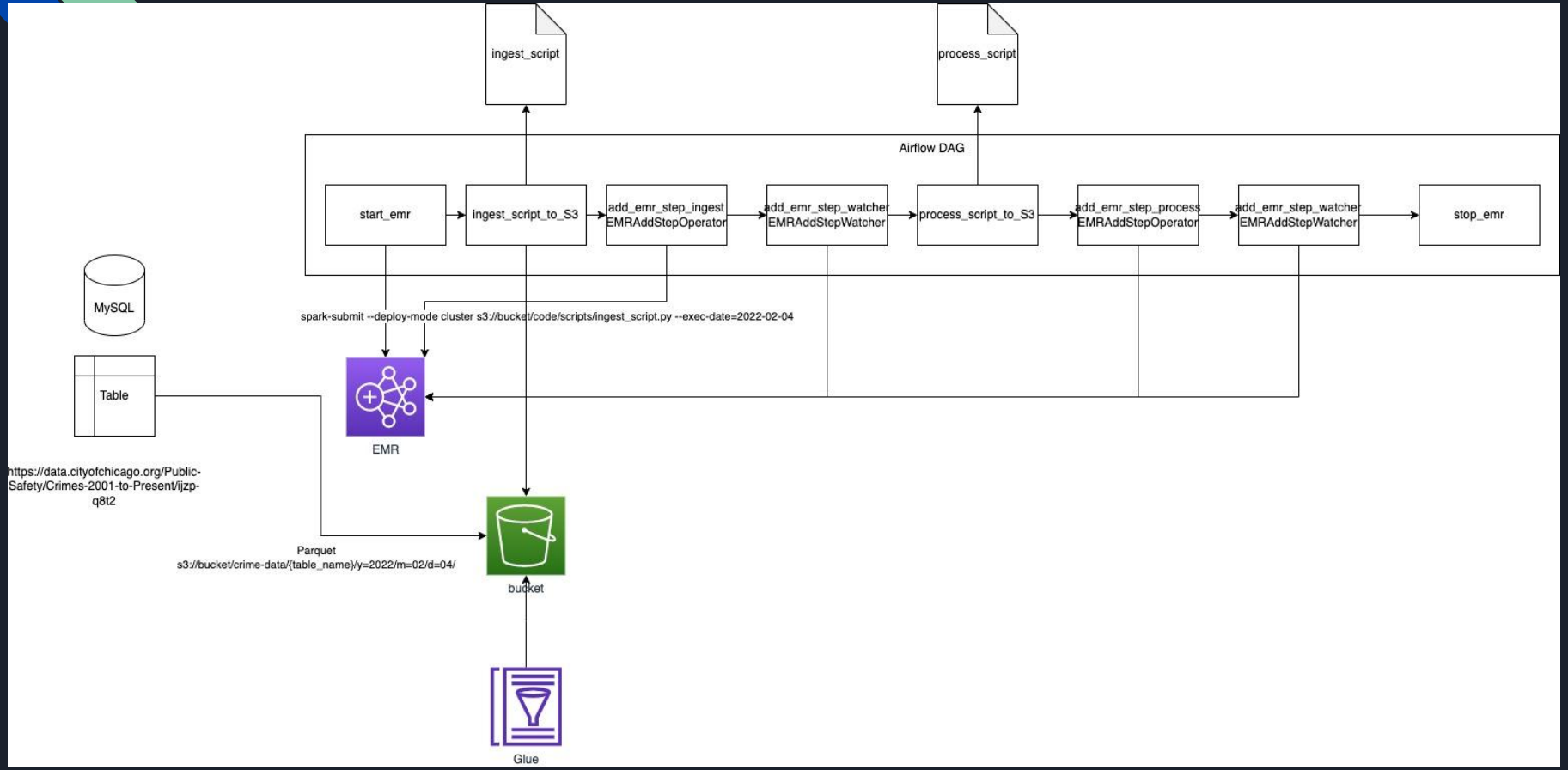
Crime data- Batch Processing

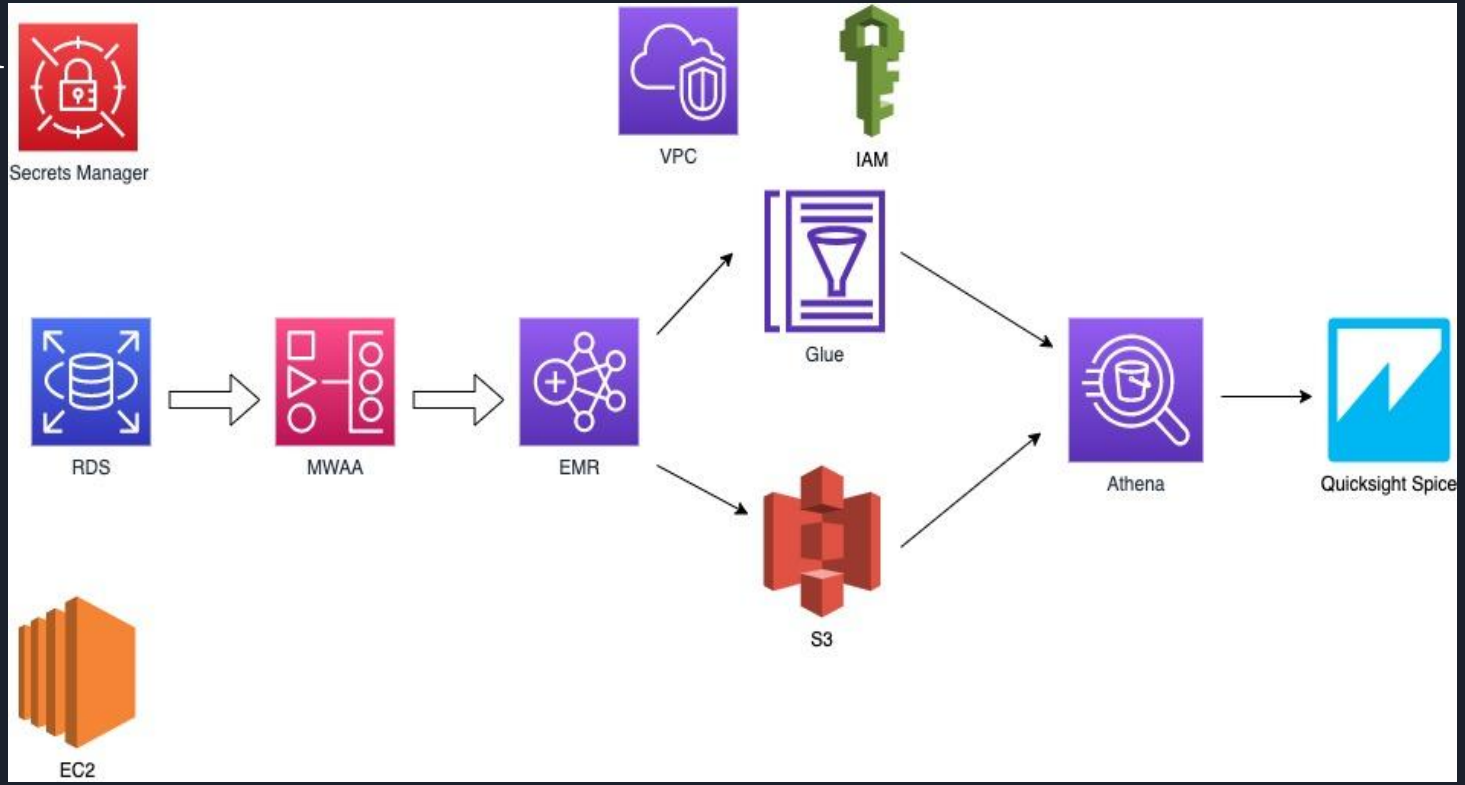
Introduction

- This project is intended to implement a solution designed to extract, transform and load Chicago crime data from an RDS instance to other services in AWS.
- The data is taken from https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-Present/ijzp-q8t2 which has the incidents of crime that happened in City of Chicago from 2001 to Present
- I have chosen this model to perform a crime type analysis where arrests have still not been taken place as of the latest batch extraction.

Architecture Diagram



AWS Services Used: Flow Diagram



Data Collection

An AWS RDS MySQL instance is created to store the batch of data.

- An EC2 instance is created to communicate with the RDS instance.

- The data is loaded onto the EC2 instance.

- The database and table are created on the RDS instance with the help of the above created EC2 instance. The data is loaded in the mentioned.

- A secret on the Secrets Manager console is stored to communicate with the RDS instance secretly. Also, password rotation after 30 days has been configured for security purposes.

Data Collection

Used:

- Amazon EMR / Spark
- Managed Workflows for Apache Airflow

- MWAA environment is used to orchestrate the ETL process using Apache airflow on AWS.

- The airflow dag triggers the EMR which in turn runs the spark applications and loads the data to AWS Glue and s3 to be analyzed and visualized later in other services like Athena and Quicksight.

- Inside the dag, the step checkers are also added to check if that last step is completed or skipped or terminated.

Data Storage and Management

Used

- Build a Data Catalog
- Trace Data Lineage

- The raw data which is taken, is pointed to a table in a database in AWS Glue while also simultaneously creating partitioned data based on the year, month and day of the execution timestamp of the pyspark scripts.

- After performing the required processing, the data is again partitioned by adding a new timestamp column and stored into s3 and Glue.

- This process provides us a lineage where we can track all the versions of the data and go back to any required snapshot of data based on the partition value.

Data Processing

Used

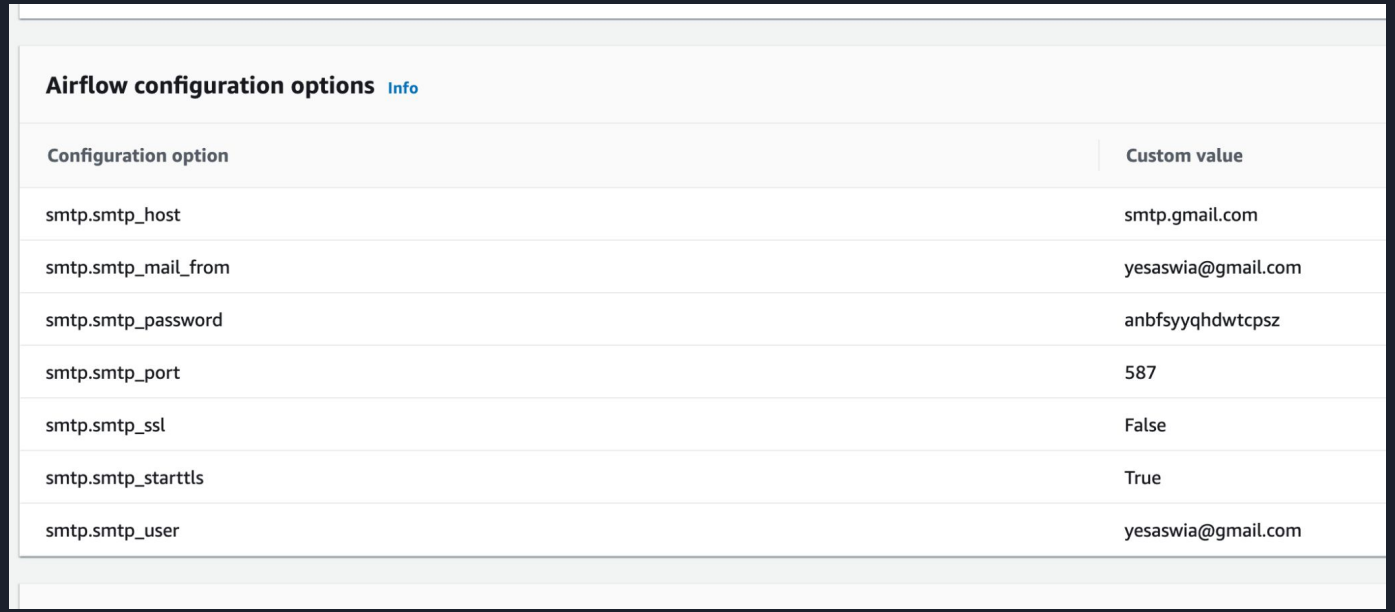
- Able to process data at scale
- Able to process the data within the defined SLA

- The instances that are created for the EMR processing are spot instances that would be able to process the data at scale.

- The process is expected to be finished in 32-34 mins in general. Therefore SLA monitoring has been placed to get an email alert when the process takes longer than that so that we could know if something has interrupted the process and making it take more time than expected.

Data Processing

For the sake of SLA monitoring and to send email regarding SLA misses, configured the airflow as follows:



Airflow configuration options Info

Configuration option	Custom value
smtp.smtp_host	smtp.gmail.com
smtp.smtp_mail_from	yesaswia@gmail.com
smtp.smtp_password	anbfsyyqhdwtcpsz
smtp.smtp_port	587
smtp.smtp_ssl	False
smtp.smtp_starttls	True
smtp.smtp_user	yesaswia@gmail.com

Data Analysis and Visualization

Used

- Able to generate dashboards and refresh the data on demand
- Creating appropriate pre-aggregations and views to reduce cost and latency
- Appropriate analysis and visualization groupings based on the use case

- The data can be analyzed in Athena to query the raw data or even the processed data.

-The processed table to sent to Quicksight Spice to provide a good visualization and comparison of the crime type analysis. In Quicksight, the data is refreshed automatically when we load any new batch into our RDS.

-Also, since we are using the processed table for our final visualization, it has only the features required for this use case analysis i.e, The Crime Type analysis grouped by the type of crime and count of it. Also only not arrested data is taken for analysis. So, it reduces the extra cost and latency.Using SPICE reduces latency too.

Securing Data Analysis Systems

- Implemented MFA at the account level
- Implement password rotation for the RDS secret using AWS Secrets Manager
- Hiding the RDS database credentials in the code in the pyspark Application. Used the secret values in place of the actual credentials.
- In MWAA, used the 16 digit App password code instead of the gmail password for the smtp password

Thank you!