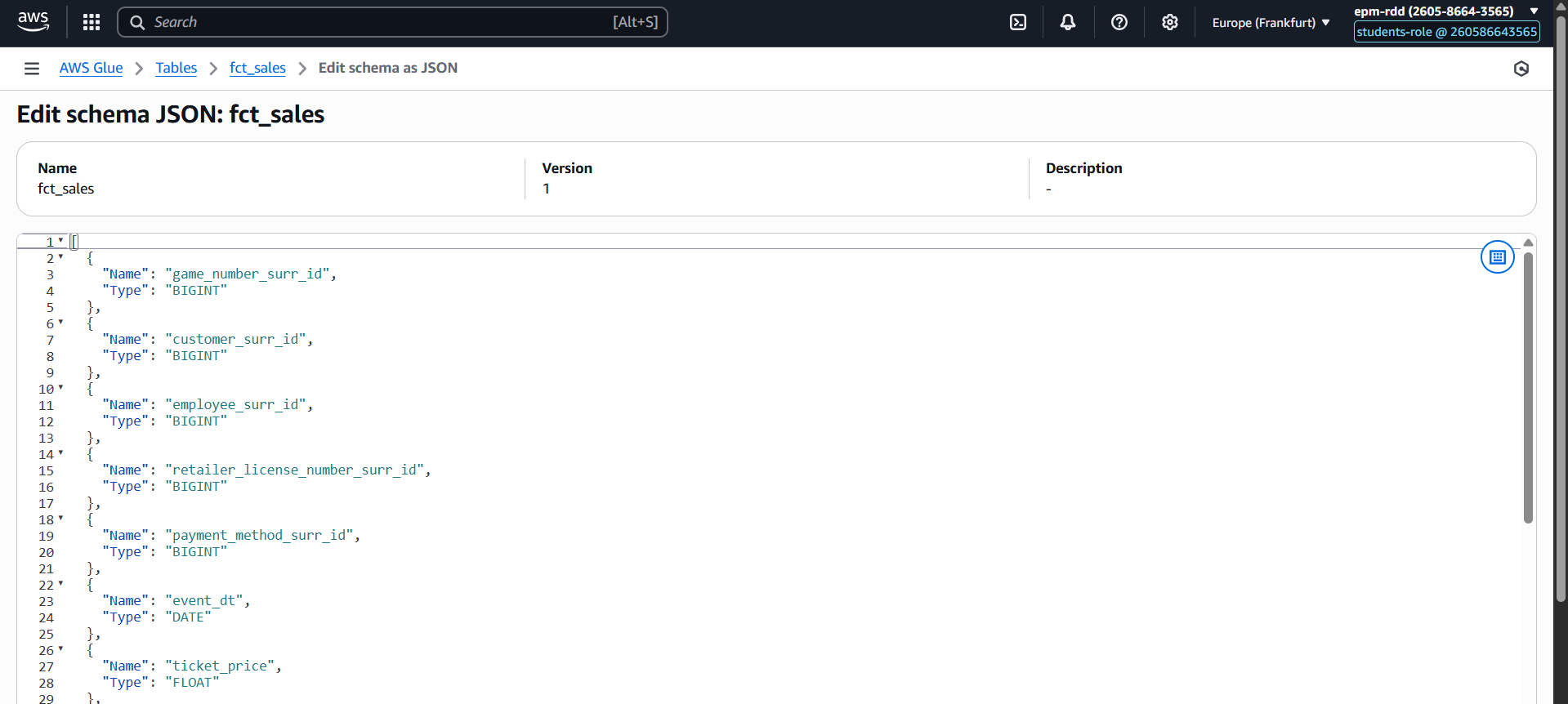


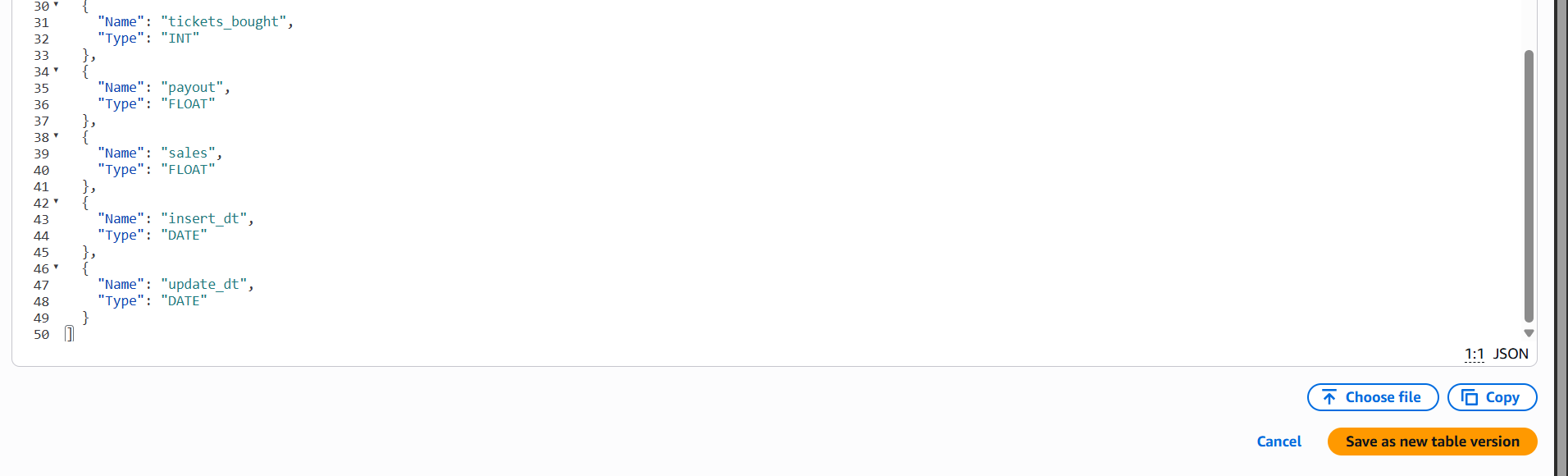












# 7. Query Execution and Validation

After making sure that the datatypes are as expected I ran some queries as seen here based on joins some select.

