Create a ETL pipeline on AWS

**✅ CSV in S3 → Transform using Glue → Store output in S3 (as CSV or Parquet)**

No RDS, no Redshift — completely manageable, and cheap or free depending on how long the job runs.

**🔧 What You’ll Need (Before We Start)**

1. **An S3 Bucket** (we’ll use this for both input and output)
2. **A sample CSV file** (I’ll give you one)
3. **AWS Glue IAM Role** with permissions to access S3
4. **Glue Job (Spark or Python Shell)**

✅ Step-by-Step: Build the ETL Pipeline

**STEP 1: Create a Sample CSV File**

Here’s a sample file: customers.csv

customer\_id,name,email,country

1,Alice,alice@example.com,USA

2,Bob,bob@example.com,UK

3,Charlie,charlie@example.com,India

4,David,david@example.com,India

**Save it locally** and upload it to your S3 bucket, e.g.:

s3://your-bucket-name/input/customers.csv

**STEP 2: Create a Glue Crawler**

1. Go to **AWS Glue > Crawlers**
2. Click **Add crawler**
3. Name it: customer-csv-crawler
4. Source type: **Data stores**
5. Choose: **S3**, point it to: s3://your-bucket-name/input/
6. IAM Role: Choose an existing or create a new one with access to S3
7. Output: Choose or create a Glue Database (e.g. demo\_db)
8. Run the crawler

💡 **Result**: It creates a table in Glue Data Catalog for your CSV.

**STEP 3: Create Glue Job to Transform CSV**

Go to **AWS Glue > Jobs > Add Job**

**Settings:**

* Name: transform\_customers\_csv
* IAM Role: same as crawler
* Type: Spark
* Script: Choose **“Script generated by Glue”** or **“Author script yourself”**

**🧪 Example Transformation Script**

Here’s a full script that reads the CSV, filters Indian customers, and writes the result to S3:

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

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

sc = SparkContext()

glueContext = GlueContext(sc)

spark = glueContext.spark\_session

job = Job(glueContext)

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

# Read from catalog

datasource = glueContext.create\_dynamic\_frame.from\_catalog(

database="demo\_db",

table\_name="database-avi", # this is created by crawler

transformation\_ctx="datasource"

)

# Filter customers from India

filtered = Filter.apply(frame=datasource, f=lambda x: x["country"] == "India")

# Write result to S3

glueContext.write\_dynamic\_frame.from\_options(

frame=filtered,

connection\_type="s3",

connection\_options={"path": "s3://your-bucket-name/output/"},

format="csv"

)

job.commit()

**STEP 4: Run the Job**

* Go to the job > Click **Run**
* Monitor the logs (CloudWatch)
* Output will appear in s3://your-bucket-name/output/

**✅ You Just Built:**

* 📥 Extract: Read from S3 CSV via Glue Data Catalog
* 🔁 Transform: Filter customers by country
* 📤 Load: Write to S3 in CSV format