Forest Fire Predictive Analytics with AWS Cloud

Team 9: Anna Chow, Hemang Behl, Jason Gonsalves, Kevin Chuang, Richita Das

# What is the project idea /What the application does?

Wildfires can cause devastating destruction and cost millions of dollars in damage, especially with increasing human development near wilderness or rural areas. The ability to predict wildfires would be a preemptive measure to prevent and/or manage wildfires. Factors affecting wildfires would be explored and used to gauge the probability of a wildfire occurring. Based on the location and the corresponding land type, a machine learning model will be trained to identify the major contributing factors of the fire. The whole process will be carried out in cloud where the model can directly consume the source file.

# What are the technologies used?

* + Python + libraries (numpy, sklearn, scipy, matplotlib, Flask, boto3, sagemaker) + IDE (Jupyter Notebook/Spyder)
  + SQL
  + AWS S3
  + AWS Glacier
  + AWS Glue
  + AWS Athena
  + AWS Redshift
  + AWS QuickSight
  + AWS SageMaker
  + AWS CloudWatch Logs

# Feature list of the dataset:

* + FOD\_ID = Global unique identifier.
  + FPA\_ID = Unique identifier that contains information necessary to track back to the original record in the source dataset.
  + SOURCE\_SYSTEM\_TYPE = Type of source database or system that the record was drawn from (federal, nonfederal, or interagency).
  + SOURCE\_SYSTEM = Name of or other identifier for source database or system that the record was drawn from.
  + NWCG\_REPORTING\_AGENCY = Active National Wildlife Coordinating Group (NWCG) Unit Identifier for the agency preparing the fire report
  + NWCG\_REPORTING\_UNIT\_ID = Active NWCG Unit Identifier for the unit preparing the fire report.
  + NWCG\_REPORTING\_UNIT\_NAME = Active NWCG Unit Name for the unit preparing the fire report.
  + SOURCE\_REPORTING\_UNIT = Code for the agency unit preparing the fire report, based on code/name in the source dataset.
  + SOURCE\_REPORTING\_UNIT\_NAME = Name of reporting agency unit preparing the fire report, based on code/name in the source dataset.
  + LOCAL\_FIRE\_REPORT\_ID = Number or code that uniquely identifies an incident report for a particular reporting unit and a particular calendar year.
  + LOCAL\_INCIDENT\_ID = Number or code that uniquely identifies an incident for a particular local fire management organization within a particular calendar year.
  + FIRE\_CODE = Code used within the interagency wildland fire community to track and compile cost information for emergency fire suppression
  + FIRE\_NAME = Name of the incident, from the fire report (primary) or ICS-209 report (secondary).
  + ICS\_209\_INCIDENT\_NUMBER = Incident (event) identifier, from the ICS-209 report.
  + ICS\_209\_NAME = Name of the incident, from the ICS-209 report.
  + MTBS\_ID = Incident identifier, from the MTBS perimeter dataset.
  + MTBS\_FIRE\_NAME = Name of the incident, from the MTBS perimeter dataset.
  + COMPLEX\_NAME = Name of the complex under which the fire was ultimately managed, when discernible.
  + FIRE\_YEAR = Calendar year in which the fire was discovered or confirmed to exist.
  + DISCOVERY\_DATE = Date on which the fire was discovered or confirmed to exist.
  + DISCOVERY\_DOY = Day of year on which the fire was discovered or confirmed to exist.
  + DISCOVERY\_TIME = Time of day that the fire was discovered or confirmed to exist.
  + STAT\_CAUSE\_CODE = Code for the (statistical) cause of the fire.
  + STAT\_CAUSE\_DESCR = Description of the (statistical) cause of the fire.
  + CONT\_DATE = Date on which the fire was declared contained or otherwise controlled (mm/dd/yyyy)
  + CONT\_DOY = Day of year on which the fire was declared contained or otherwise controlled.
  + CONT\_TIME = Time of day that the fire was declared contained or otherwise controlled (hhmm where hh=hour, mm=minutes).
  + FIRE\_SIZE = Estimate of acres within the final perimeter of the fire.
  + FIRE\_SIZE\_CLASS = Code for fire size based on the number of acres within the final fire perimeter expenditures (A=greater than 0 but less than or equal to 0.25 acres,

B=0.26-9.9 acres, C=10.0-99.9 acres, D=100-299 acres, E=300 to 999 acres, F=1000 to 4999 acres, and G=5000+ acres).

* + LATITUDE = Latitude (NAD83) for point location of the fire (decimal degrees).
  + LONGITUDE = Longitude (NAD83) for point location of the fire (decimal degrees).
  + OWNER\_CODE = Code for primary owner or entity responsible for managing the land at the point of origin of the fire at the time of the incident.
  + OWNER\_DESCR = Name of primary owner or entity responsible for managing the land at the point of origin of the fire at the time of the incident.
  + STATE = Two-letter alphabetic code for the state in which the fire burned (or originated), based on the nominal designation in the fire report.
  + COUNTY = County, or equivalent, in which the fire burned (or originated), based on nominal designation in the fire report.
  + FIPS\_CODE = Three-digit code from the Federal Information Process Standards (FIPS) publication 6-4 for representation of counties and equivalent entities.
  + FIPS\_NAME = County name from the FIPS publication 6-4 for representation of counties and equivalent entities.

**Features post feature engineering:**

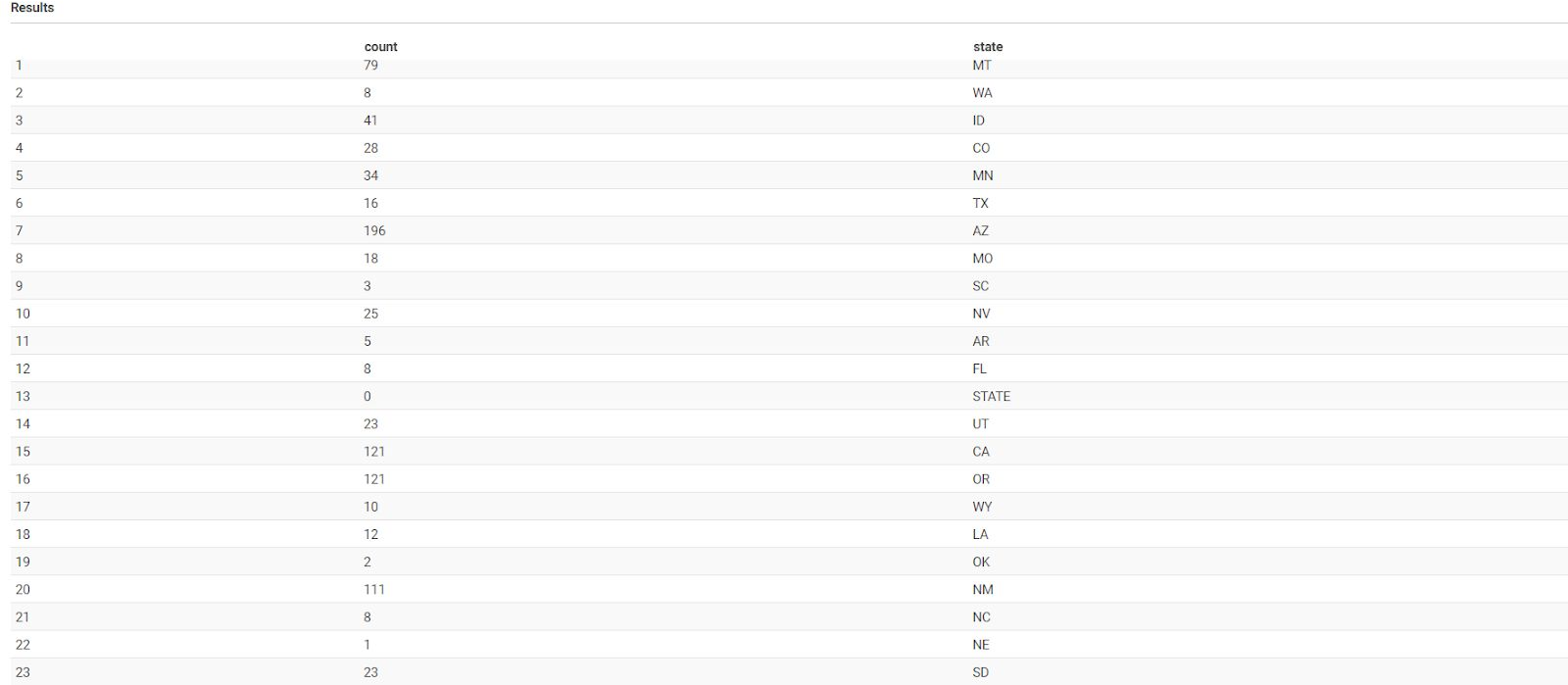
* + - FIRE\_YEAR (long)
    - STAT\_CAUSE\_DESCR (string)
    - FIRE\_SIZE (double)
    - FIRE\_SIZE\_CLASS (string)
    - LATITUDE (double)
    - LONGITUDE (double)
    - STATE (string)
    - COUNTY (long)
    - DISCOVERY\_DATE (double)
    - CNT\_DATE (double)

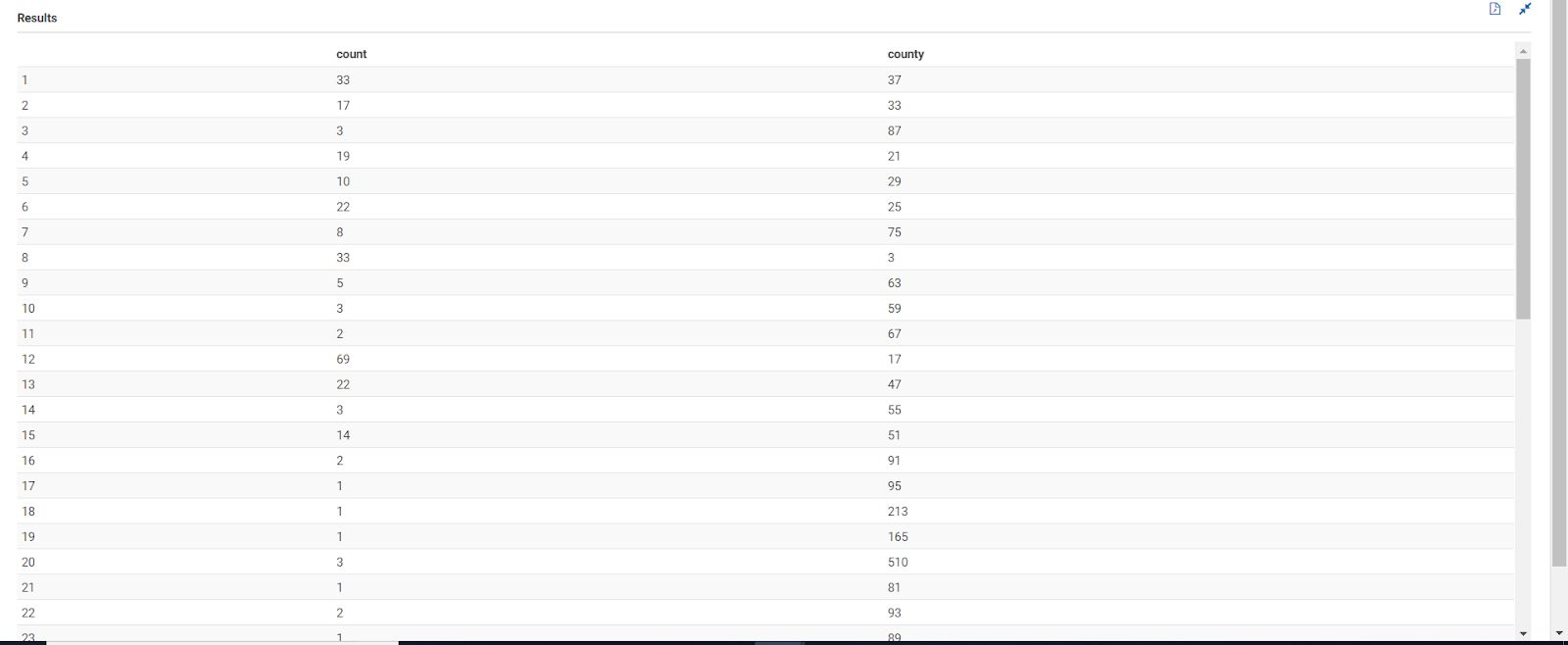
# Sample Demo Screenshots

## AWS Athena

SQL Queries

SELECT count(fod\_id) as count, state FROM sampledb.test\_266 GROUP BY state;



SELECT count(fod\_id) as count, county FROM sampledb.test\_266 GROUP BY county;

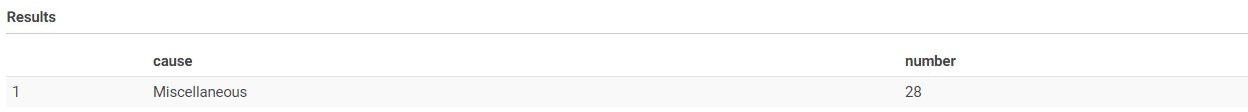
select max(count) as c, state from

(SELECT state,count(fod\_id) as count FROM sampledb.test\_266 GROUP BY state order by count desc) group by state order by c desc limit 1 ;

select \* from

(SELECT state,count(fod\_id) as count FROM sampledb.test\_266 GROUP BY state order by count limit 2) where state <> 'STATE';

select max(stat\_cause\_descr) as cause,count(stat\_cause\_descr) as number from sampledb.test\_266 where state='CO' ;



select max(stat\_cause\_descr) as cause,count(stat\_cause\_descr) as number from sampledb.test\_266 where state='AZ' ;



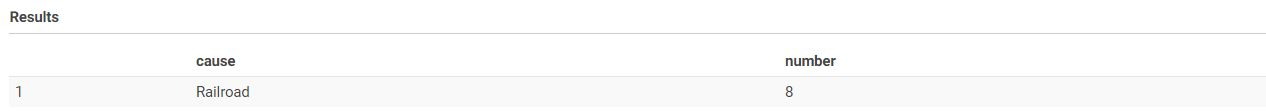
select max(stat\_cause\_descr) as cause,count(stat\_cause\_descr) as number from sampledb.test\_266 where state='MT' ;



select max(stat\_cause\_descr) as cause,count(stat\_cause\_descr) as number from sampledb.test\_266 where state='UT' ;



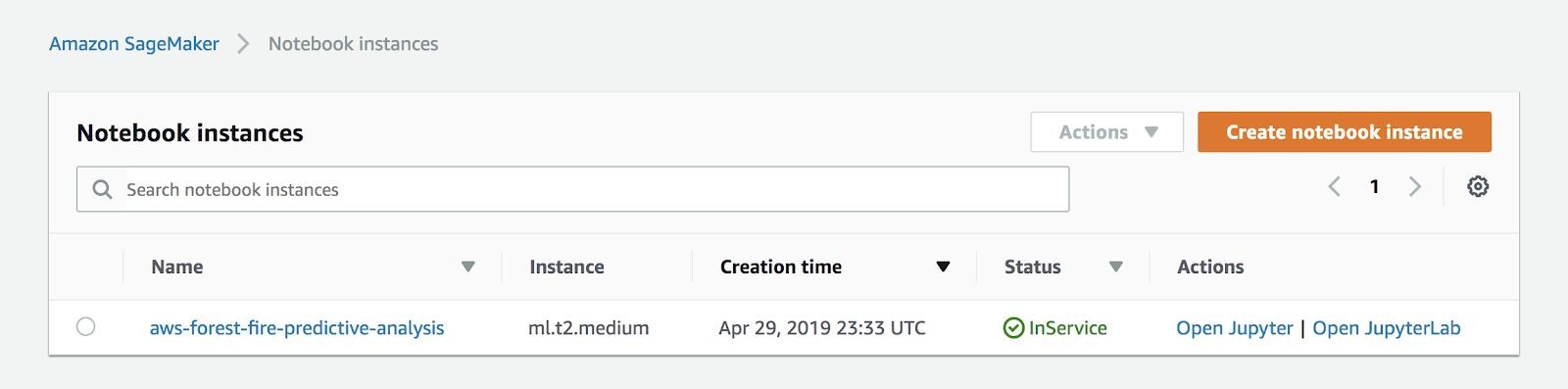
select max(stat\_cause\_descr) as cause,count(stat\_cause\_descr) as number from sampledb.test\_266 where state='WA' ;

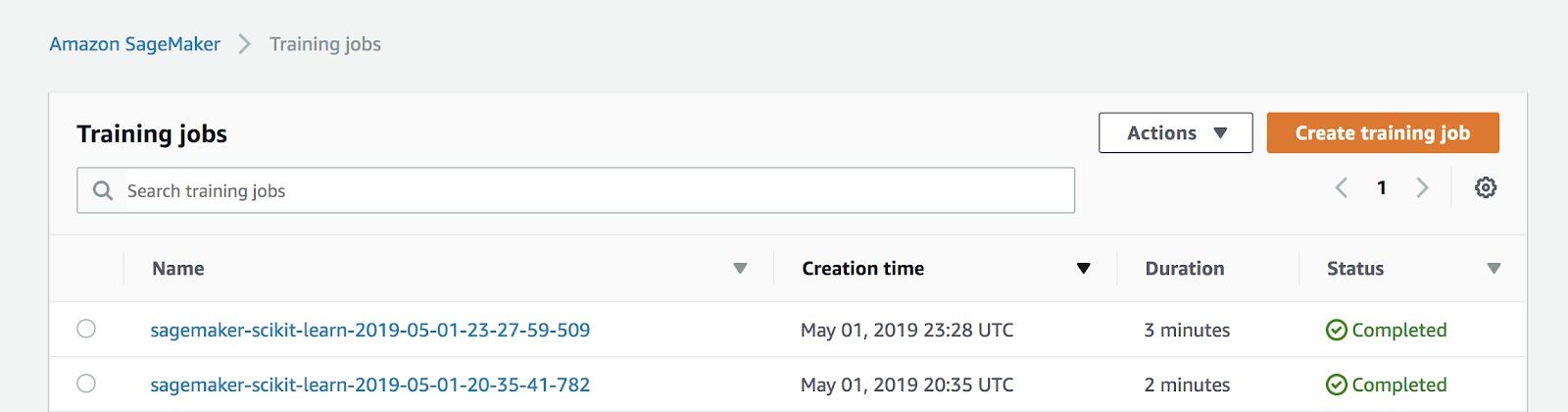


select max(stat\_cause\_descr) as cause,count(stat\_cause\_descr) as number from sampledb.test\_266 where state='OR' ;

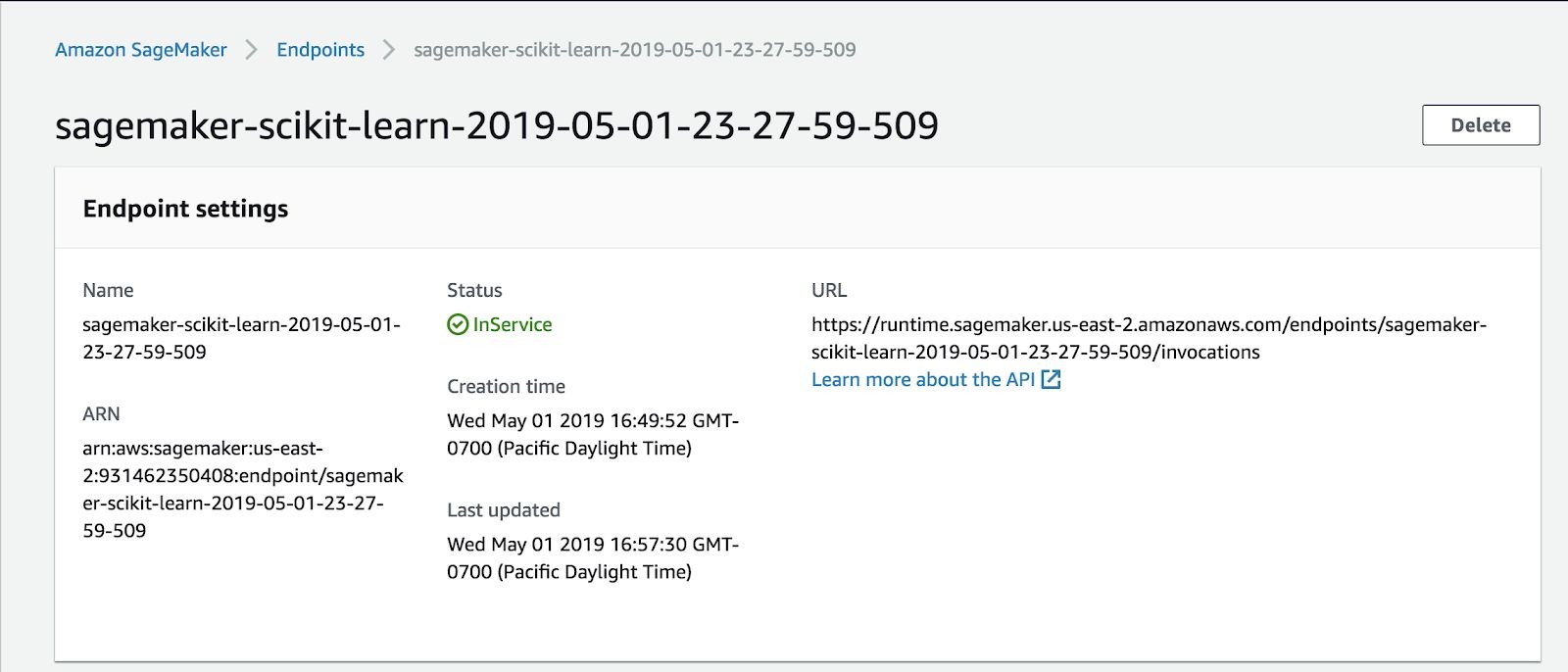


## AWS SageMaker

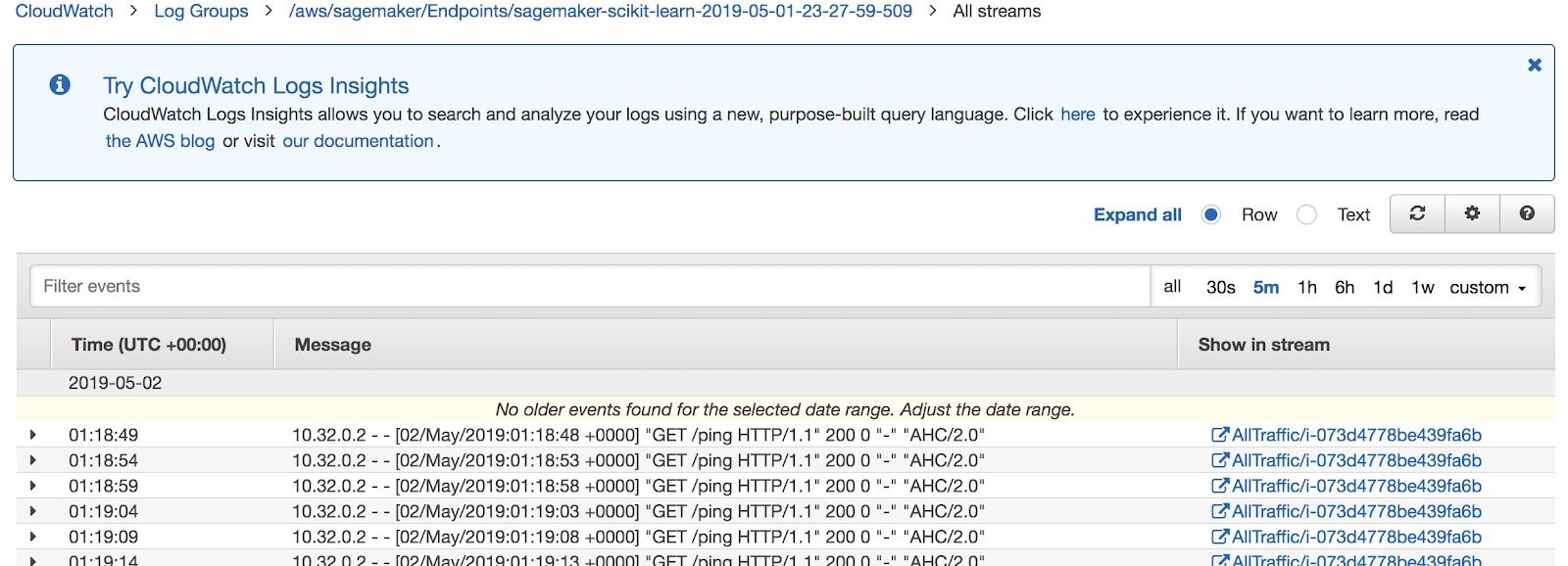
**Notebook Instances**

**Training Jobs**

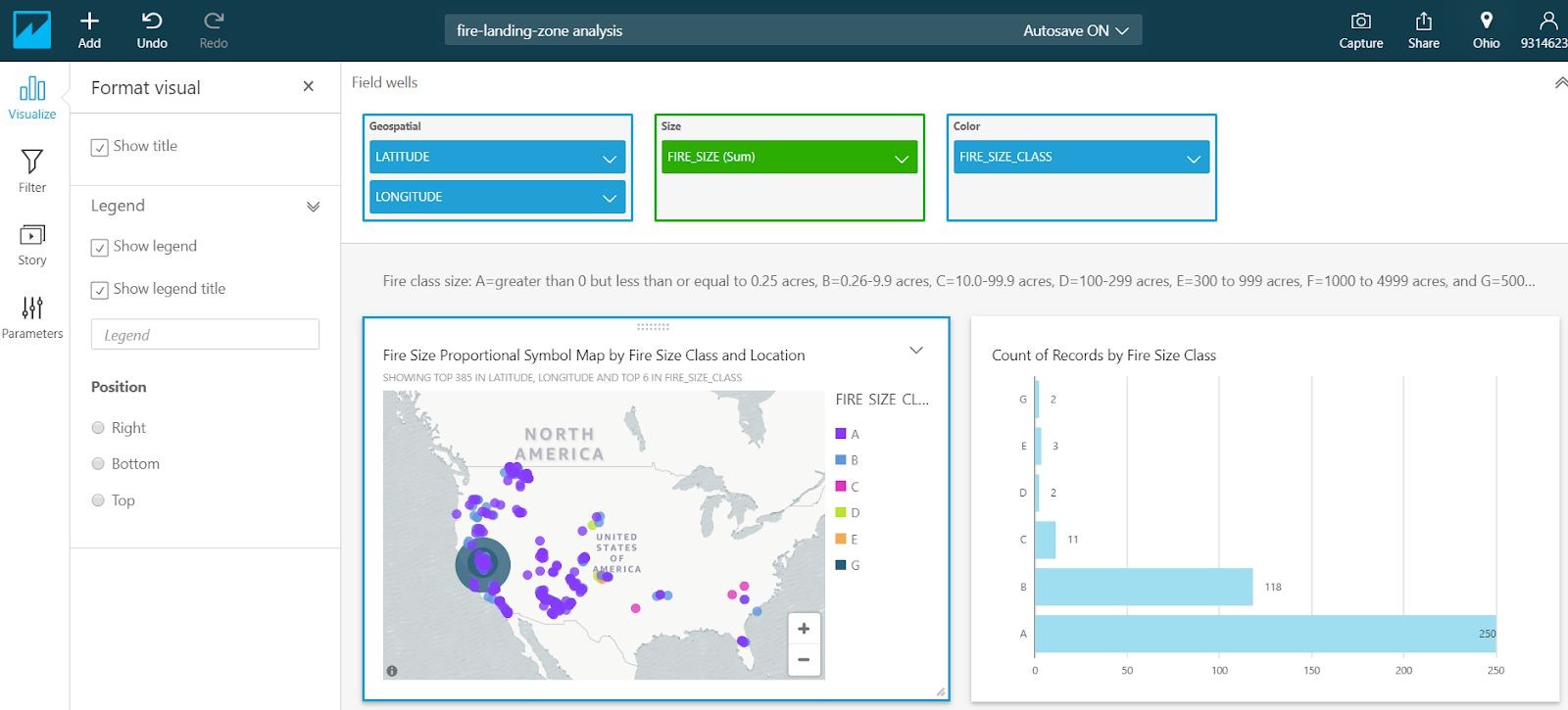
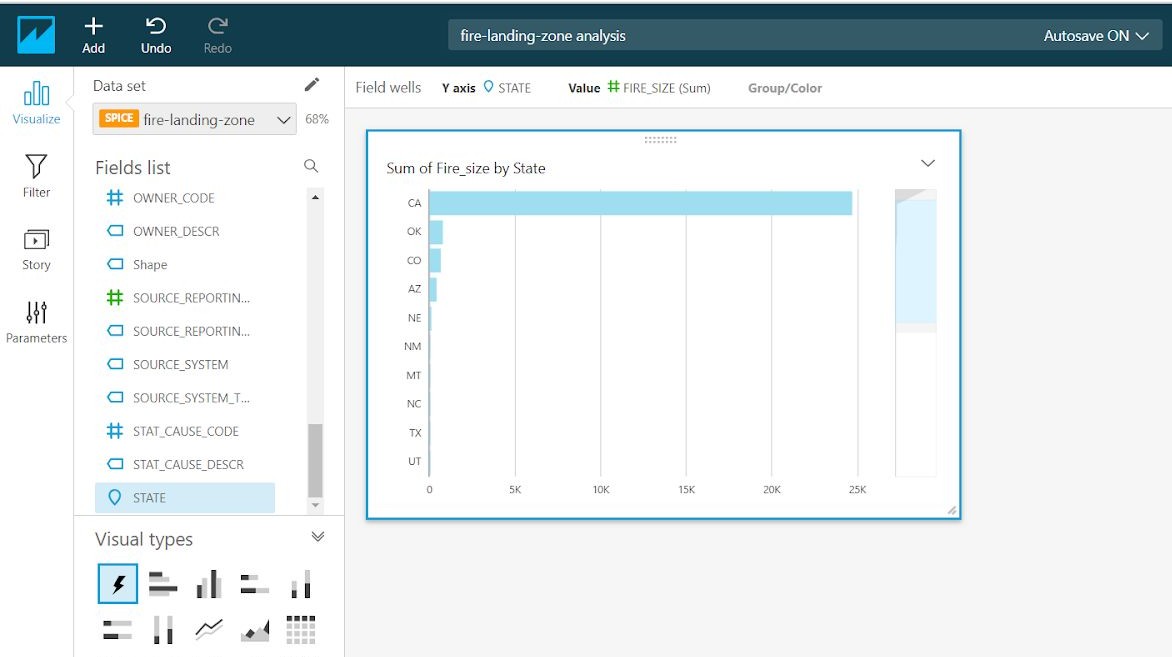
**SageMaker Endpoint (Deployed Model)**

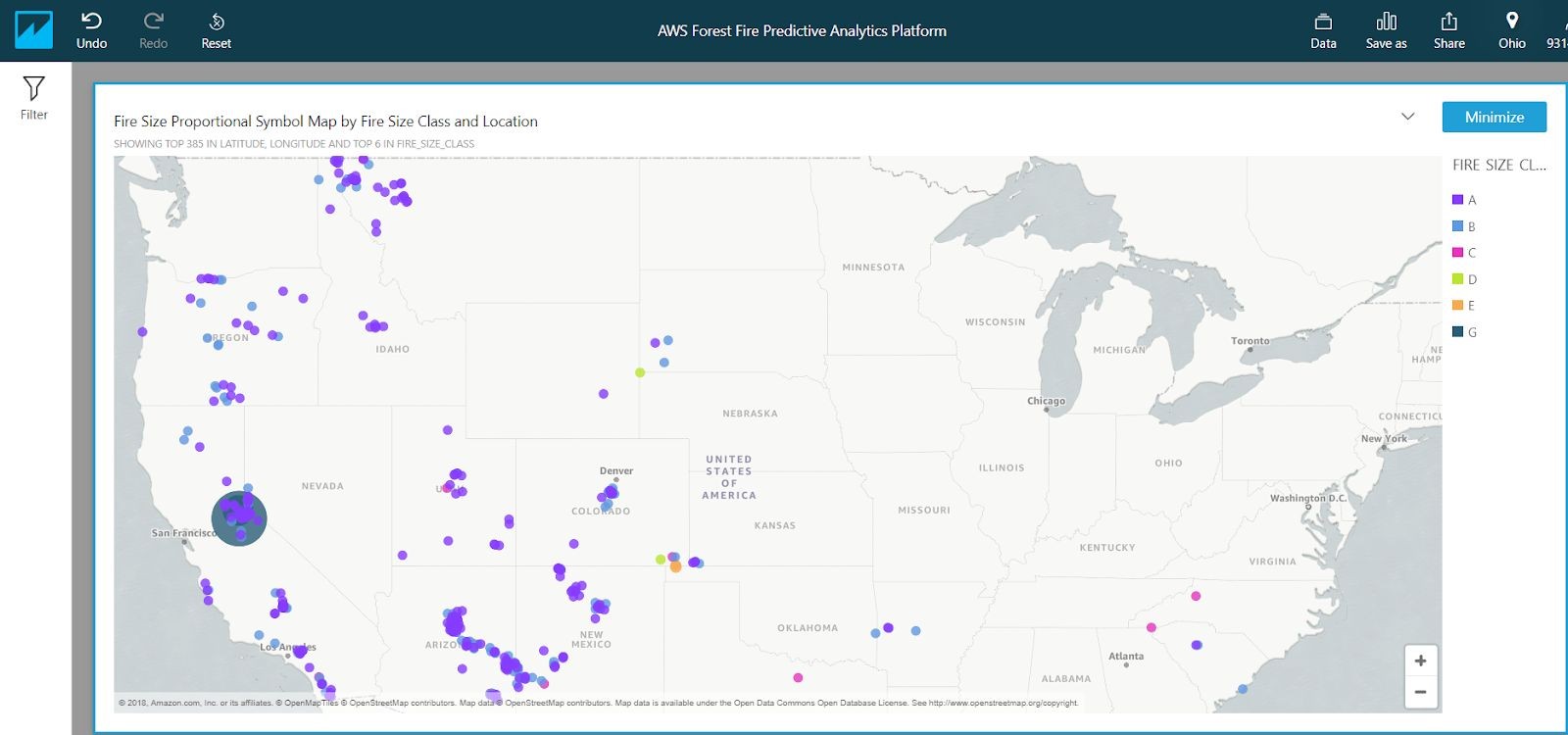
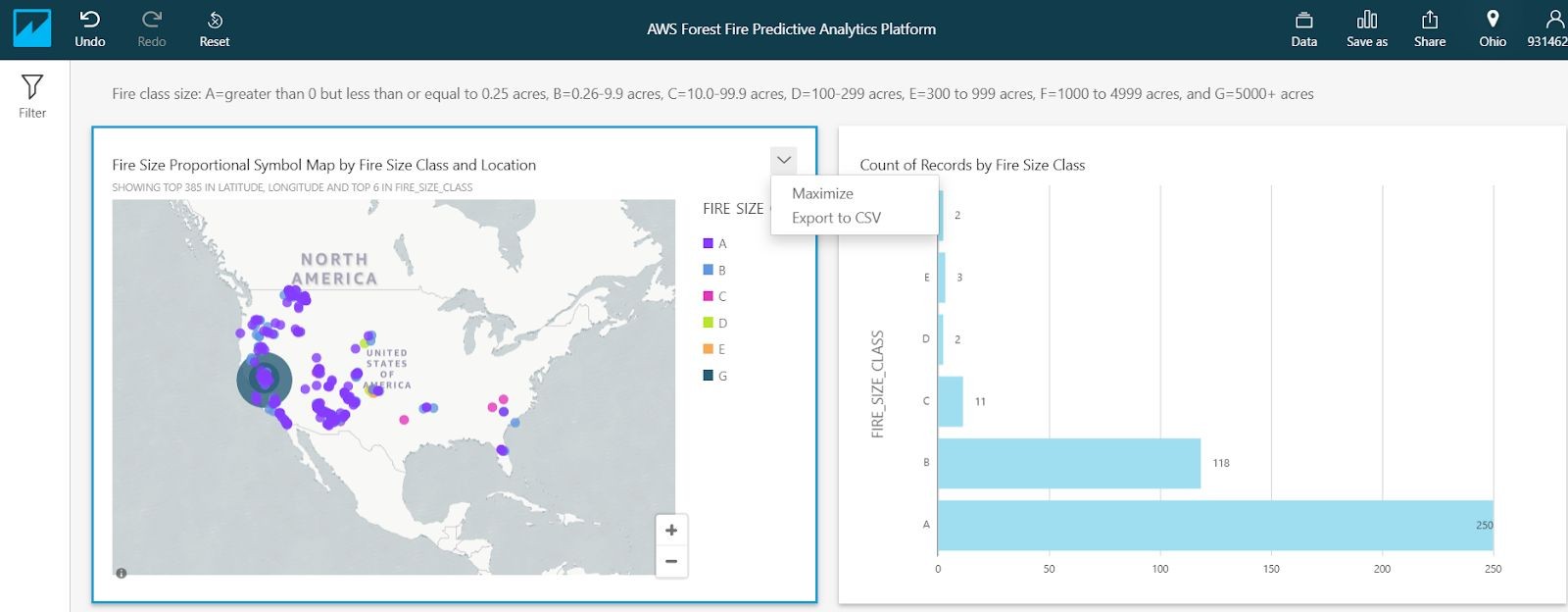
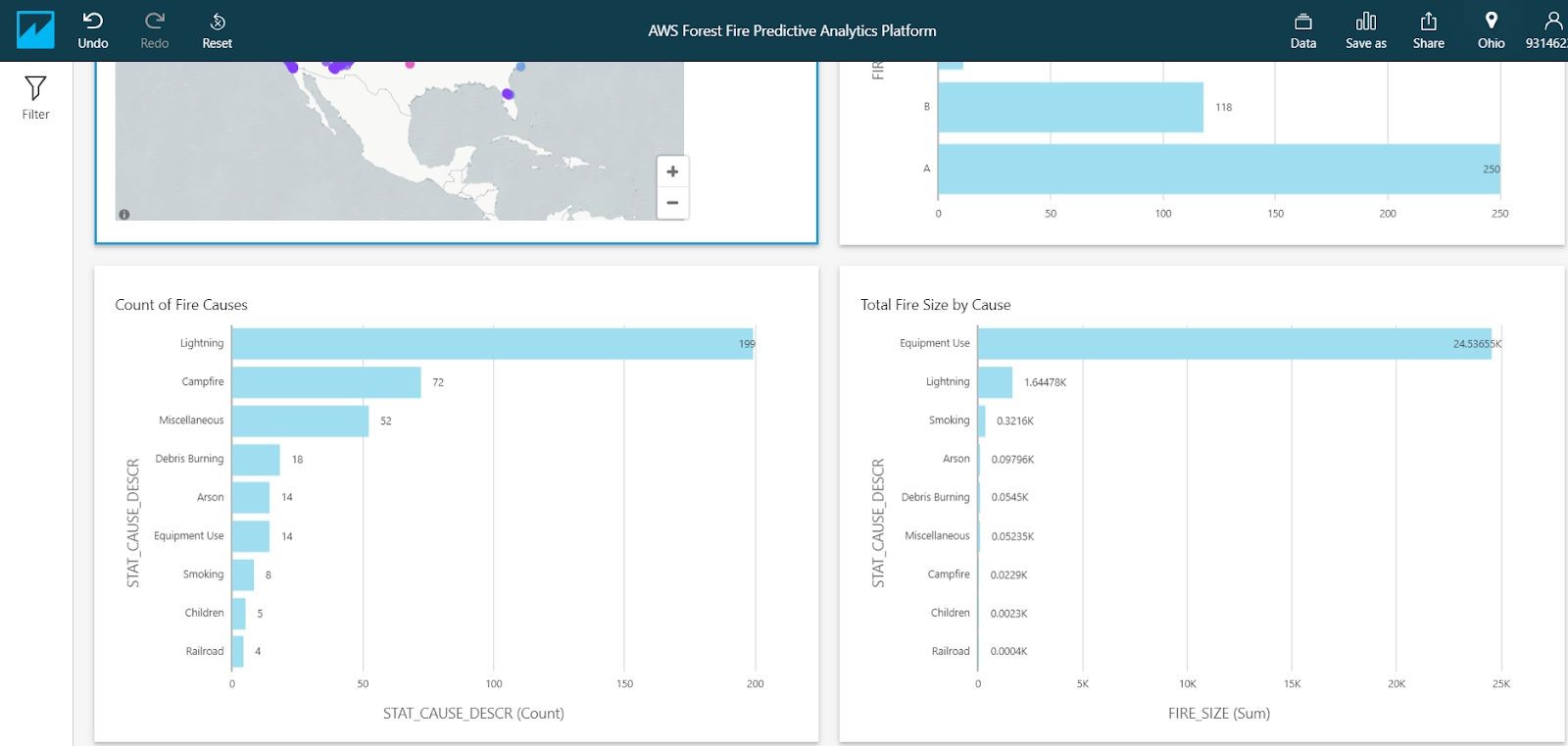


**CloudWatch Logs**

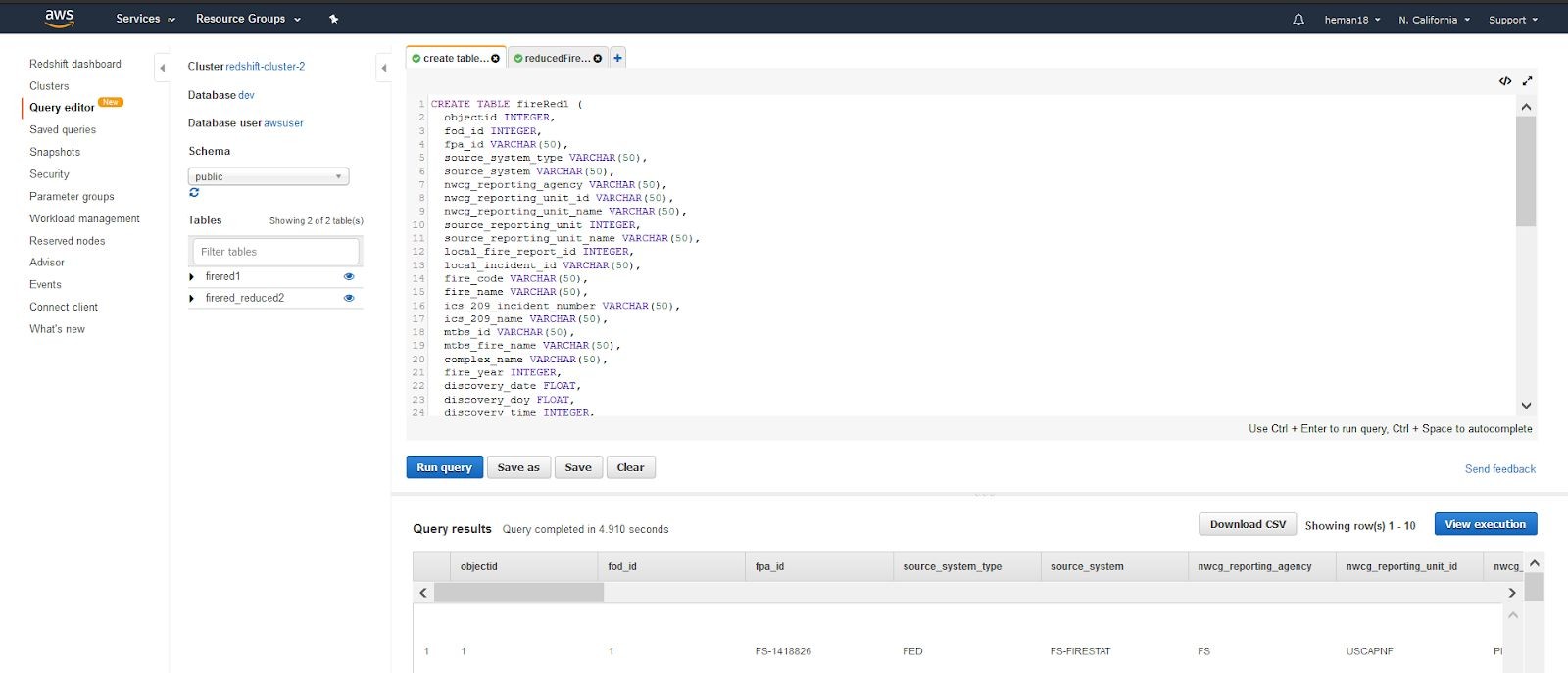


**QuickSight Analysis**

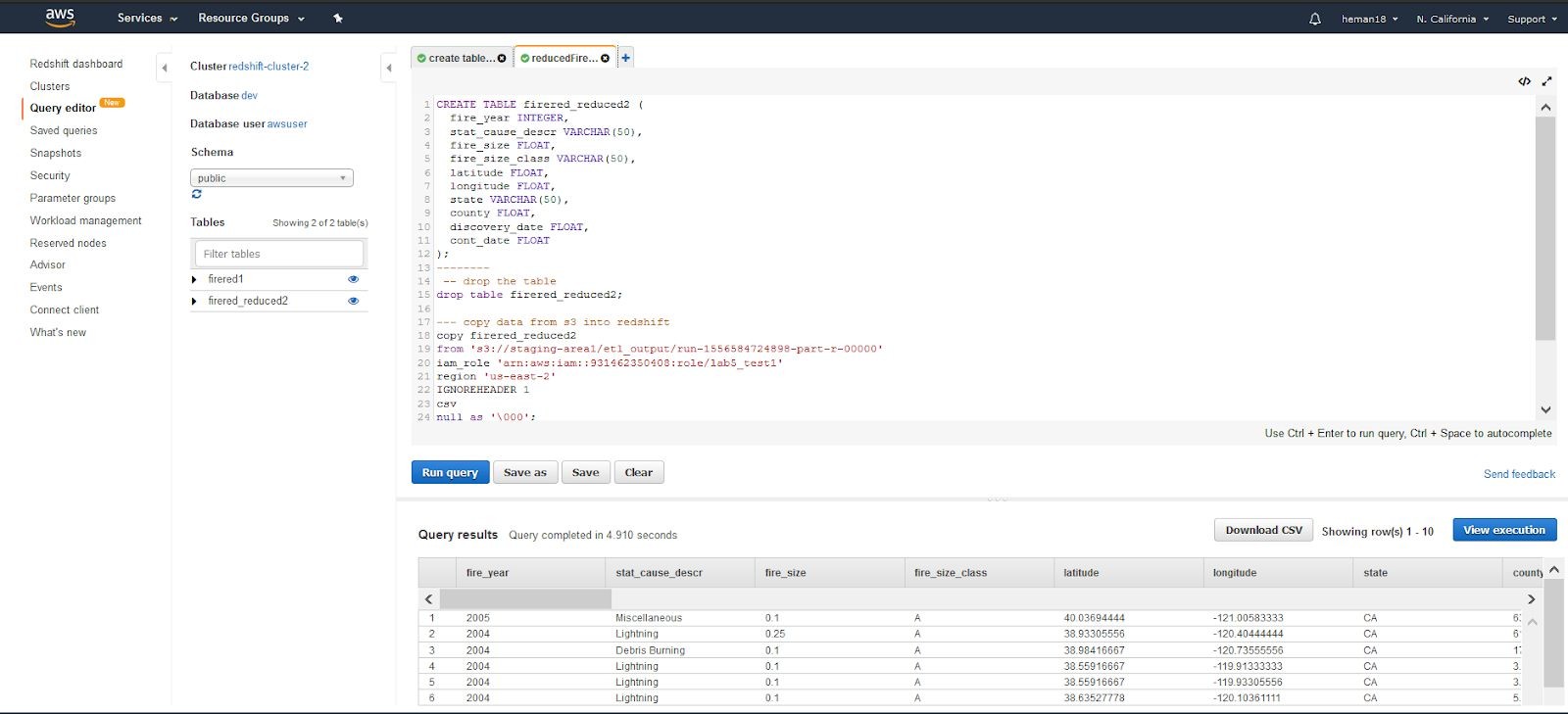
**Dashboard Setup**

**QuickSight Dashboard Overview**

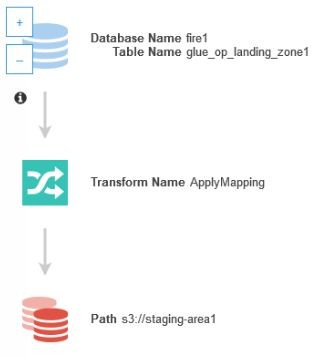
**RedShift Demo**

Querying data loaded from S3 (raw data)

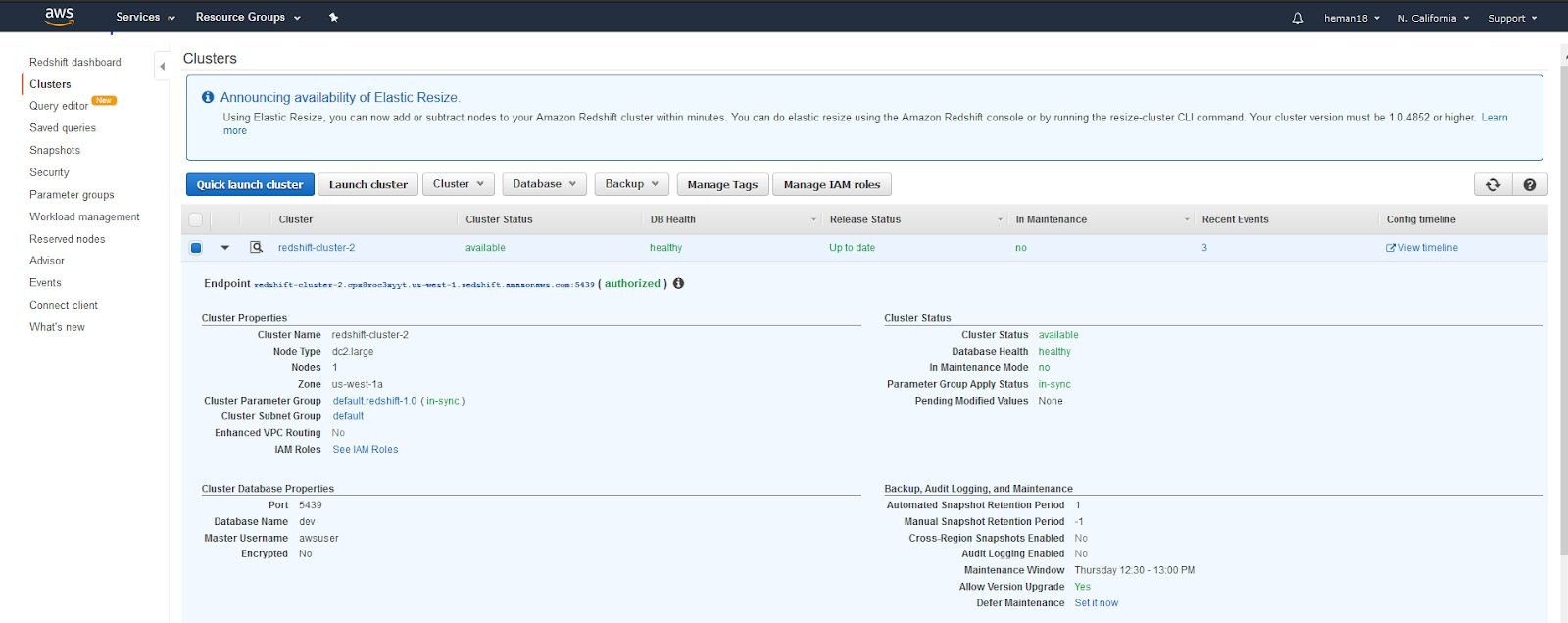
Querying data loaded from S3 (reduced data set)



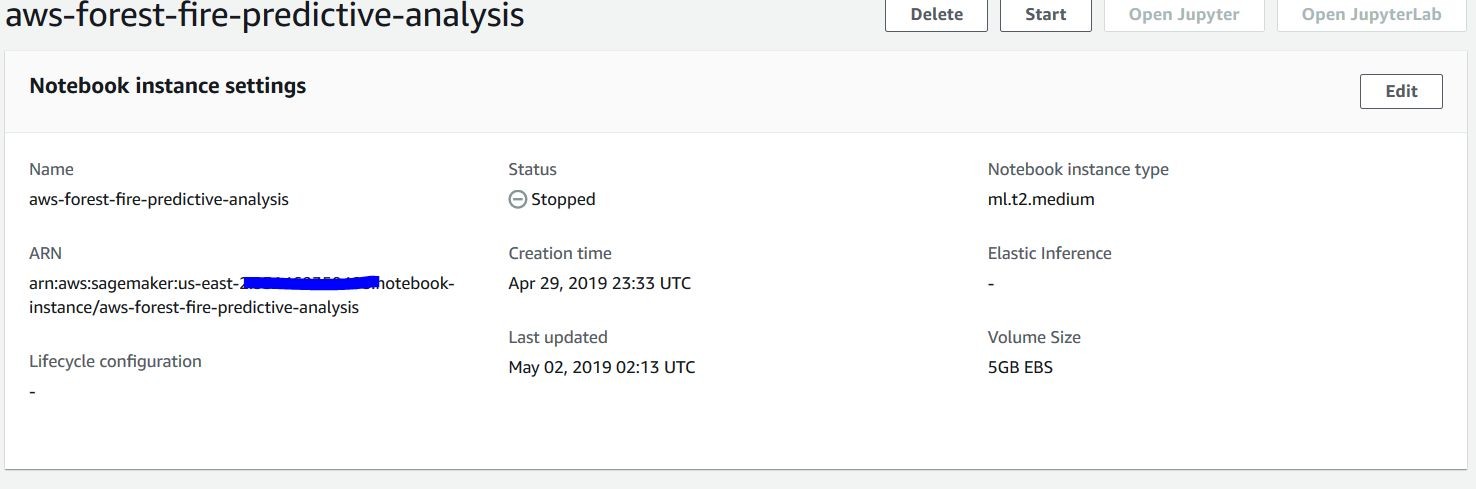
**AWS Glue**

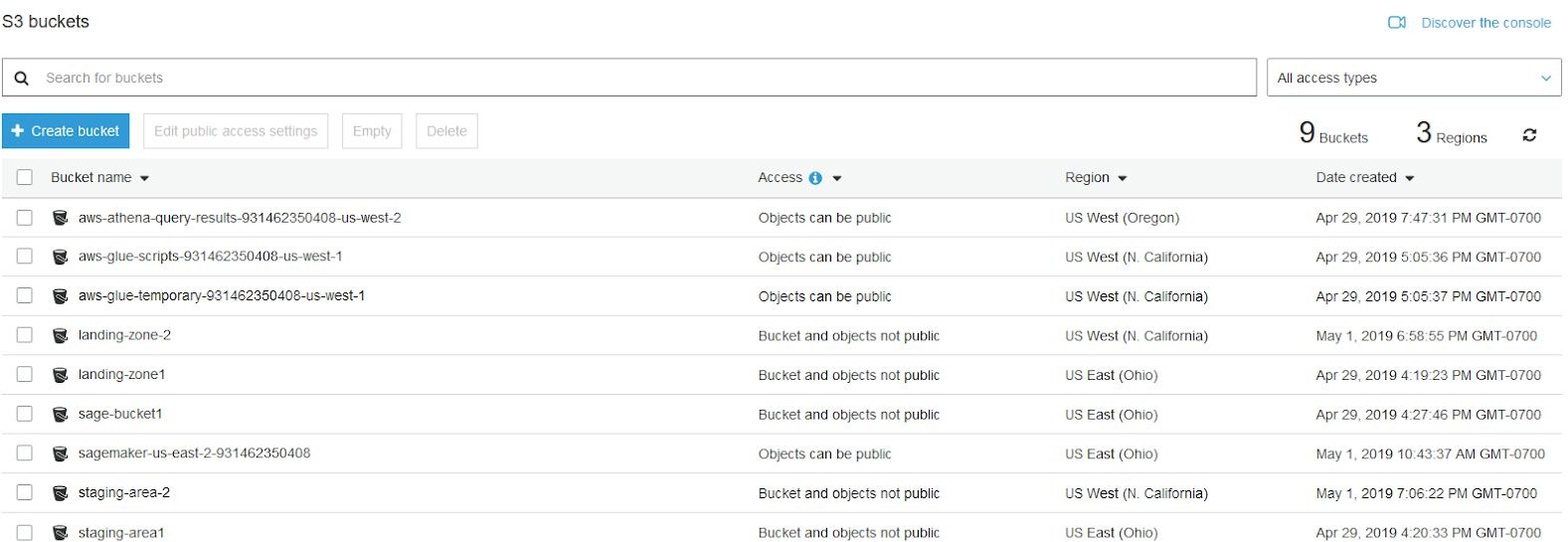


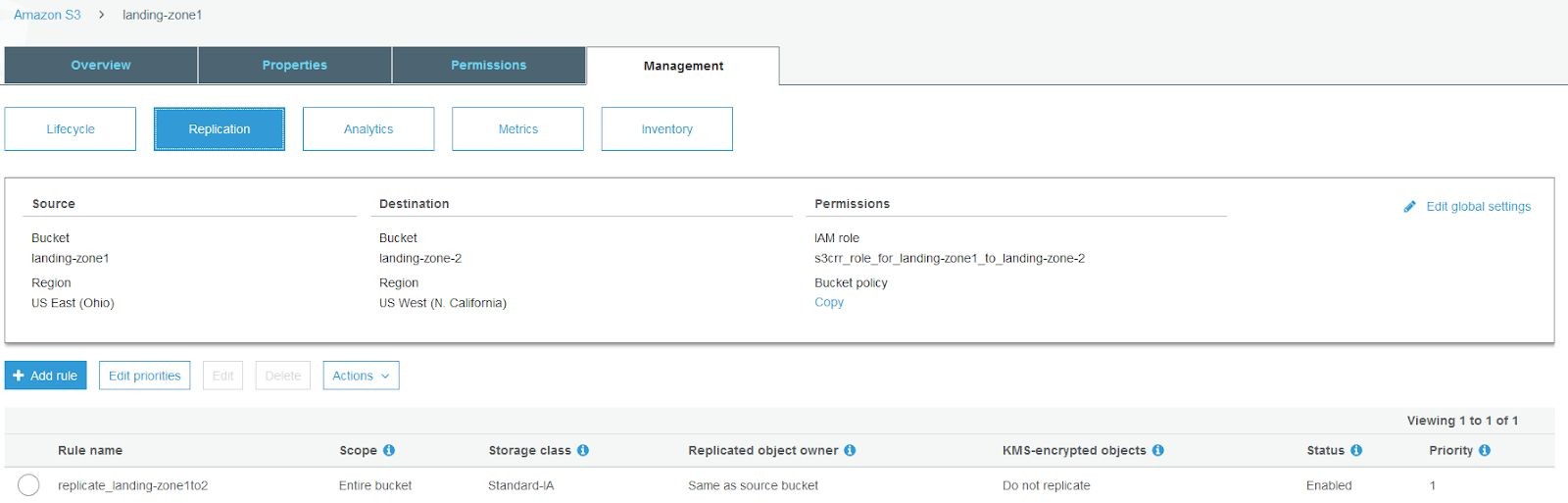
1. **AWS Configurations Screenshots**
   1. **Redshift Cluster**

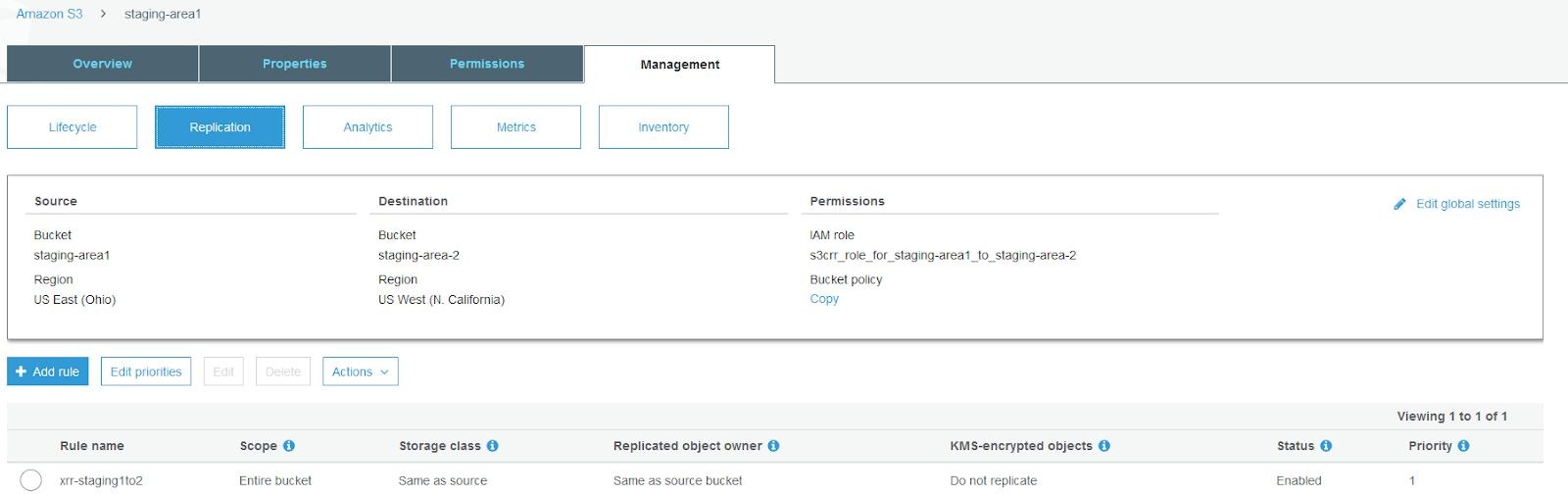


* 1. **Sagemaker**

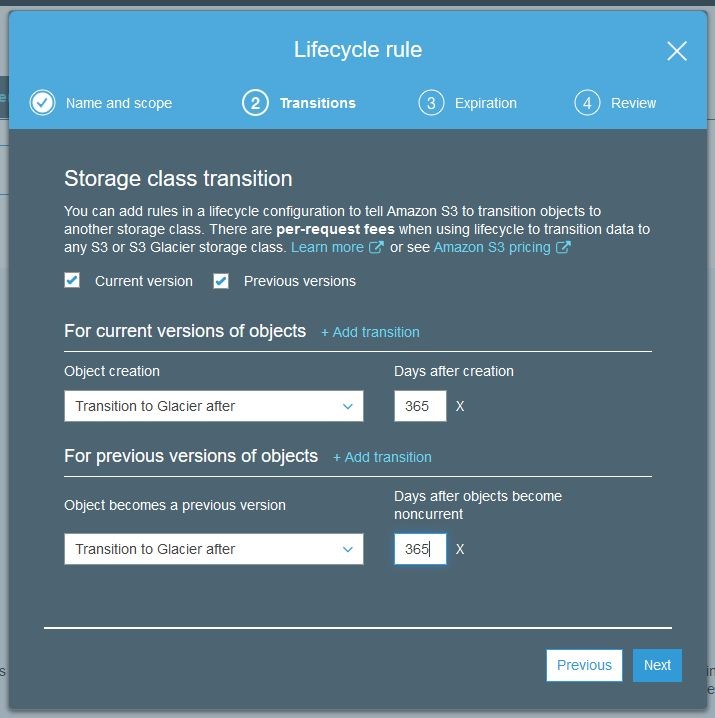


* 1. **S3**
     1. **Cross Region Replication**





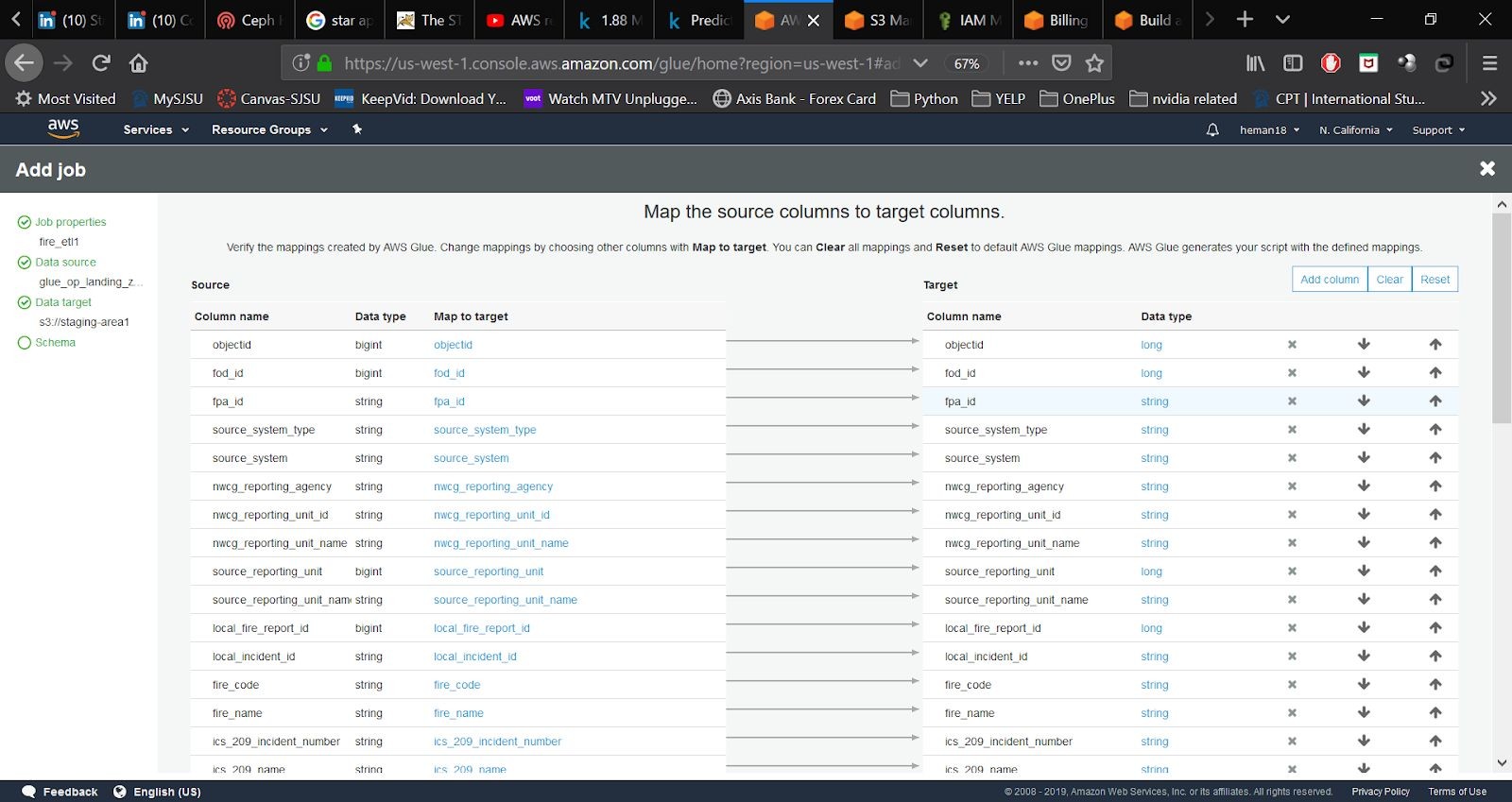
* + 1. **Transition to Glacier**

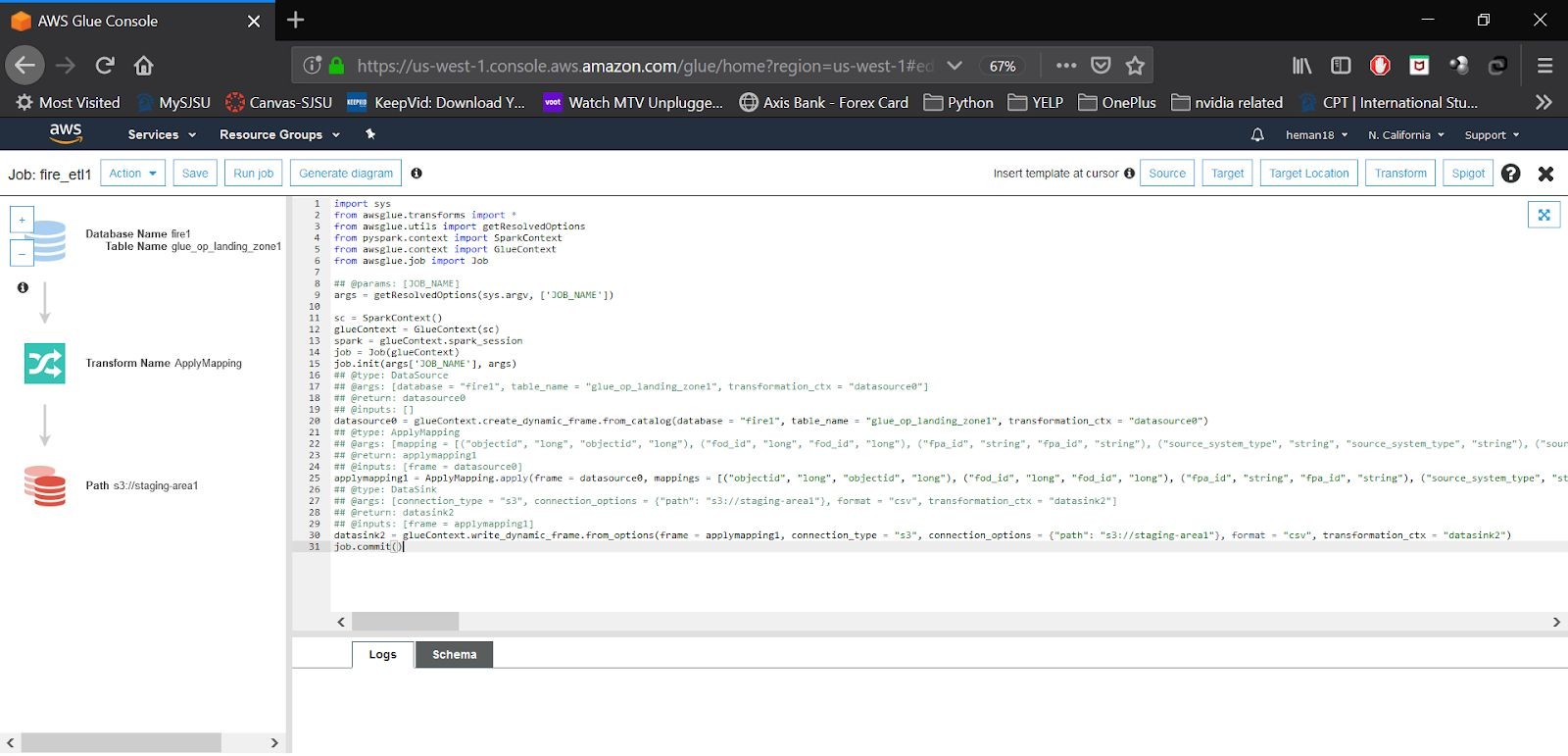


* 1. **Glue**
     1. **Crawler**

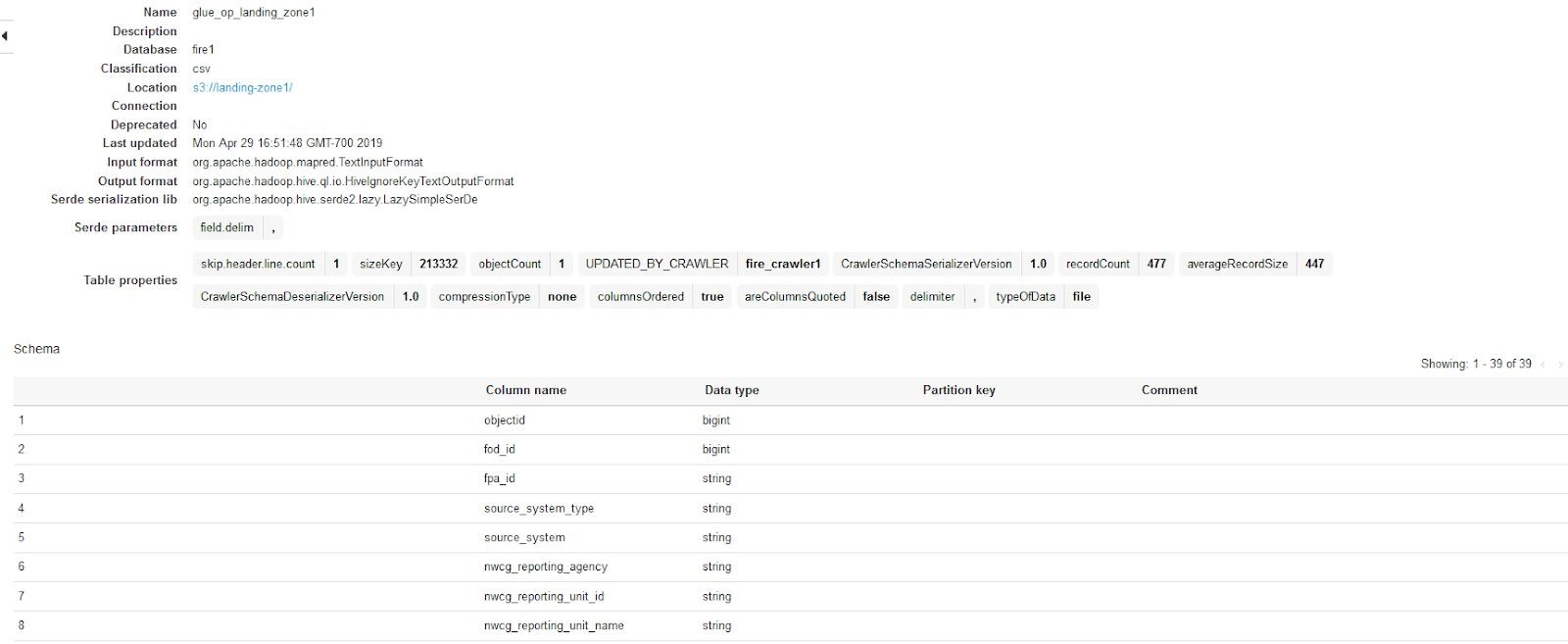


* + 1. **ETL Mapping**

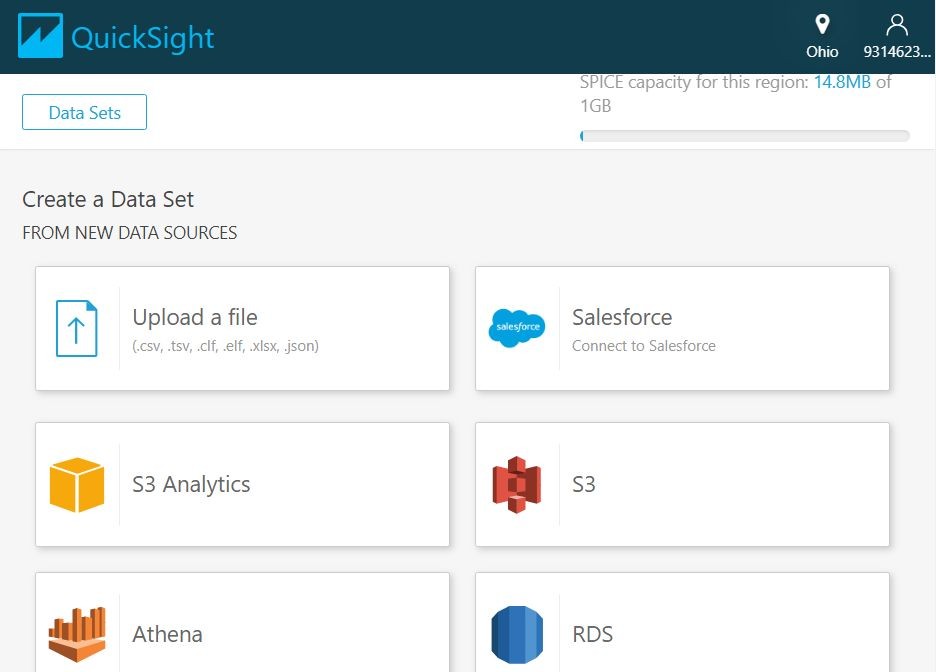


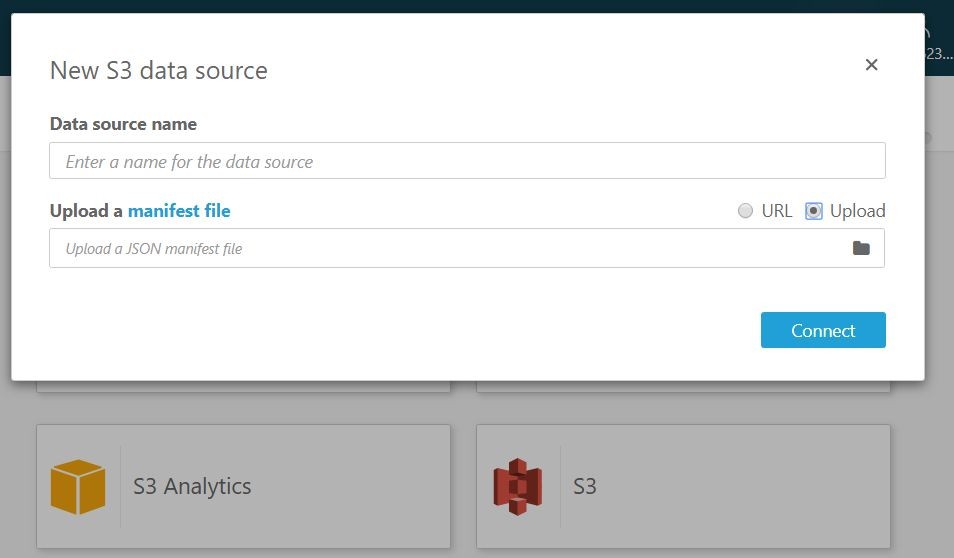


* + 1. **Glue Table**

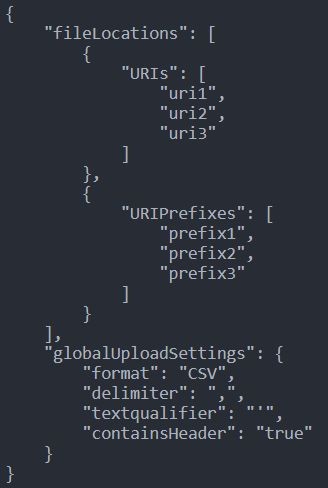


* 1. **QuickSight**

Connect an S3 bucket to QuickSight

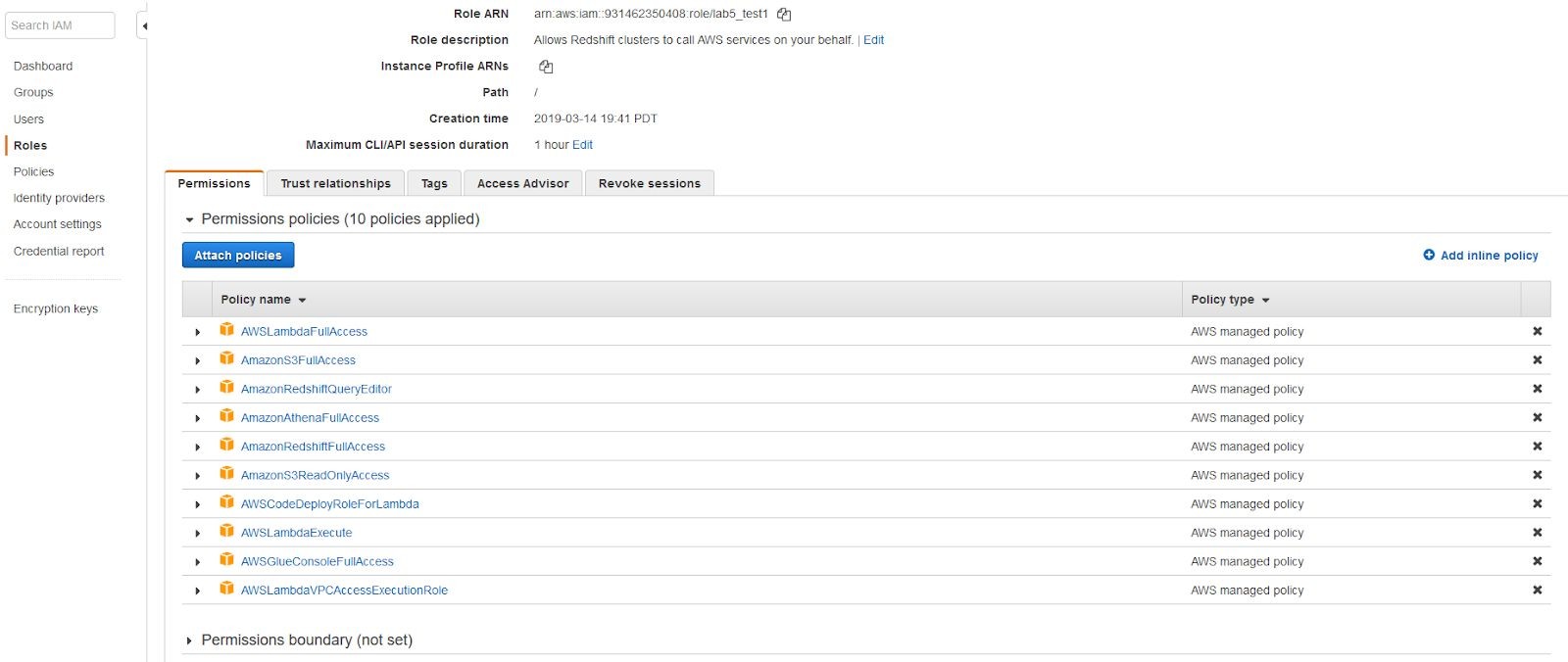


Manifest JSON file template and S3 file specifications



* 1. **IAM**

Policies:



# Solution Architecture

* + The main aim of the project is to develop a model which can identify and categorize the various fire incidents based on their cause.
  + The data file is first uploaded onto the S3 bucket.
  + We use Athena and Glue (using Glue crawlers) to read the data from the S3 buckets.
  + Since the initial data has multiple columns, we perform feature engineering to extract only the required columns
  + The data is cleaned using ETL functions and then the data is sent to the SageMaker where it is used to train the model
  + A model has been developed using Python libraries to categorize the various fire causes.
  + We are using QuickSight to analyze the data from the S3 buckets and create dashboards presenting the data in charts showing the various areas affected by fire. We created histograms depicting various information.
  + We are using three S3 buckets, each in different regions to increase availability and reliability.

# URL to GitHub.

<https://github.com/k-chuang/aws-forest-fire-predictive-analytics>

# Future Scope

* Using Lambda to automate the flow so that as soon as the document lands in the S3 landing area, the other functions such as ETL jobs and training models are triggered automatically.
* Online Training (i.e. training on the fly for data as it comes in) for SageMaker
* Create an interactive frontend for the user to upload the document for which analysis is wanted.
* The user can also query the system via the frontend.