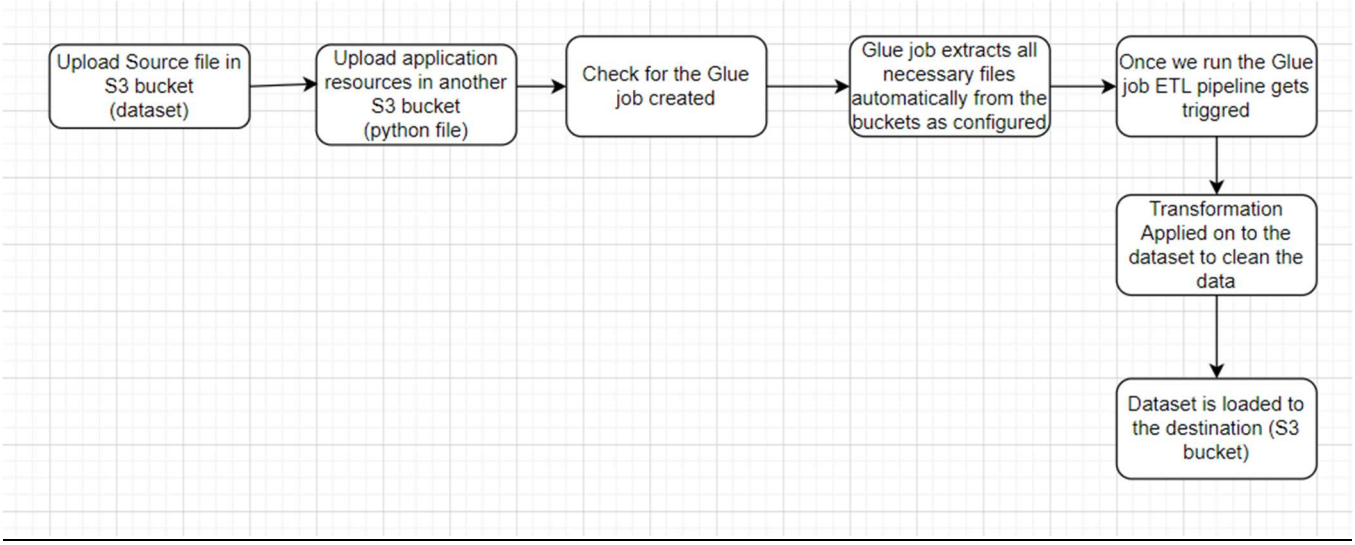
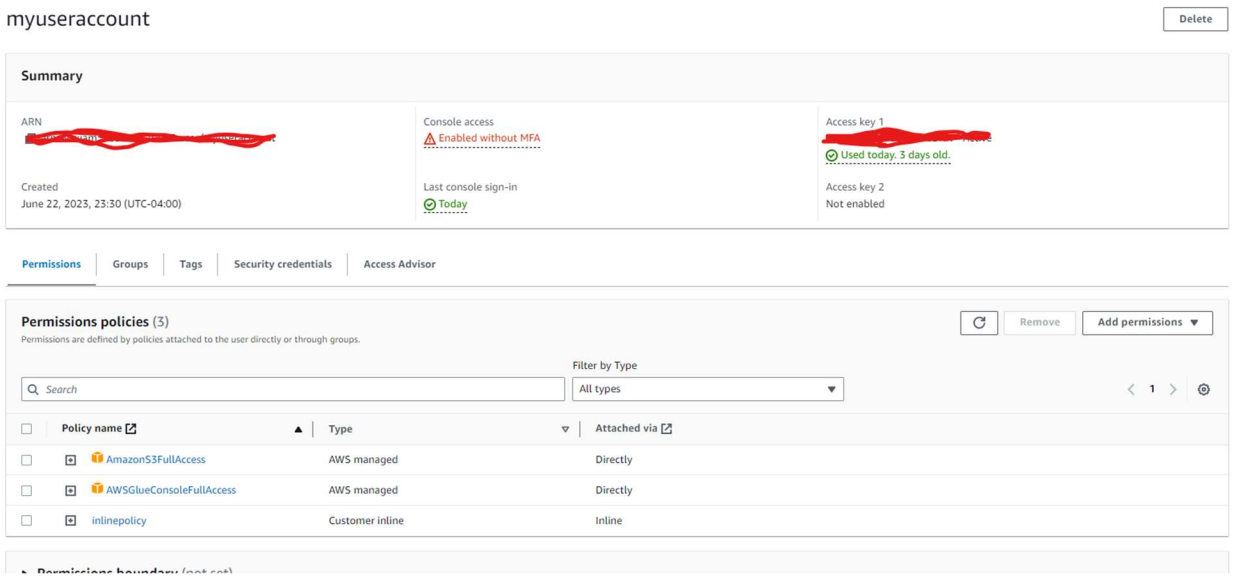
Documentation

Workflow



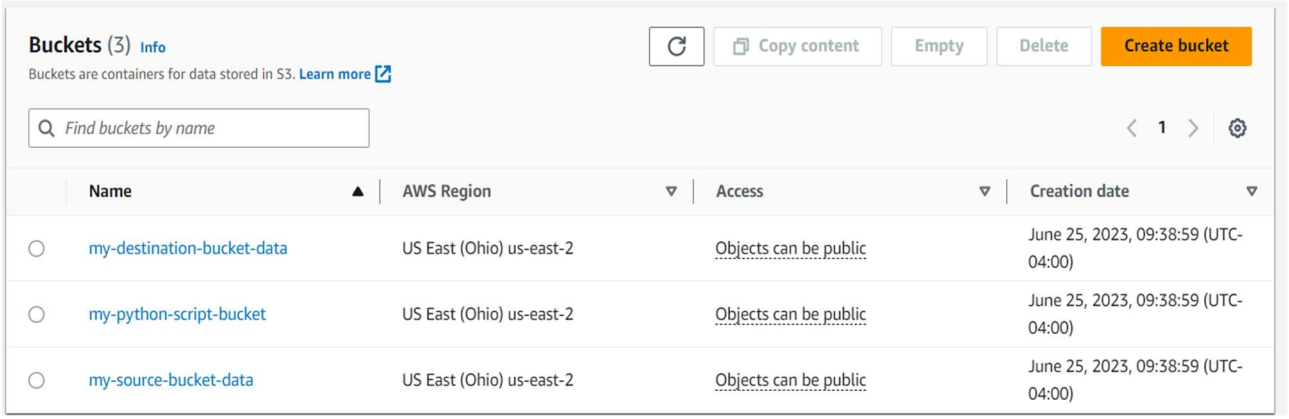
Initial Setup and configuration

Step1: Create AWS IAM user account and provide console access and programmatic access to the user account.

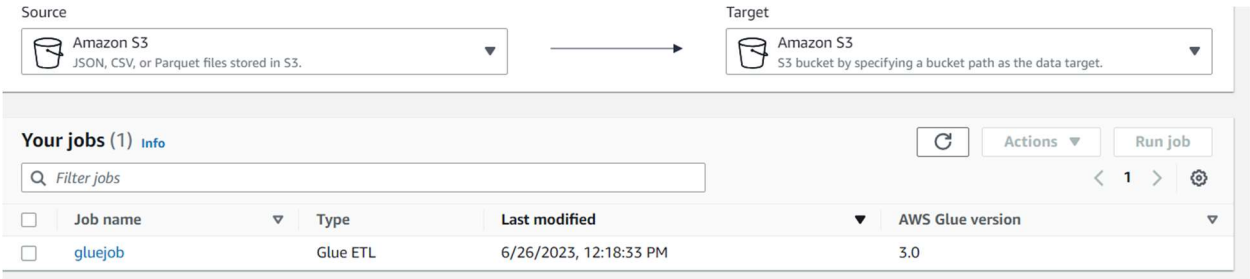


Step2: Configure the access key and secret key with the local environment and terraform installed.

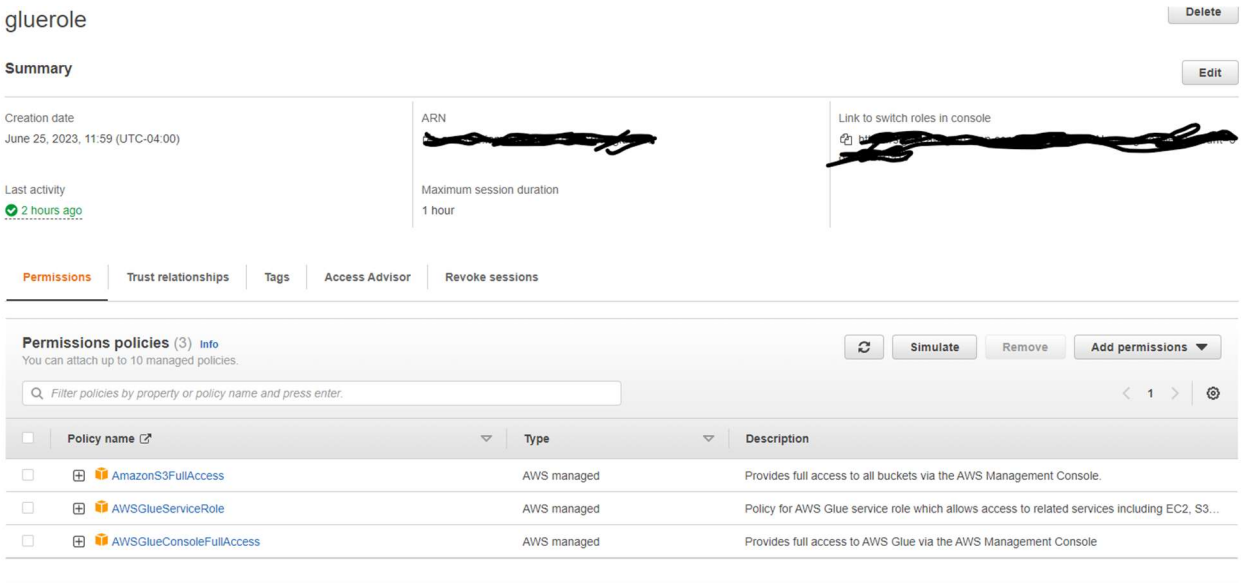
Step3: Develop Terraform scripts to provision S3 buckets for the source, destination, and to store application resources (python files).



Step4: Develop Terraform script to create a Glue job and provide path to the S3 bucket fetch the python file.



Step5: Create an IAM role with permissions to access AWS Glue and S3 service.

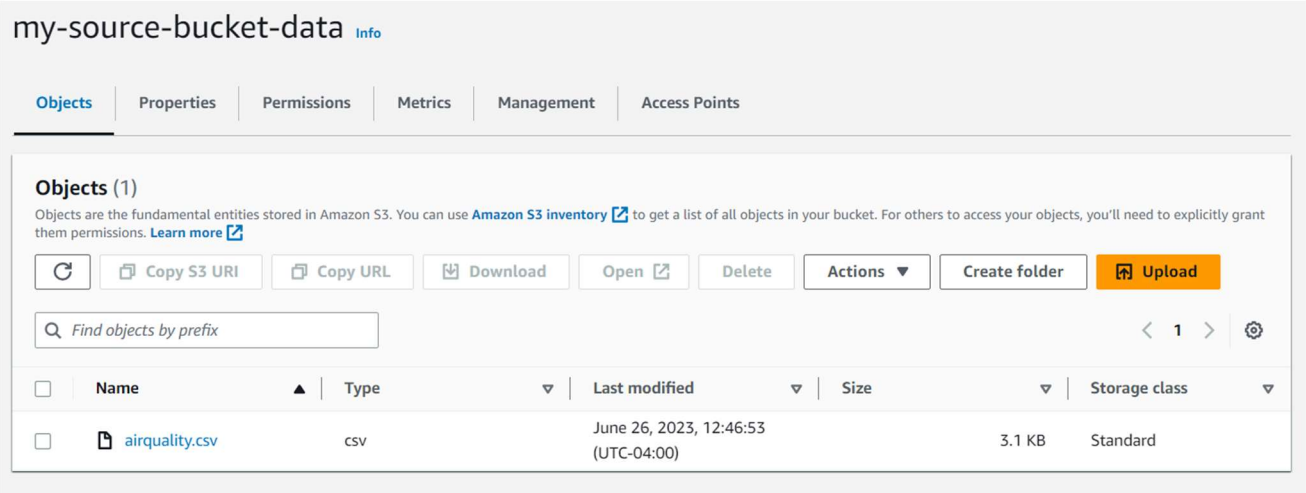


Step6: Provide AWS Glue service and S3 service permissions to the IAM user created and pass the role to the IAM user with the help of an inline policy.

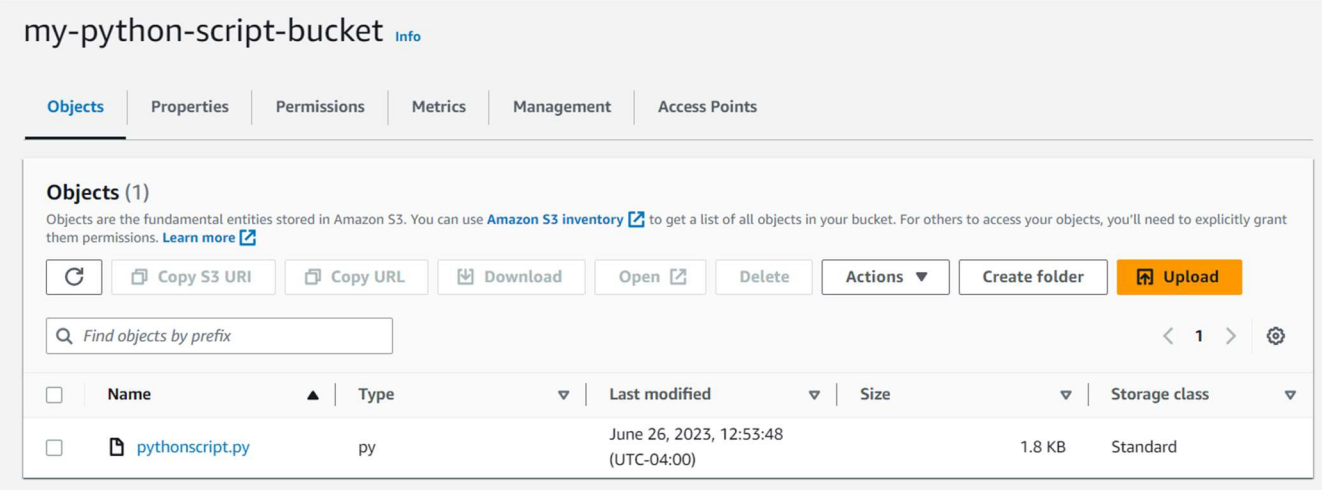
Deploying and Testing ETL pipeline

Step1: once we run “terraform apply” command all the infrastructure gets provisioned.

Step2: Ensure we have the .csv file (dataset) in source S3 bucket, if not upload the file.

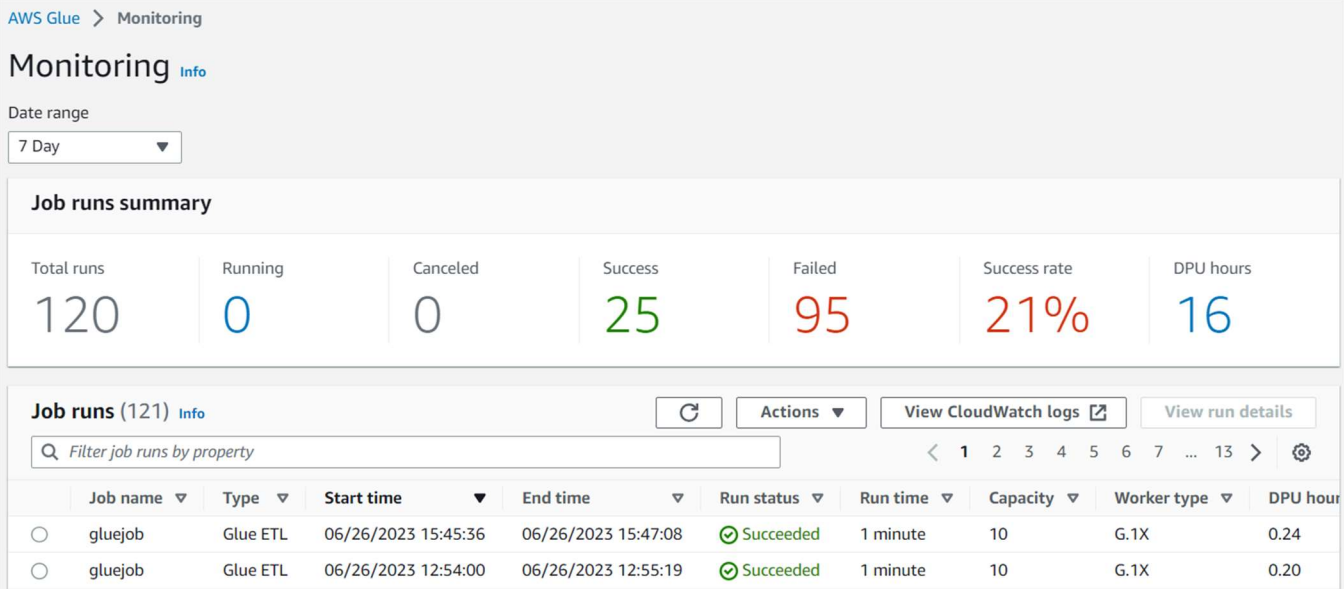


Step3: Ensure we have the python file uploaded in my-python-script-bucket, if not upload the file.

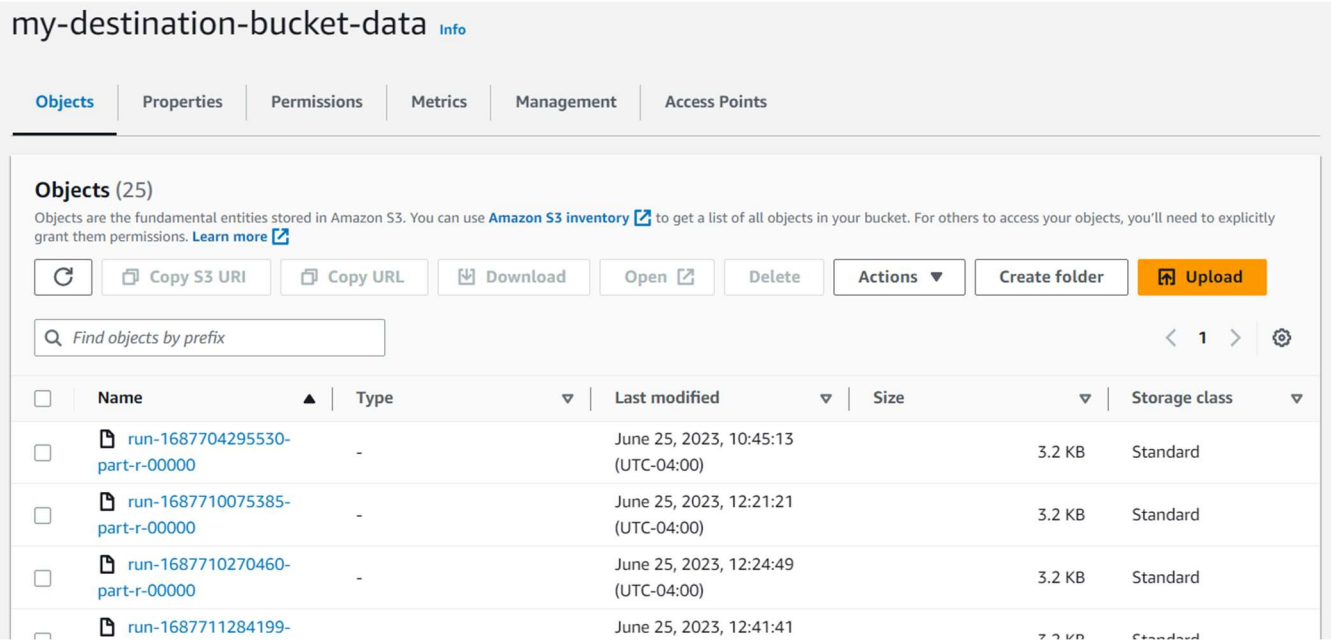


Step4: Check if the glue job has been created. If it is created, then we should be able to get the python script into the glue job (remember we configured the path of S3 bucket using terraform script).

Step5: Once we run the glue job, the python file will be able to extract the dataset, transform it and load it to the destination.



Step6: python code in the glue will check for the null values in the dataset and remove them thus handles the inconsistencies in the dataset. Then the dataset gets loaded to the destination S3 bucket.



Step7: Python script runs the ETL pipeline once the glue job is triggered.