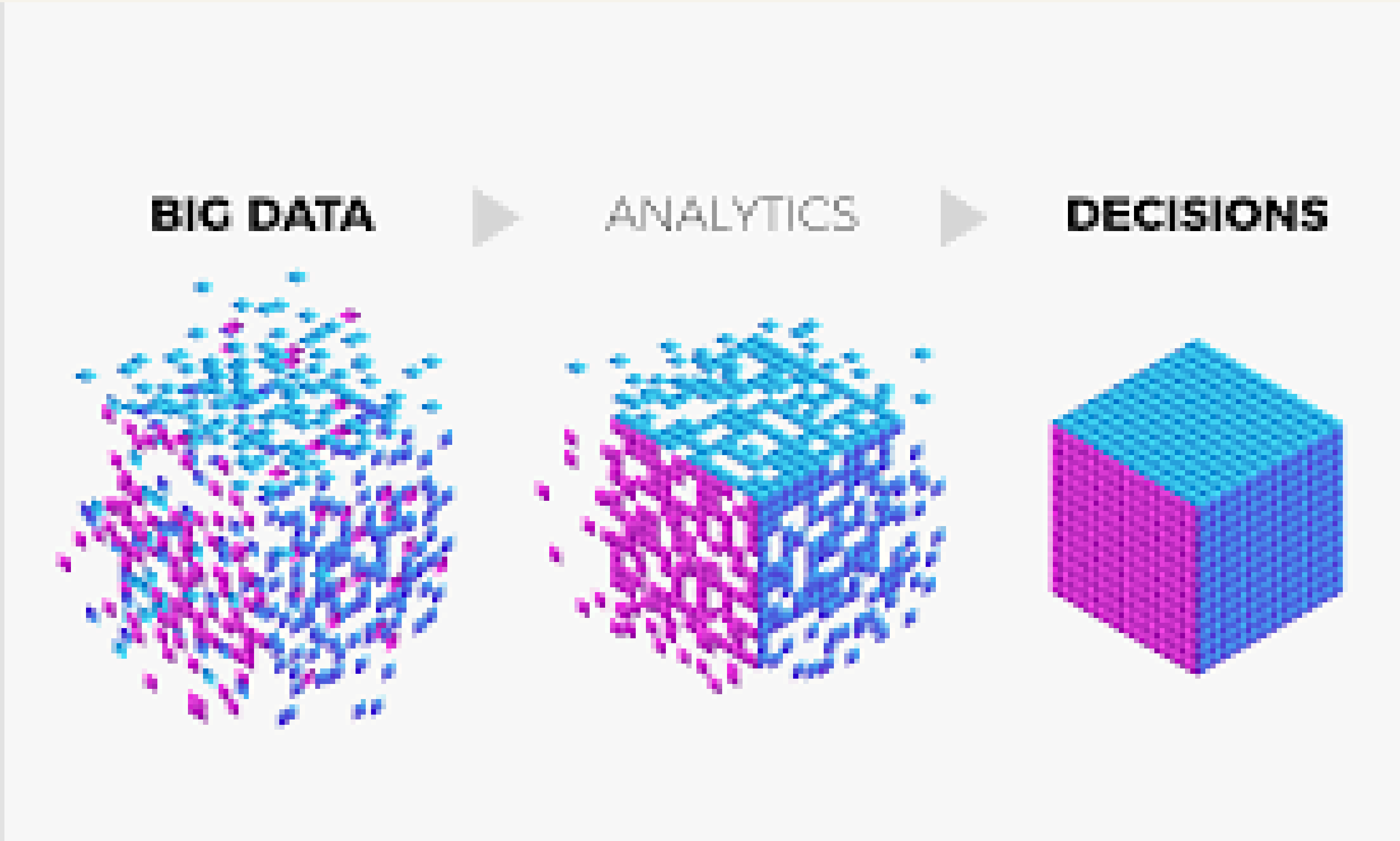
BIG DATA ANALYTICS GROUP PROJECT

GROUP 14

Under the guidance of : Dr. Sonali Agarwal



GROUP 14

Mohit Kumar - IIT2020220

Janhavi Bawaskar - IIT2020263

Pratyaksh Singh - IIB2020015

Harsh Kant Bhatnagar - IEC2020113

OVERVIEW

- Problem Statement
- Introduction
- Literary Review
- Overview of requirements
- Database
- Project Hypothesis
- Methodology
- Implementation
- Result
- Timeline
- Future Scope

PROBLEM STATEMENT

Design an InfluxDB with Apache superset model for real time smart building operations monitoring.

INTRODUCTION

- Real-time data stream processing is important in environmental monitoring systems to enable timely detection of anomalies and events.
- A CEP engine is a software system that processes data in a distributed manner and is used for datastream processing in environmental monitoring.
- This work presents a complex event processing (CEP) engine for detecting anomalies in real time

LITERARY REVIEW

Author name	Year	Paper Title	Application domain	Achieved Performance
Alejandro Buchmann, TU Darmstadt, Boris Koldehofe, Universität Stuttgart	2019	Complex Event Processing	IoT Smart Cities CEP	Studies focusing on performance metrics, scalability, and optimization techniques in CEP systems. Comparative analysis of various CEP systems
Alexander Y. Suna, Zhi Zhonga, Hoonyoung Jeongb, Qian Yanga	2019	Building complex event processing capability for intelligent environmental monitoring	Real-time Event Detection Complex event processing Predictive Analytics for Environmental Patterns	Complex event processing (CEP) engine for detecting anomalies in real time, and demonstrates it using a series of real monitoring data from the geological carbon sequestration domain.
Nithin Krishna Reghunathan	2020	Real-Time Streaming in Big Data: Kafka and Spark with SingleStore	Real-Time Streaming Big Data Technologies Stream Processing Pipelines Real-Time Analytics	Investigates the integration and capabilities of Kafka and Spark in conjunction with SingleStore to facilitate real-time streaming and processing within the realm of big data applications.
Haruna Isah,Tariq Abughofa, Sazia Mahfuz, Dharmitha Ajerla, Farhana Zulkernine, Shahzad Khan	Jan 2021	A Survey of Distributed Data Stream Processing Frameworks	Distributed Data Stream Processing Framework Analysis and Evaluation	Synthesizing and presenting existing information regarding the strengths, weaknesses, and characteristics of different distributed data stream processing frameworks based on available literature and evaluations performed by others

REQUIREMENTS

InfluxDB

- Real-time insights from any time series data with a single, purpose-built database.
- retrieval, and processing of large volumes of time-series data

Apache Spark

Spark is used for processing vast amounts of data in a distributed and parallel manner. It can handle both batch and real-time data processing

Apache Kafka

Apache Kafka is an open-source distributed streaming system used for stream processing, real-time data pipelines, and data integration at scale.

Grafana

Spark is used for processing vast amounts of data in a distributed and parallel manner. It can handle both batch and real-time data processing

DATASET

- The dataset which we will be using is Occupancy Detection Data Set UCI.
- The "Occupancy Detection Data Set" from the UCI Machine Learning Repository is a dataset that contains information related to occupancy detection in an office building. The dataset is typically used for binary classification tasks where the goal is to determine whether a room is occupied or not based on various sensor measurements.
- The attributes include: Date and Time, Temperature, Relative Humidity, Light, CO2, Humidity Ratio, Occupancy.

Code Blame 8144 lines (8144 loc) · 583 KB

```
"date","Temperature","Humidity","Light","CO2","HumidityRatio","Occupancy"
"1","2015-02-04 17:51:00",23.18,27.272,426,721.25,0.00479298817650529,1
"2","2015-02-04 17:51:59",23.15,27.2675,429.5,714,0.00478344094931065,1
"3","2015-02-04 17:53:00",23.15,27.245,426,713.5,0.00477946352442199,1
"4","2015-02-04 17:54:00",23.15,27.2,426,708.25,0.00477150882608175,1
"5","2015-02-04 17:55:00",23.1,27.2,426,704.5,0.00475699293331518,1
"6","2015-02-04 17:55:59",23.1,27.2,419,701,0.00475699293331518,1
"7","2015-02-04 17:57:00",23.1,27.2,419,701.666666666667,0.00475699293331518,1
"8","2015-02-04 17:57:59",23.1,27.2,419,699,0.00475699293331518,1
"9","2015-02-04 17:58:59",23.1,27.2,419,689.333333333333,0.00475699293331518,1
"10","2015-02-04 18:00:00",23.075,27.175,419,688,0.00474535071966655,1
"11","2015-02-04 18:01:00",23.075,27.15,419,690.25,0.00474095189694268,1
"12","2015-02-04 18:02:00",23.1,27.1,419,691,0.00473937073052061,1
"13","2015-02-04 18:03:00",23.1,27.1666666666667,419,683.5,0.00475111875560951,1
"14","2015-02-04 18:04:00",23.05,27.15,419,687.5,0.0047337317970825,1
"15","2015-02-04 18:04:59",23,27.125,419,686,0.00471494214590473,1
"16","2015-02-04 18:06:00",23,27.125,418.5,680.5,0.00471494214590473,1
"17","2015-02-04 18:07:00",23,27.2,0,681.5,0.00472807794966877,0
"18","2015-02-04 18:08:00",22.945,27.29,0,685,0.00472795137178073,0
"19","2015-02-04 18:08:59",22.945,27.39,0,685,0.0047454083970941,0
"20","2015-02-04 18:10:00",22.89,27.39,0,689,0.00472950615591001,0
"21","2015-02-04 18:10:59",22.89,27.39,0,689.5,0.00472950615591001,0
"22","2015-02-04 18:11:59",22.89,27.39,0,689,0.00472950615591001,0
"23","2015-02-04 18:13:00",22.89,27.445,0,691,0.00473907551474663,0
"24","2015-02-04 18:14:00",22.89,27.5,0,688,0.00474864516581148,0
"25","2015-02-04 18:15:00",22.89,27.5,0,689.5,0.00474864516581148,0
```

METHODOLOGY

- ## Phase 1

  Create a data delivery system, that continuously produces the data. For this the same data frame is iterated again and again.

- ## Phase 2

  Create a data ingestion system that delivers the data to spark, then processes the data and ingests it into influx db

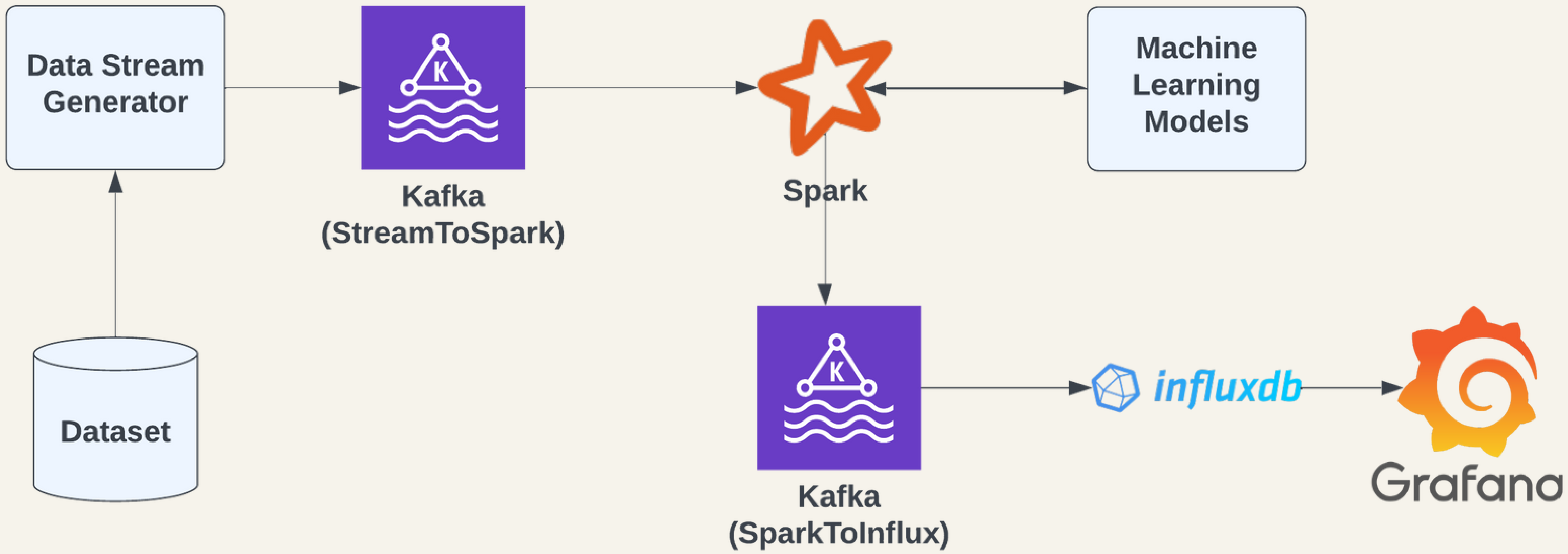
- ## Phase 3

  Create a connection from influx DB to Grafana so that data can be queried and graphs can be analyzed.

- ## Phase 4

  Create alerts in grafana which will help to analyse any abnormalities in the building data

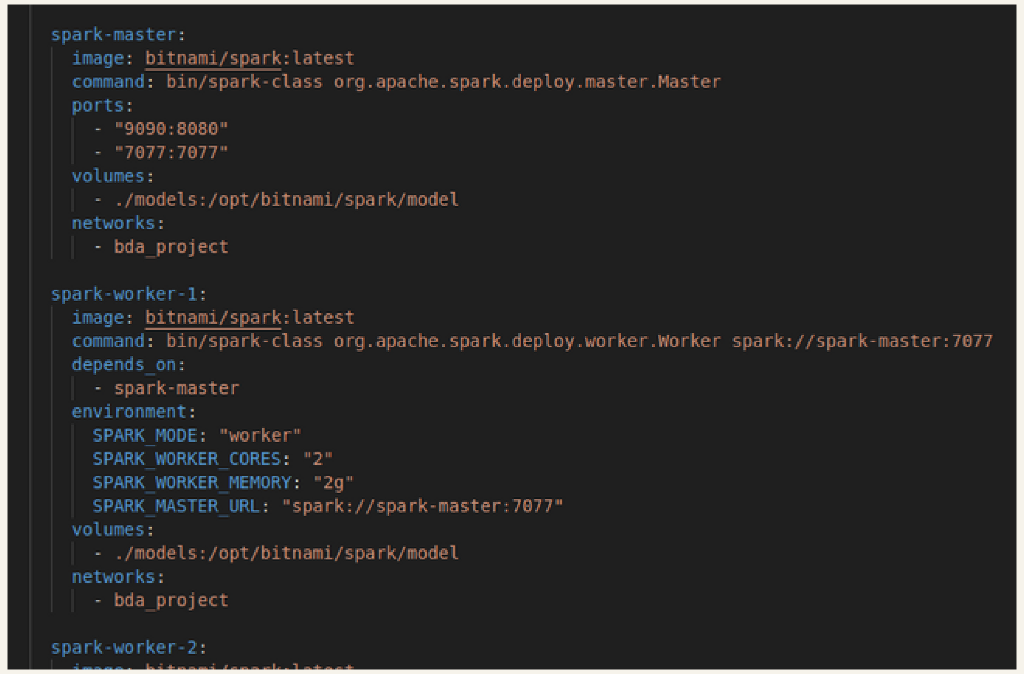
SYSTEM ARCHITECTURE



Docker used for running kafka, zookeeper, spark, influx-db and grafana in containers

Docker-Compose used to configure the services

Kafka acting as a message broker between data source & spark and spark & influxDB



Master Slave Configuration

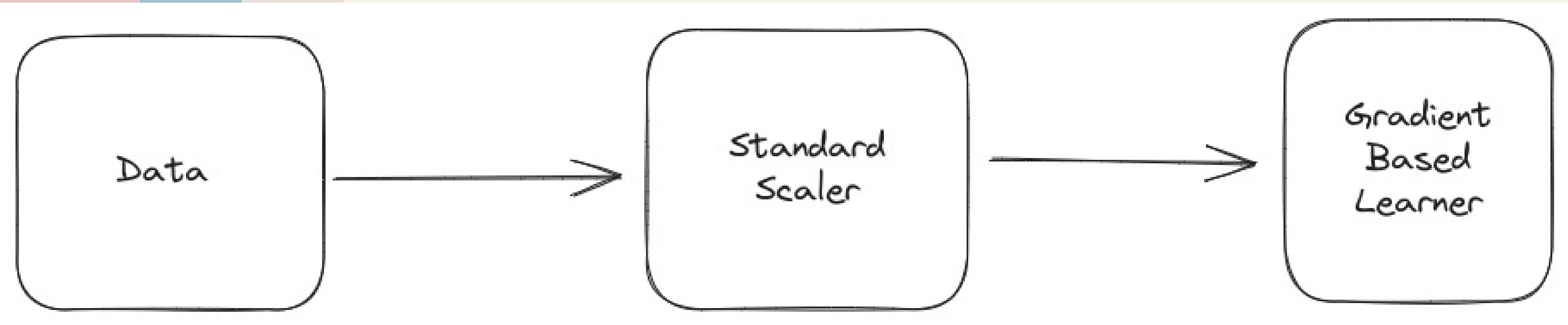
Spark used for real time processing and prediction

Multiple spark brokers running simultaneously (master-slave architecture) for efficient computation

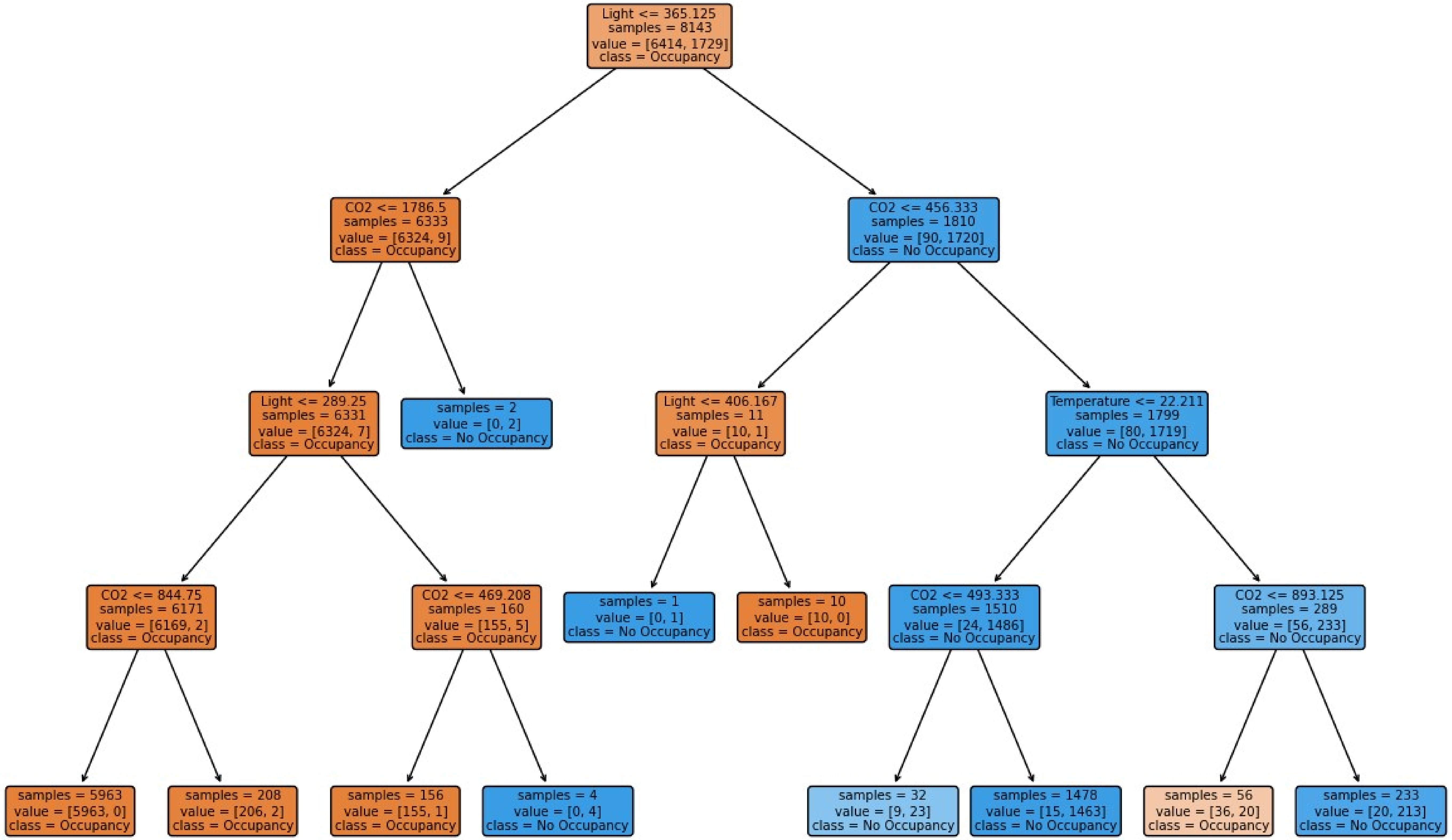
InfluxDB is a high speed read and write noSQL database used in time series applications

Grafana is a data visualization tool used for creating dashboards, alerts etc

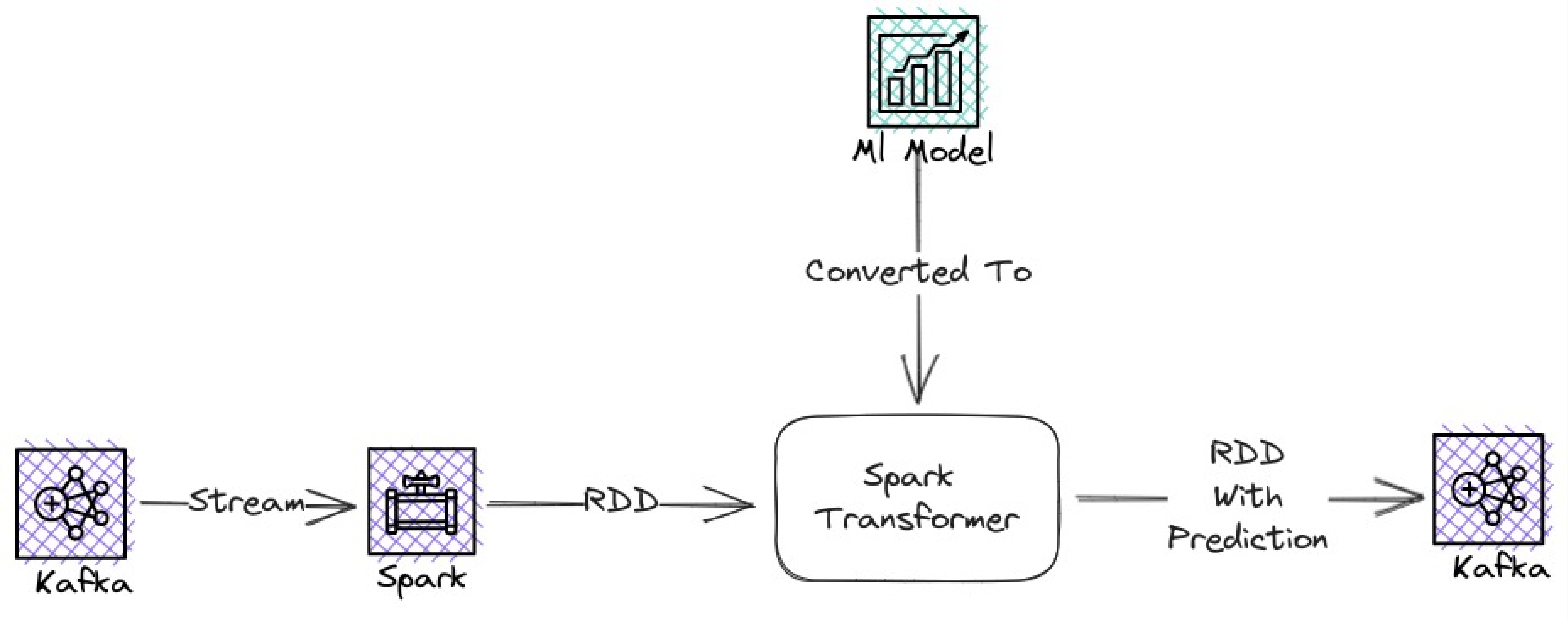
GRADIENT BASED LEARNER



TREE BASED LEARNER



PREDICTION ARCHITECTURE



LOCAL OUTLIER FACTOR

- The Local Outlier Factor (LOF) is a machine learning algorithm used for anomaly detection and outlier detection in datasets.
- It was introduced by Markus M. Breunig, Hans-Peter Kri~~egel, Raymond T. Ng, and Jörg S~~ander in a 2000 research paper.
- LOF is particularly useful for finding anomalies in high-dimensional datasets and those with complex structures.

GRAFANA

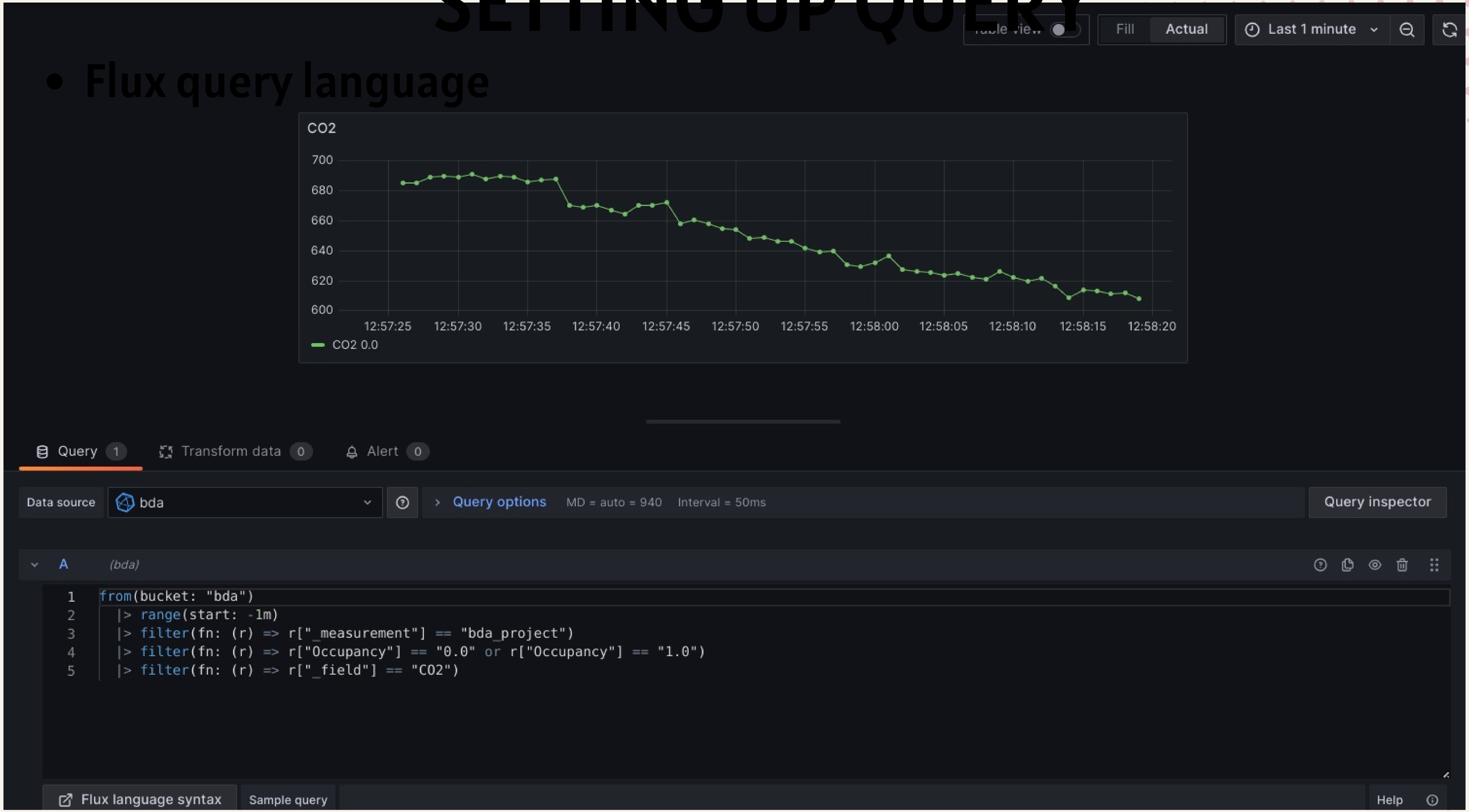
- Visualization is instrumental for assisting knowledge discovery and decision support.
- Our problem statement mentions the use of Apache Superset but due to certain limitations at hand, we have chosen to move forward with Grafana as our visualization instrument.
- It has a rich collection of data visualization tools, easy-to-use interface for uploading and transforming data, and seamless integration with commonly used data stores.

GRAFANA

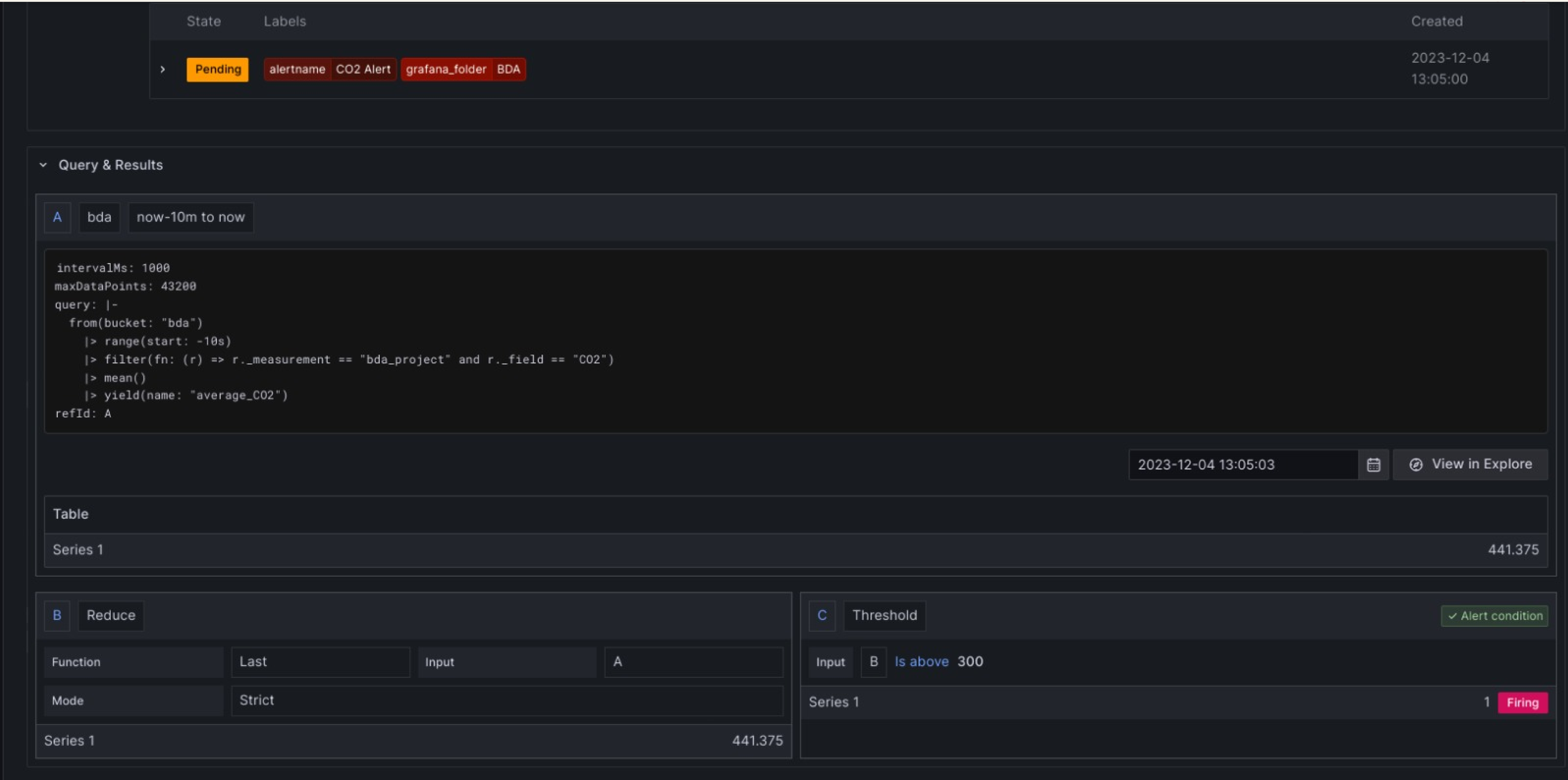
- Visualized data on the basis of various features.
- The features include:
  1. CO_2 level
  2. Humidity
  3. Temperature
  4. Light
- Flux query language is used to display the dashboards.
- Added alerts when CO_2 goes above 300. Here, 300 is the threshold value.

SETTING UP QUERY

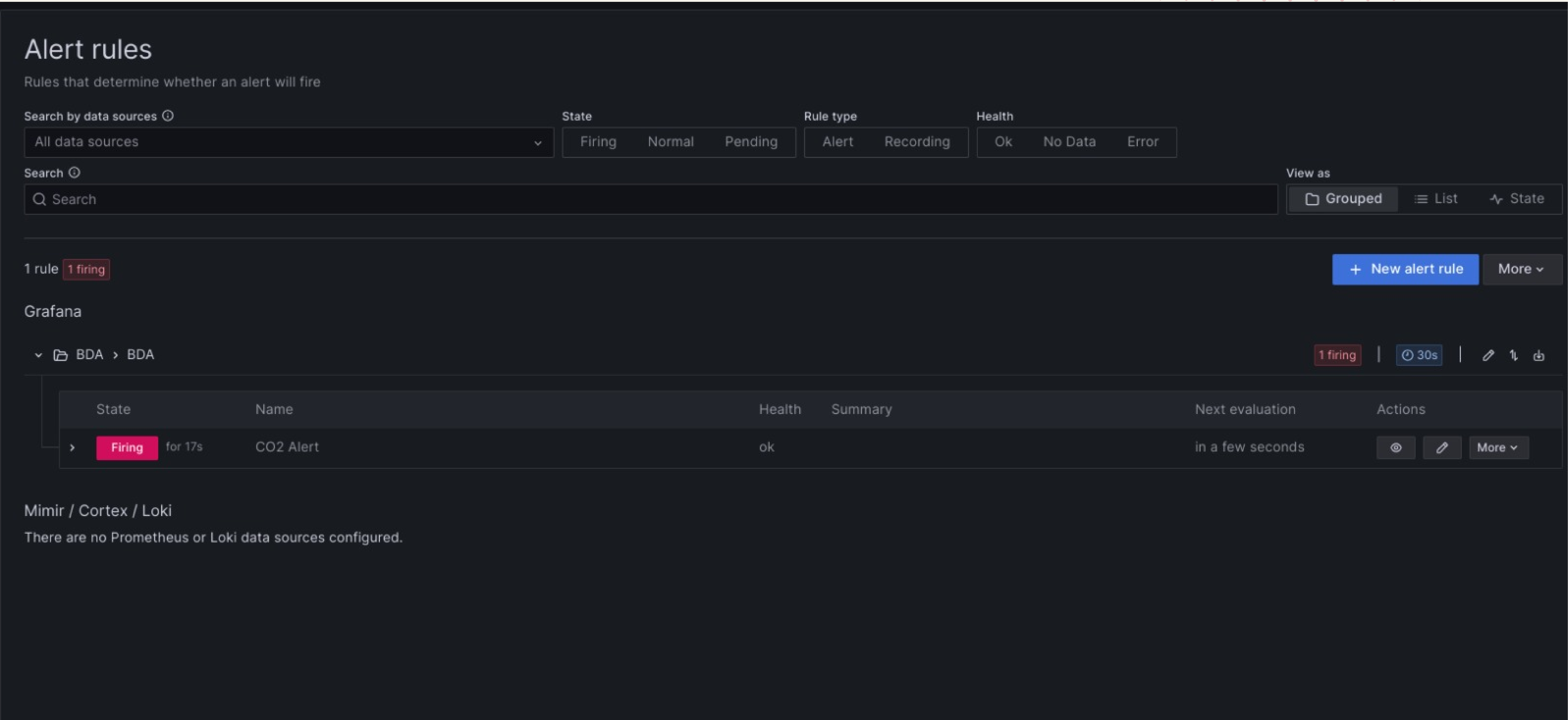
- Flux query language



SETTING UP ALERT



ALERTS FIRING



FINAL DASHBOARD

THANK YOU

Presented By : Group 14