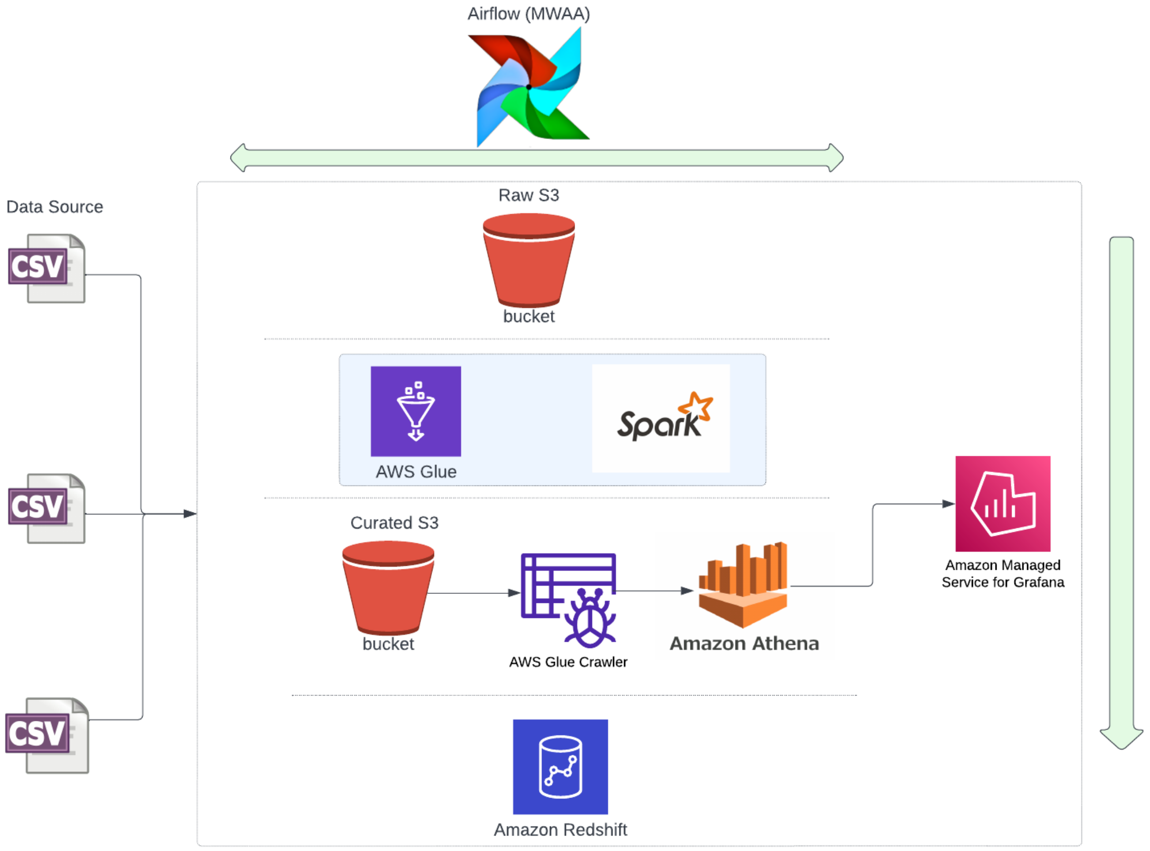
Architecture Design:



The whole architecture is based on AWS technologies primarily ETL Glue, Spark, Amazon Athena, AWS Glue Crawler, and Amazon managed Grafana. If the job needs to be executed periodically it can be achieved by either scheduling the ETL Glue jobs or using MWAA (Managed Airflow in AWS).

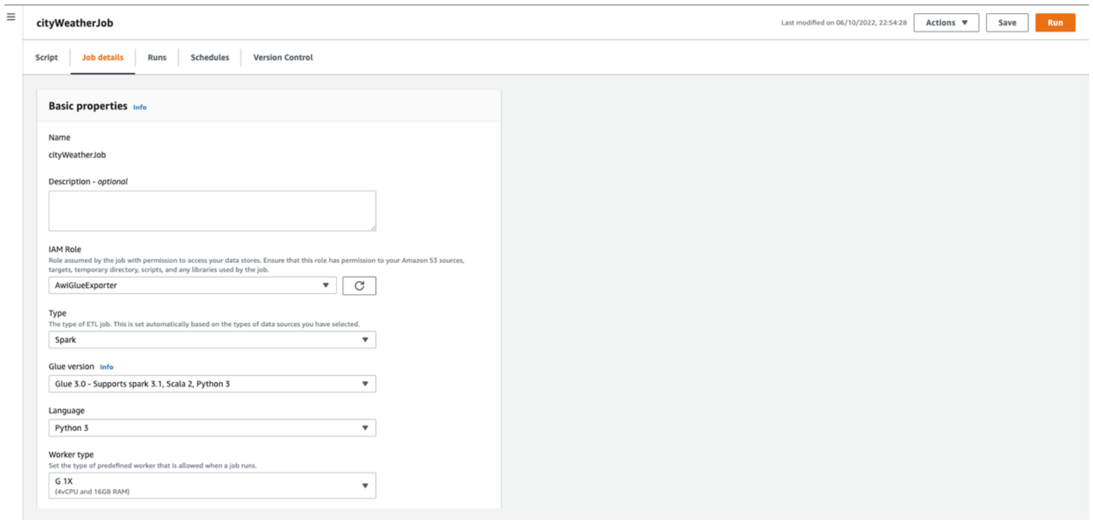
The process is the traditional ETL Pipeline setup where the raw data is read from the CSV files stored in either S3 raw buckets or fetched from the APIs or external services, it is then transformed using Apache Spark and loaded again to S3 curated buckets. From there the data is crawled using AWS Glue Crawler and exported to Glue Catalog tables. It is then exposed to Athena for querying using SQL. Finally, the Athena can be used a source for the Grafana Dashboards.

Steps of this ETL Pipeline:

1. **Identify the Data Sources**
   For this POC (proof of concept) we are using multiple CSV files of weather data.
   The dataset consist of weather data from three different cities in India with attributes like temperature, pressure, humidity, and precipitation etc.

Source: https://www.kaggle.com/datasets/hiteshsoneji/historical-weather-data-for-indian-cities

2. **Create a ETL Glue, and Spark job to transform the data**
   For transforming the huge volume of data, there is a need of distributive architecture like spark which can perform the parallel computation on your data. We are using ETL Glue jobs in AWS to run the Spark script. Below are the steps to create and deploy the Glue jobs
   - Create a ETL Glue job from console (AWS Glue -> Jobs)



Basic properties Info

Name
cityWeatherJob

Description - *optional*

IAM Role
Role assumed by the job with permission to access your data stores. Ensure that this role has permission to your Amazon S3 sources, targets, temporary directory, scripts, and any libraries used by the job.
AwiGlueExporter

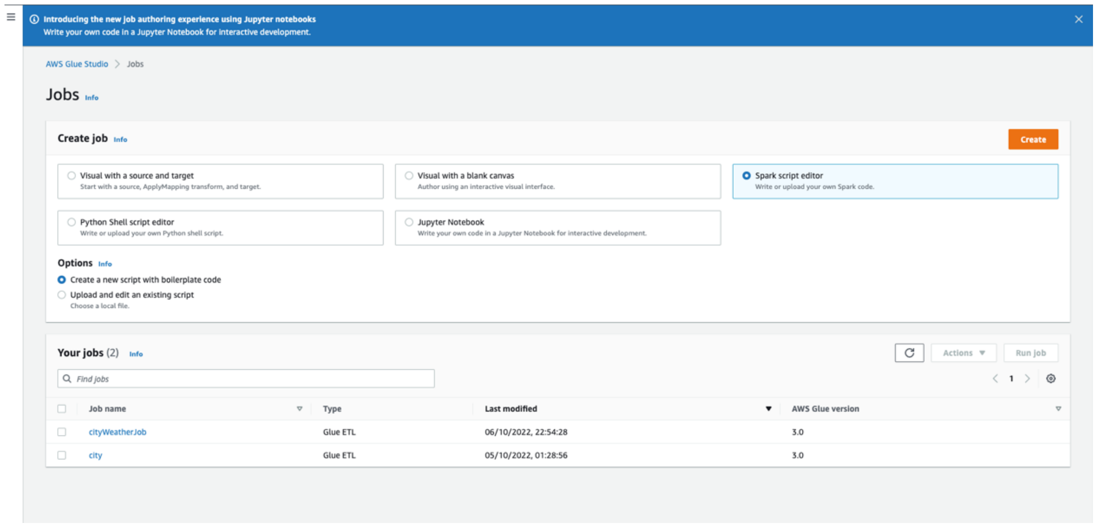
Type
The type of ETL job. This is set automatically based on the types of data sources you have selected.
Spark

Glue version Info
Glue 3.0 - Supports spark 3.1, Scala 2, Python 3

Language
Python 3

Worker type
Set the type of predefined worker that is allowed when a job runs.
G 1X
(4vCPU and 16GB RAM)

Here you can select the Glue version, Spark Version, Python, and Worker Type



Introducing the new job authoring experience using Jupyter notebooks
Write your own code in a Jupyter Notebook for interactive development.

Jobs Info

Create job Info

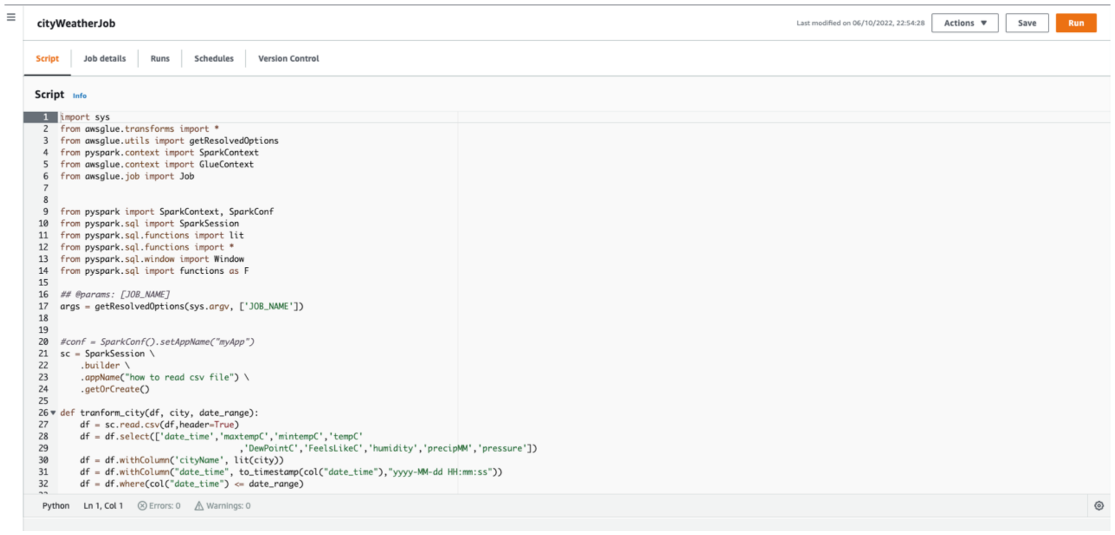
- Visual with a source and target — Start with a source, ApplyMapping transform, and target.
- Visual with a blank canvas — Author using an interactive visual interface.
- [✓ Spark script editor] — Write or upload your own Spark code.
- Python Shell script editor — Write or upload your own Python shell script.
- Jupyter Notebook — Write your own code in a Jupyter Notebook for interactive development.

Options Info
- [✓ Create a new script with boilerplate code]
- Upload and edit an existing script — Choose a local file.

Your jobs (2) Info

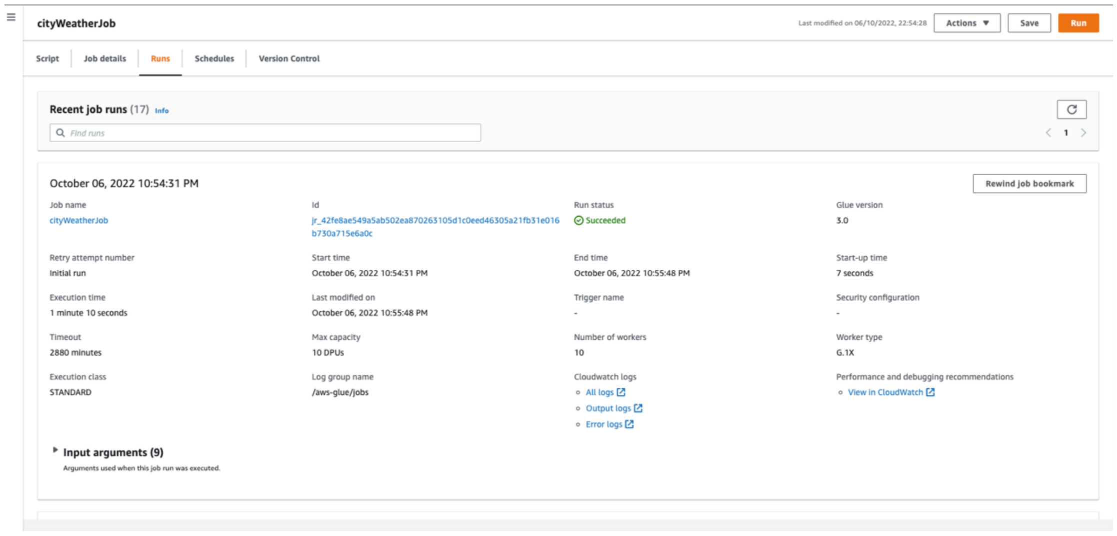
Job name	Type	Last modified	AWS Glue version
cityWeatherJob	Glue ETL	06/10/2022, 22:54:28	3.0
city	Glue ETL	05/10/2022, 01:28:56	3.0

- Add the Spark Script (etl.py) under the script section



```python
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 pyspark import SparkContext, SparkConf
from pyspark.sql import SparkSession
from pyspark.sql.functions import lit
from pyspark.sql.functions import *
from pyspark.sql.window import Window
from pyspark.sql import functions as F

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])


#conf = SparkConf().setAppName("myApp")
sc = SparkSession \
    .builder \
    .appName("how to read csv file") \
    .getOrCreate()

def tranform_city(df, city, date_range):
    df = sc.read.csv(df,header=True)
    df = df.select(['date_time','maxtempC','mintempC','tempC'
                                   ,'DewPointC','FeelsLikeC','humidity','precipMM','pressure'])
    df = df.withColumn('cityName', lit(city))
    df = df.withColumn("date_time", to_timestamp(col("date_time"),"yyyy-MM-dd HH:mm:ss"))
    df = df.where(col("date_time") <= date_range)
```

- Explore the job status from the Runs section



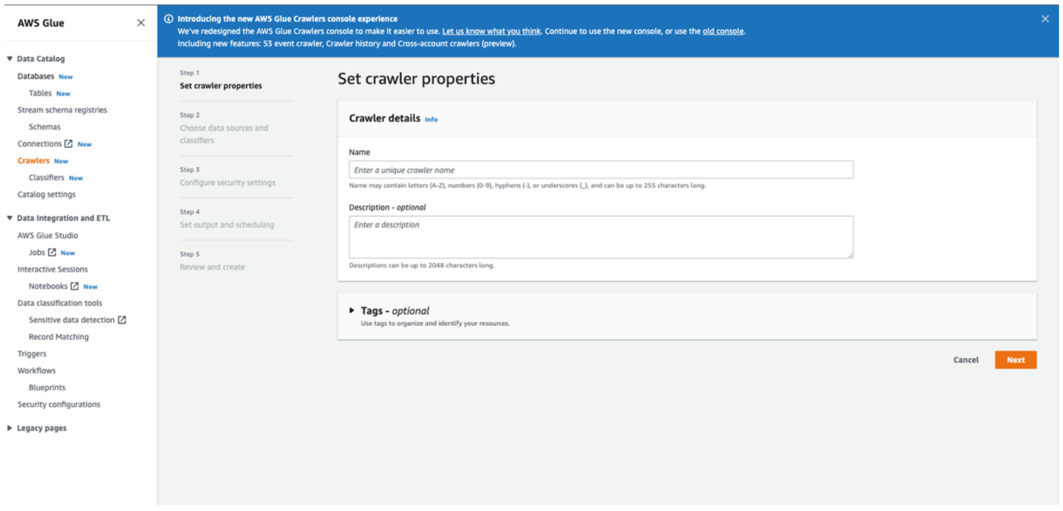
October 06, 2022 10:54:31 PM

Job name cityWeatherJob	Id jr_42fe8ae549a5ab502ea870263105d1c0eed46305a21fb31e016b730a715e6a0c	Run status Succeeded	Glue version 3.0
Retry attempt number Initial run	Start time October 06, 2022 10:54:31 PM	End time October 06, 2022 10:55:48 PM	Start-up time 7 seconds
Execution time 1 minute 10 seconds	Last modified on October 06, 2022 10:55:48 PM	Trigger name -	Security configuration -
Timeout 2880 minutes	Max capacity 10 DPUs	Number of workers 10	Worker type G.1X
Execution class STANDARD	Log group name /aws-glue/jobs	Cloudwatch logs All logs Output logs Error logs	Performance and debugging recommendations View in CloudWatch

Input arguments (9)

Arguments used when this job run was executed.

- Once the job is succeeded, create the Glue crawler to crawl the final data from curated S3 bucket. (AWS Glue -> Crawlers)



Introducing the new AWS Glue Crawlers console experience
We've redesigned the AWS Glue Crawlers console to make it easier to use. Let us know what you think. Continue to use the new console, or use the old console.
Including new features: S3 event crawler, Crawler history and Cross-account crawlers (preview).

Set crawler properties

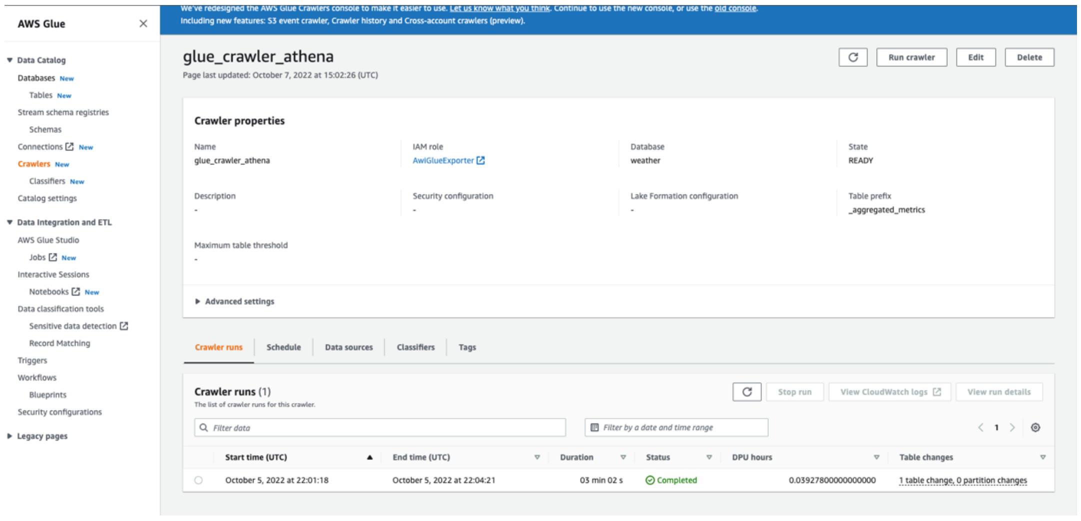
Crawler details Info

Name
Enter a unique crawler name
Name may contain letters (A-Z), numbers (0-9), hyphens (-), or underscores (_), and can be up to 255 characters long.

Description - *optional*
Enter a description
Descriptions can be up to 2048 characters long.

Tags - *optional*
Use tags to organize and identify your resources.

- Create a Database (weather) and Table (aggreagated_metricsoutput) in Glue Catalog.
- Run the Glue Crawler job to export data to Glue Tables (AWS



We've redesigned the AWS Glue Crawlers console to make it easier to use. Let us know what you think. Continue to use the new console, or use the old console.
Including new features: S3 event crawler, Crawler history and Cross-account crawlers (preview).

glue_crawler_athena
Page last updated: October 7, 2022 at 15:02:26 (UTC)

Crawler properties

Name	IAM role	Database	State
glue_crawler_athena	AwiGlueExporter	weather	READY

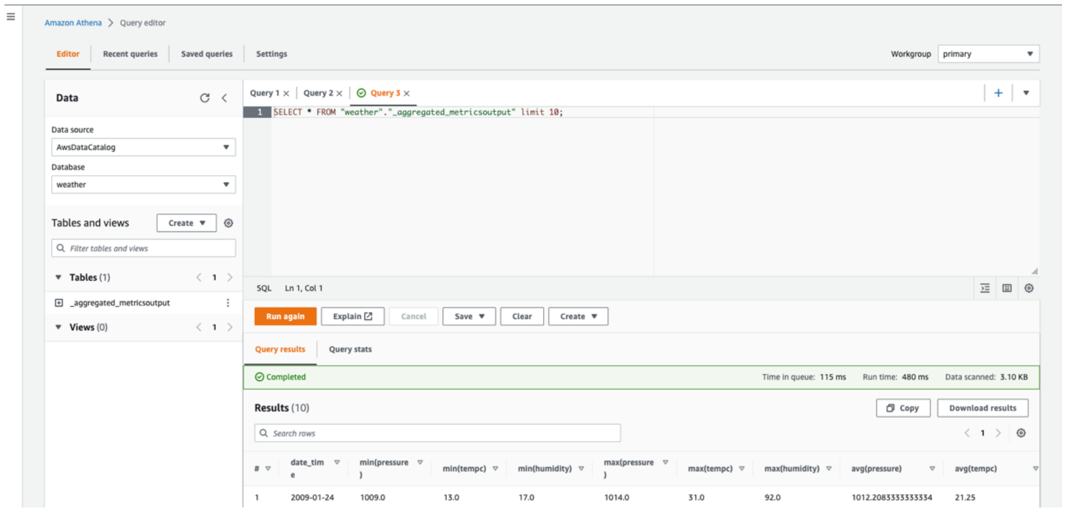
Description	Security configuration	Lake Formation configuration	Table prefix
-	-	-	_aggregated_metrics

Maximum table threshold
-

Crawler runs (1)
The list of crawler runs for this crawler.

	Start time (UTC)	End time (UTC)	Duration	Status	DPU hours	Table changes
○	October 5, 2022 at 22:01:18	October 5, 2022 at 22:04:21	03 min 02 s	Completed	0.03927800000000000	1 table change, 0 partition changes

- Expose the Glue Tables to AWS Athena and Query using SQL



Workgroup: primary

Data source: AwsDataCatalog

Database: weather

_aggregated_metricsoutput

```
SELECT * FROM "weather"."_aggregated_metricsoutput" limit 10;
```

Completed — Time in queue: 115 ms — Run time: 480 ms — Data scanned: 3.10 KB

Results (10)

#	date_time	min(pressure)	min(tempc)	min(humidity)	max(pressure)	max(tempc)	max(humidity)	avg(pressure)	avg(tempc)
1	2009-01-24	1009.0	13.0	17.0	1014.0	31.0	92.0	1012.2083333333334	21.25

3. Use Athena as a Data Source for Grafana/Amazon Quicksight (dashboard attached as pdf separately)

Note: This can also be achieved programmatically using Airflow, but it is out of scope for this project as we are dealing with one time static CSV files.