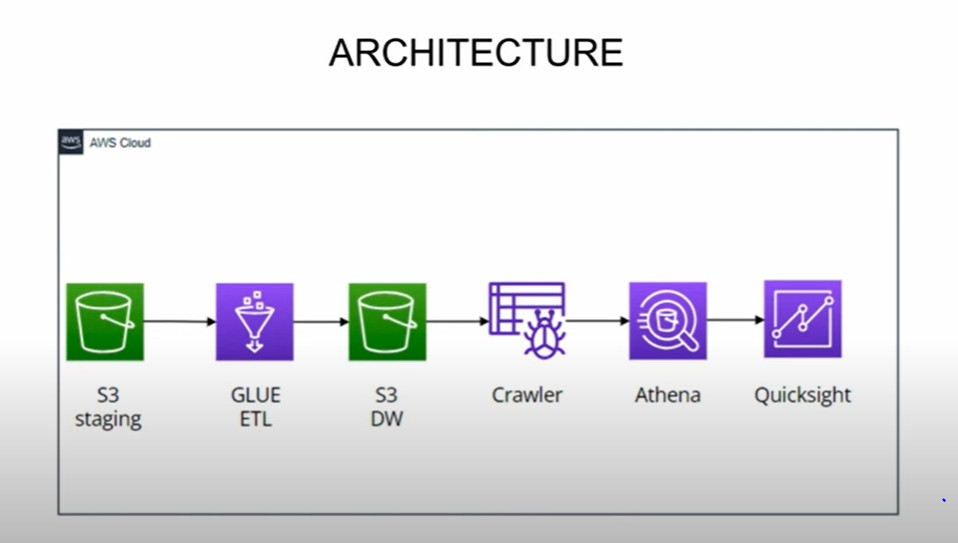
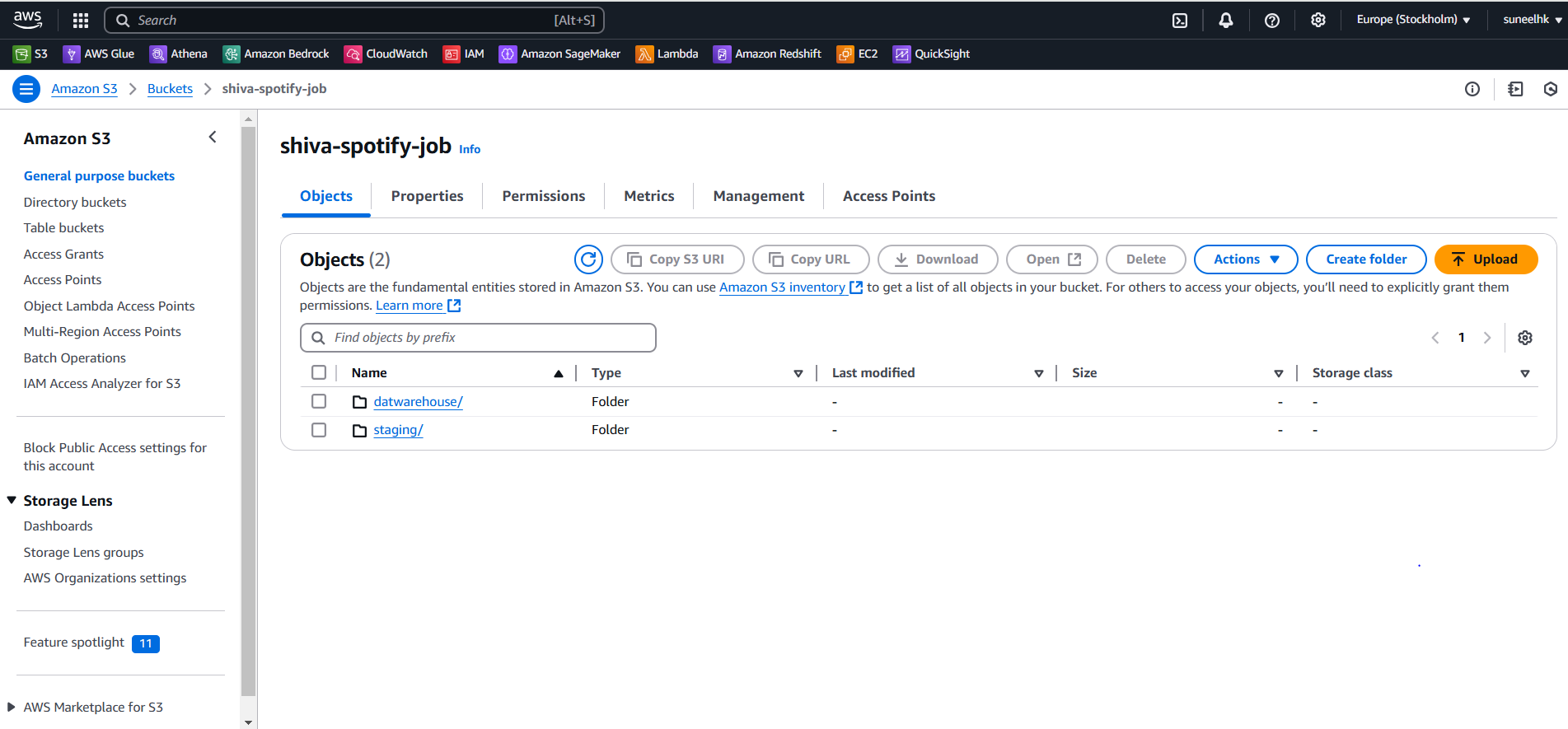
**Job Desing:**

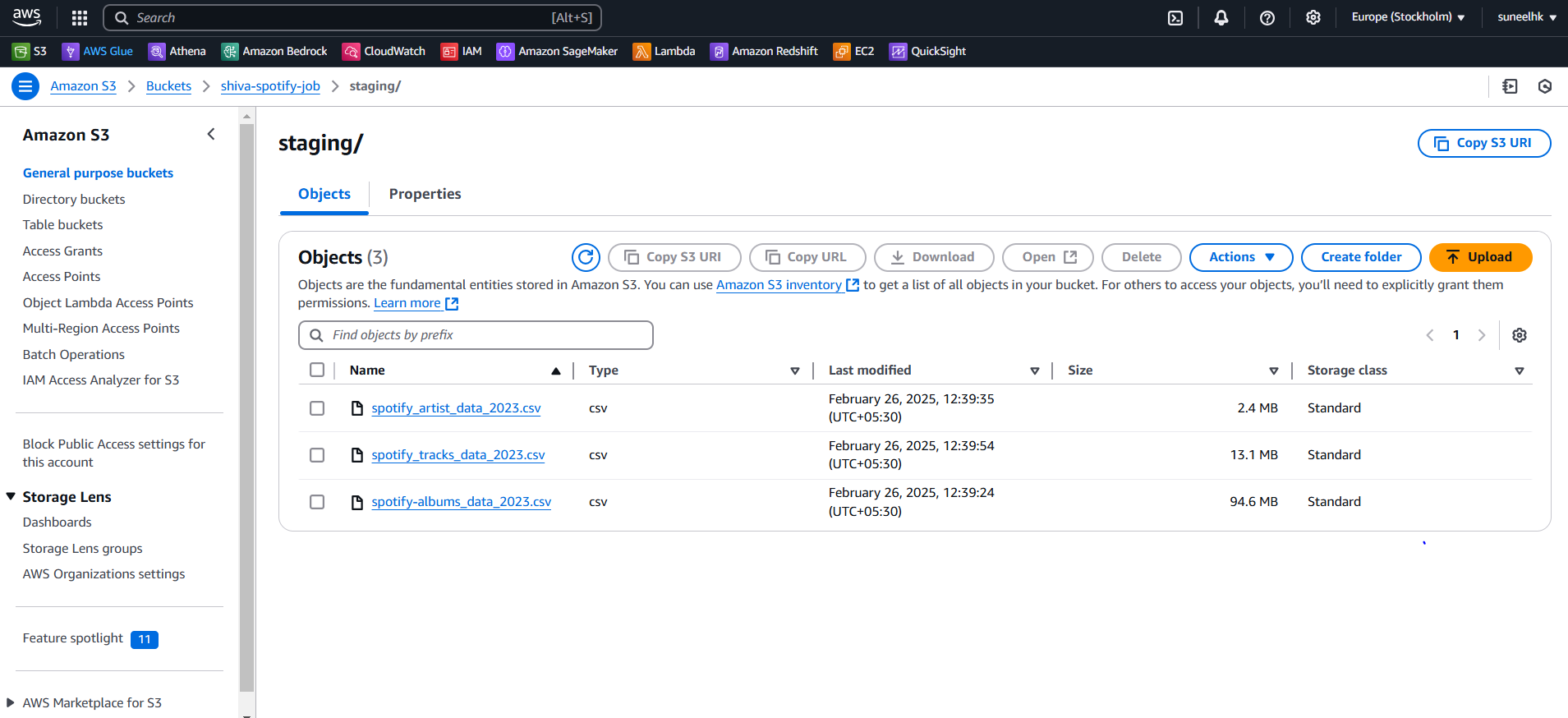


**Creating two folder for input and output:**

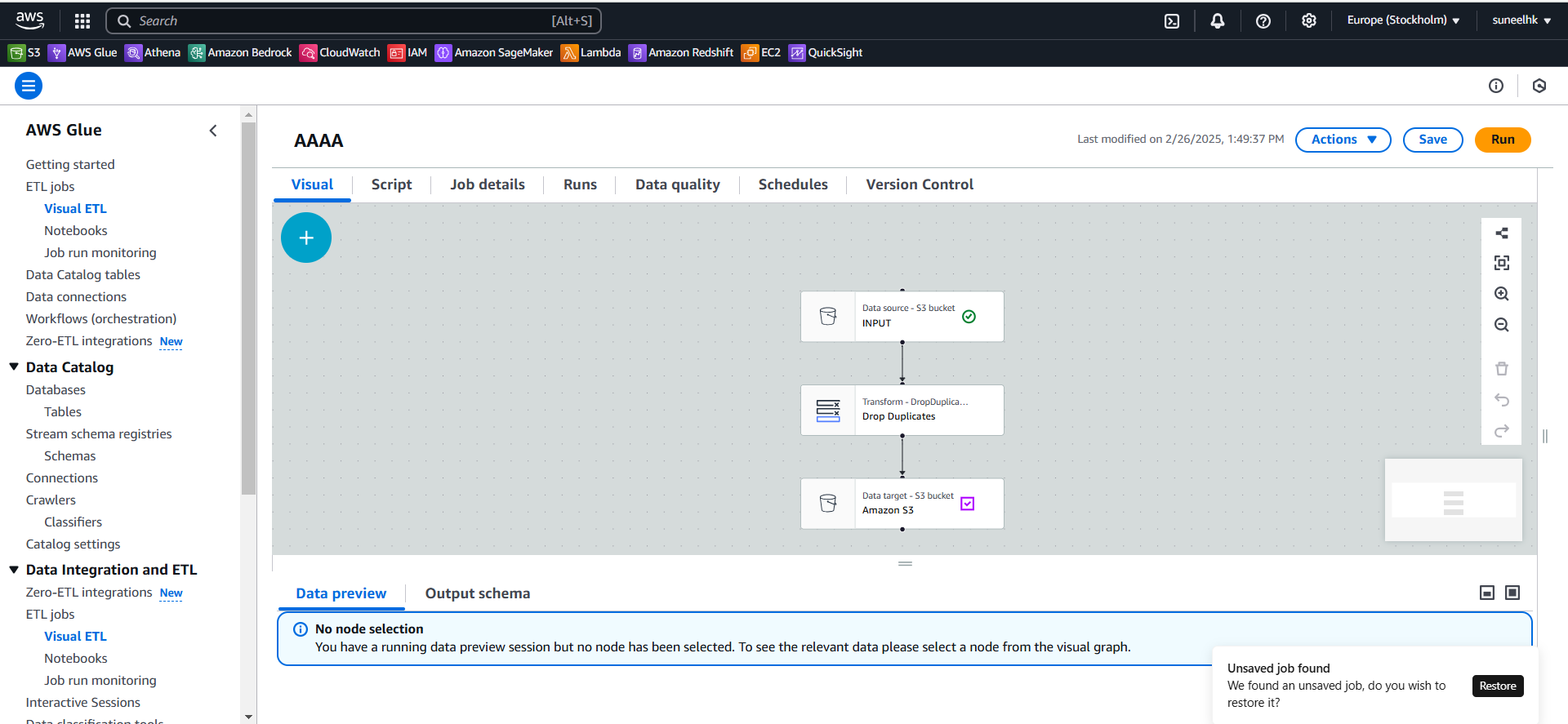
Staging is input and Dataware house in output floder



Inside staging uploaded 3 csv files.



Using AWS GLUE Create ETL Pipeline:



**Once data loaded successfully into Target.**

**Always target file format should be parquet by default.**

**Python script:**

import sys

from awsglue.transforms import \*

from awsglue.utils import getResolvedOptions

from pyspark.context import SparkContext

from awsglue.context import GlueContext

from awsglue.job import Job

from awsgluedq.transforms import EvaluateDataQuality

from awsglue.dynamicframe import DynamicFrame

from pyspark.sql import functions as SqlFuncs

args = getResolvedOptions(sys.argv, ['JOB\_NAME'])

sc = SparkContext()

glueContext = GlueContext(sc)

spark = glueContext.spark\_session

job = Job(glueContext)

job.init(args['JOB\_NAME'], args)

# Default ruleset used by all target nodes with data quality enabled

DEFAULT\_DATA\_QUALITY\_RULESET = """

Rules = [

ColumnCount > 0

]

"""

# Script generated for node INPUT

INPUT\_node1740557852420 = glueContext.create\_dynamic\_frame.from\_options(format\_options={"quoteChar": "\"", "withHeader": True, "separator": ",", "optimizePerformance": False}, connection\_type="s3", format="csv", connection\_options={"paths": ["s3://shiva-spotify-job/staging/spotify-albums\_data\_2023.csv"], "recurse": True}, transformation\_ctx="INPUT\_node1740557852420")

# Script generated for node Drop Duplicates

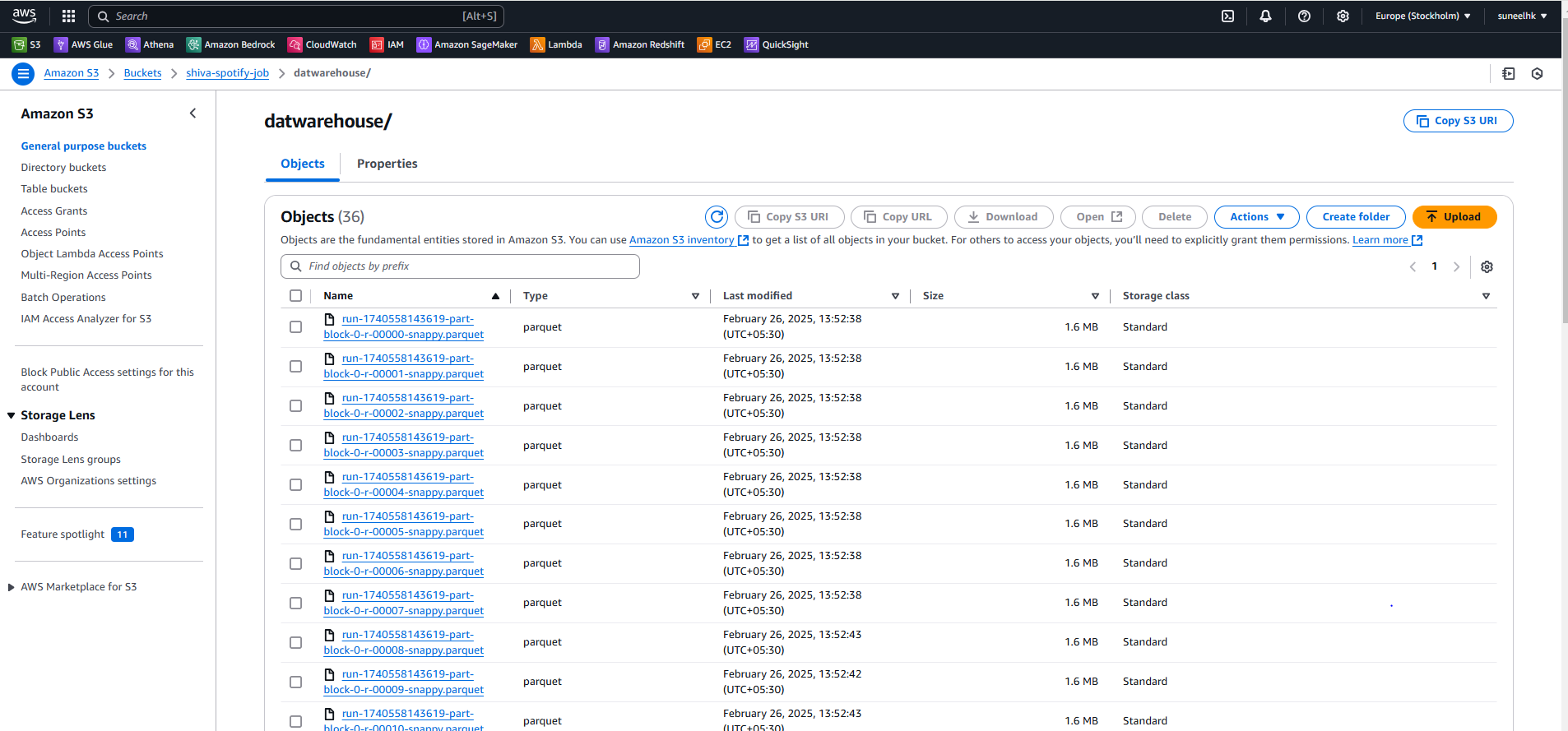
DropDuplicates\_node1740557886076 = DynamicFrame.fromDF(INPUT\_node1740557852420.toDF().dropDuplicates(), glueContext, "DropDuplicates\_node1740557886076")

# Script generated for node Amazon S3

EvaluateDataQuality().process\_rows(frame=DropDuplicates\_node1740557886076, ruleset=DEFAULT\_DATA\_QUALITY\_RULESET, publishing\_options={"dataQualityEvaluationContext": "EvaluateDataQuality\_node1740557844090", "enableDataQualityResultsPublishing": True}, additional\_options={"dataQualityResultsPublishing.strategy": "BEST\_EFFORT", "observations.scope": "ALL"})

AmazonS3\_node1740557894668 = glueContext.write\_dynamic\_frame.from\_options(frame=DropDuplicates\_node1740557886076, connection\_type="s3", format="glueparquet", connection\_options={"path": "s3://shiva-spotify-job/datwarehouse/", "partitionKeys": []}, format\_options={"compression": "snappy"}, transformation\_ctx="AmazonS3\_node1740557894668")

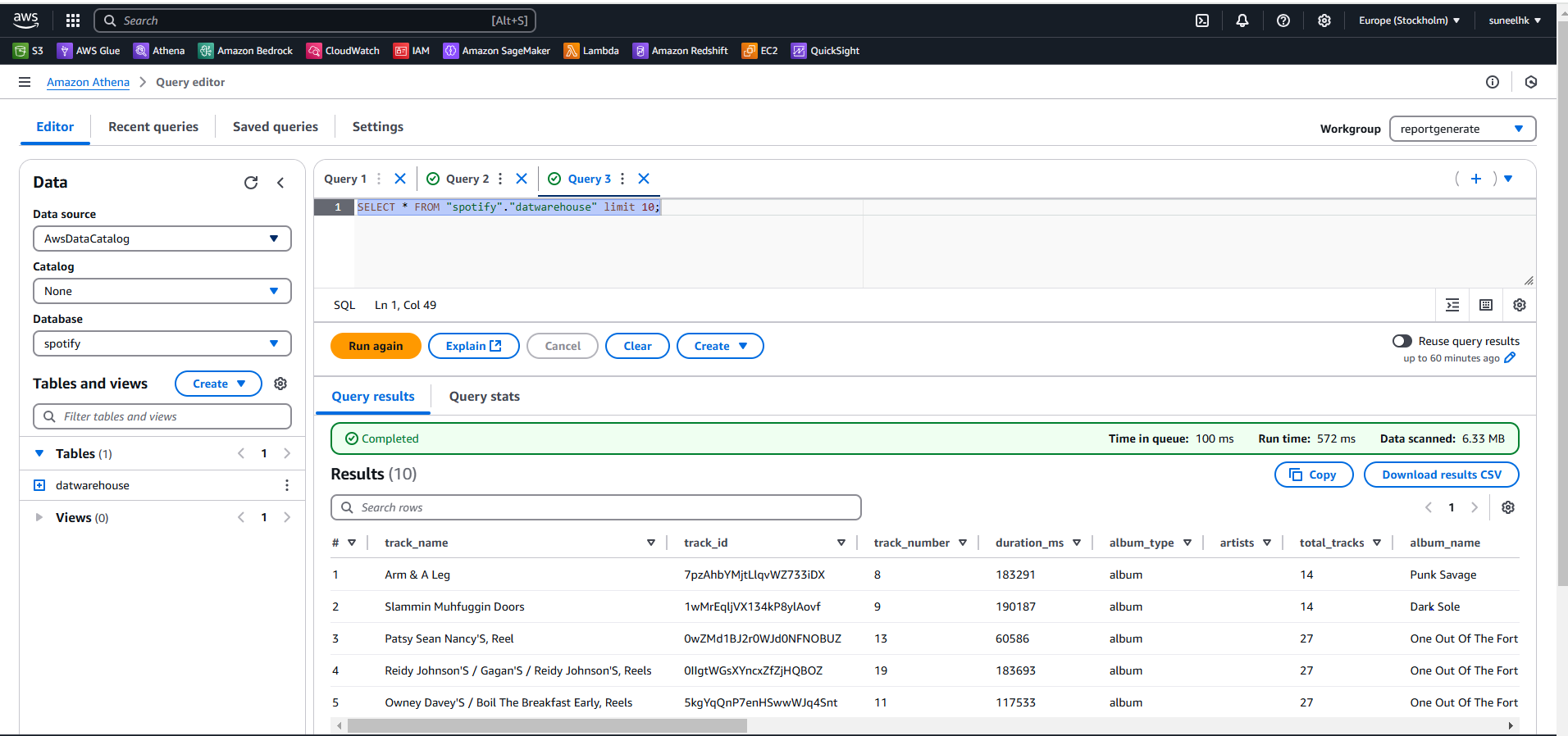
job.commit()



Once data loaded into target create table name and database name and convert that data into table using crawler.

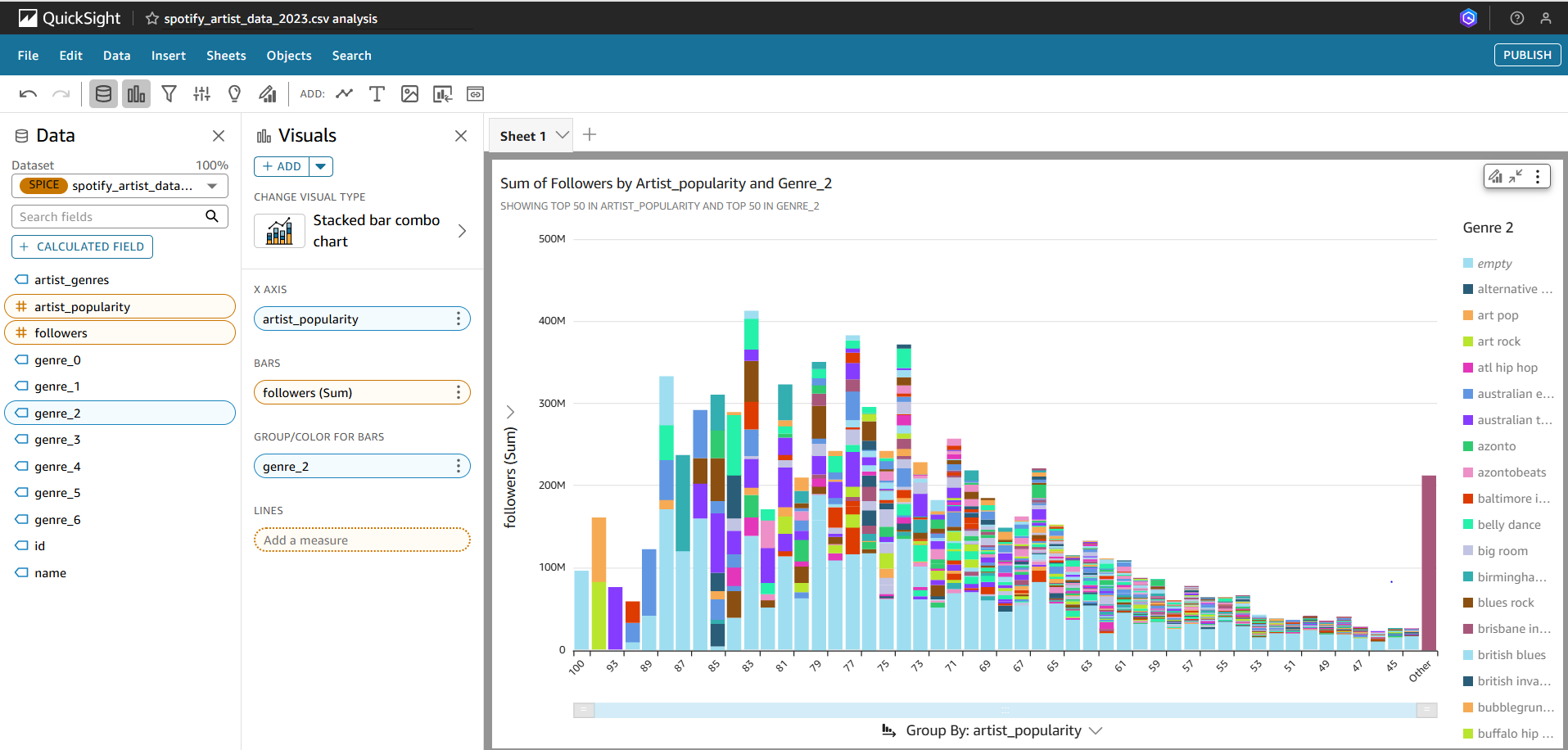
**Table name : spotify**

**Data base name : Datawarehouse**



**Using AWS quick sight we can form a graphs for analysis.**

**Bar chat:**



**Pi Chart :** 