




DECEMBER 11, 2024

CRIME DATA ANALYTICS SYSTEM

A DISTRIBUTED DATA SYSTEM

MIDATHALA VAMSI KRISHNA
UNIVERSITY OF THE PACIFIC
989467375



INDEX

- 1. Introduction**
 - 1.1. Objectives of the Project**
 - 1.2. Data Source**
- 2. System Architecture**
 - 2.1. Data system Components**
 - 2.2. Architecture Overview**
- 3. Environment Setup**
 - 3.1. Ubuntu configuration**
 - 3.2. Containerization with Docker**
- 4. Data Storage Management**
 - 4.1. Overview of HDFS**
 - 4.2. Database Management with SSMS**
- 5. Distributed Systems**
 - 5.1. Apache Spark**
 - 5.2. Spark with python**
- 6. Interactive Systems**
 - 6.1. JupyterHub setup**
 - 6.2. Jupyter Notebooks**
 - 6.3. Querying Distributed Data**
- 7. Data Flow**
 - 7.1. Data Ingestion**
 - 7.2. ETL Data Pipeline**
- 8. Dashboarding**
 - 8.1. Tableau Integration**
 - 8.2. Executive Summary Dashboard**
- 9. References**

1. Introduction

1.1. Objectives of the Project

The big task statement involves the design of a Distributed Data system that will get raw or unstructured data, process the same, and transform the same into actionable insights to be shown on an interactive dashboard. This uses SFPD data retrieved via an API as the feed of the data to create structured data and visualizations henceforth.

This will further help stakeholders formulate hypotheses that they can analyze more deeply, thereby coming up with more solid predictive models useful in the understanding and retardation of the trends of crime in San Francisco. Initial insights from the dashboard will enable stakeholders to make data-driven decisions that will help in making San Francisco safe for both its residents and tourists, the added value brought into this project by analyzing crime trends, while simultaneously showing that structured data has immense potential in enhancing the precision of a predictive model toward proactive measures of public safety.

1.2. Data source

The city of San Francisco will provide a wide variety of datasets to the public via its portal of data.sfgov.org. In the framework of the work, the San Francisco police department incident reports dataset was selected, which had historical data from 2018 to 2024. It contains about 80,000 entries with 32 attributes of detailed information on incidents.

The on-duty officers create data in this dataset with the help of reports submitted to the SFPD reporting system by the public. This dataset allows for many options in terms of data processing and analytics, such as JSON, CSV, and GeoJSON. Once this dataset is

updated, every record goes through the review and approval process conducted by a supervising sergeant or lieutenant, reflecting quality and reliability within data. This data is under the license of Open Data Commons Public Domain Dedication and License, and it is updated daily by the authorities, This dataset would be a good fit for big data analytics about the project because it is diverse, high in scale, and rapid in velocity, those features just happen to exactly fit the needs of our distributed data system for broad analysis and deriving insight.

2. System Architecture

Data System Architecture gives a high-level knowledge of how the system functions by outlining the interconnection between different components and illuminating the data flow inside the system.

2.1. Data system Components

Understanding the various components of the data system and their functions in creating an effective crime data analytics system is crucial before delving into the architectural overview. The following elements make up this project's ecosystem:

Ubuntu:

The underlying operating system, Ubuntu, offers a reliable and secure foundation for the distributed data system's deployment. It is a solid option due to its open-source nature and interoperability with data engineering tools.

Technical configurations:

- Version – Ubuntu 24.04.2 LTS
- Kernel – GNU/Linux 5.15.167.4-Microsoft-standard-WSL2
- Architecture – x86_64

Docker - Facilitates containerization, guaranteeing the uniformity and expandability of programs in various settings. System

component deployment and administration are made easier using Docker containers.

Technical configurations:

- Version – Docker Desktop 4.36.0 (175267)
- OS – Linux
- RAM – 6GB
- Disk Space – 1020GB

Hadoop Cluster – HDFS

HDFS enables the management of a substantial amount of SFPD crime data by offering a distributed storage solution. HDFS is perfect and unprocessed data because it guarantees scalability and fault tolerance.

Technical configurations:

- Version – Hadoop 3.0.0-cdh6.2.0
- Source - Cloudera Hadoop repository
- Compiled by - Jenkins on 2019-03-14T06:39Z
- Protoc Version – 2.5.0

Python and Apache Spark (Pyspark)

Pyspark makes it easier to process and analyze big datasets using distributed computing. Pyspark is used for data transformation and analytics because of its strong data processing capabilities and easy integration with Python.

Technical configurations:

- Version – 2.4.0-cdh6.2.0
- Python 3
- JVM - OpenJDK 64-Bit Server VM 1.8.0_201
- Compiled by: Jenkins on 2019-03-14T07:01:24Z

MS SQL Server and SSMS

It serves as the structured data storage system for relational database management systems (RDBMS). Effective database management and querying is accomplished with SSMS

Technical configurations:

- Version – 20.2.30.0
- OS – Windows Server 2016
- CPU – 8 cores
- RAM – 8 GB

Tableau

An effective data visualization tool for making dynamic dashboards. Tableau facilitates the clear and useful presentation of trends and insights from crime data.

Technical configurations:

- Version – 2024.3.0
- OS – Windows 11
- CPU – 8 cores
- RAM – 8 GB

Together, they allow for effective data ingestion, processing, analysis, and visualization to meet the project's goals. Each component is essential to the system.

2.2. Architecture Overview

High-level view Diagram of the Data system

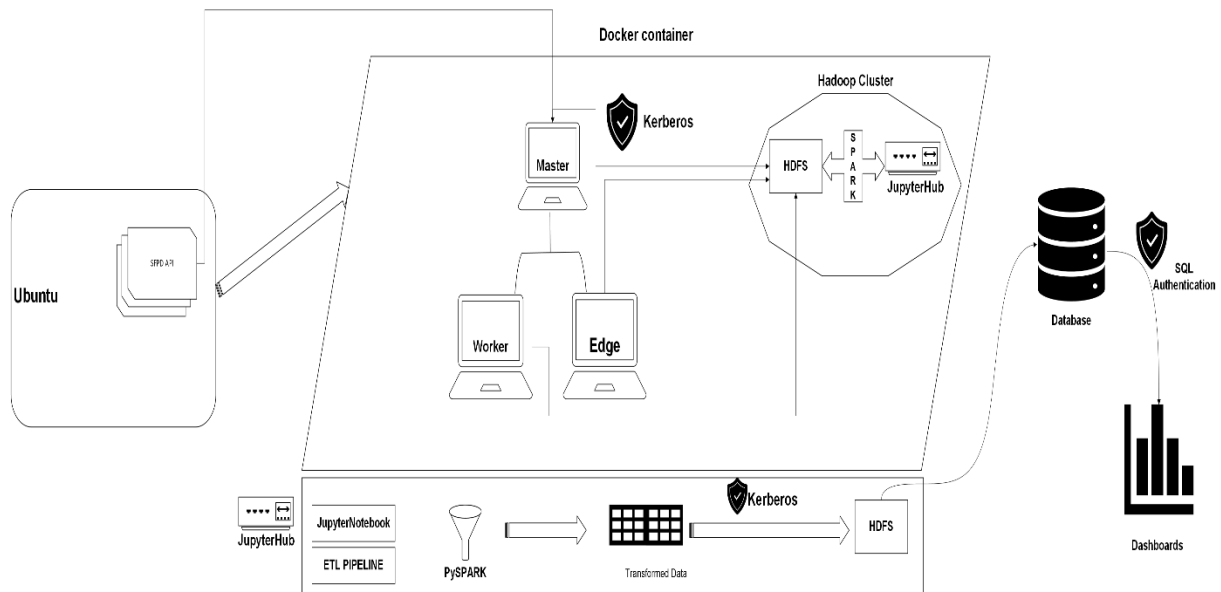


Fig 1.1

Data Flow:

- Data ingestion
SFPD API pulls raw crime data using the CURL method. Secure Kerberos authentication is used to send data to the Hadoop cluster.
- Data processing
Using the ETL pipeline, PySpark transforms unstructured data into structured formats in Jupyter Notebook. HDFS is where transformed data is kept.
- Storage and Querying
Structured data is managed by an MS SQL server for additional analysis. Data access is secured by SQL authentication.
- Visualization
To create crime trend dashboards, Tableau connects to SQL Server.

3. Environment Setup

3.1. Ubuntu Configuration

The Windows Subsystem for Linux(WSL) is used to enable a Linux environment on a Windows computer to set up Ubuntu for the distributed data system. Ubuntu must be downloaded and configured as a WSL distribution. Ubuntu application can be downloaded from the Microsoft store. In Ubuntu OS create a directory to store all the project parts in one place.

Commands:

```
wsl --install #to install wsl
```

```
wsl -ls -v #to list all installed distributions and current versions
```

3.2. Containerization with Docker

For containerization, use an already existing docker image containing all the necessary configurations and dependencies therein. These will then be included in the Ubuntu project directory for further facilitation of smooth deployment.

Steps of containerization:

The docker image and docker-compose file in the git repository need to be downloaded to the project directory using the git clone method in the Ubuntu terminal.

The repository contains all the configuration files; thus, it will oversee defining services, settings, and dependencies.

Correctness and compatibility of the Docker file and docker-compose.yml shall be verified inside the Ubuntu environment right after cloning.

The Docker images are built and start running containers, installing the needed services of the data system.

Docker Desktop:

Docker Desktop can be used to control containers like start pause and restart with easy clicks on the user interface. This is a GUI-based application and includes several factors such as integration with Docker Hub: Easy procurement of pre-constructed images.

Container Management - simplifies monitoring and management of running containers, including viewing logs and restarting services.

Resource Monitoring - Monitors CPU, memory, and disk usage of containers.

Resource Dashboards - Graphically visualize resource utilization and performance for each container.

Benefits of containerization:

Docker containers are consistent across different environments providing portability. Isolate each component within the container providing isolation. New containers can be added easily thus simplifying the process of scaling services.

Commands:

`git clone <repository name>`

`docker compose up -d #to start docker containers`

`docker ps #to list all the running containers and port information`

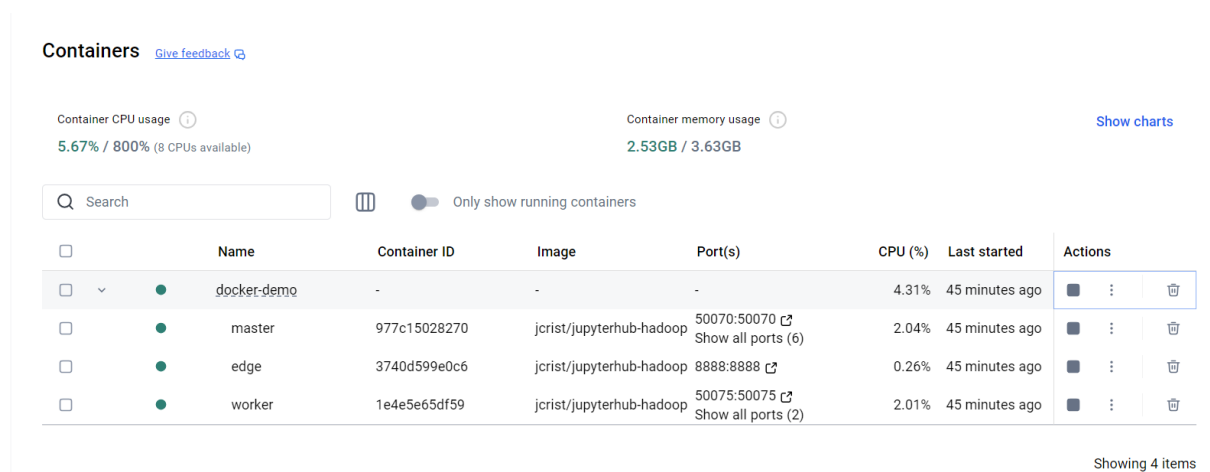


Fig 1.2

4. Data Storage Management

In this project, HDFS is used as a storage component of raw and processed data. SSMS is used as a database platform to store processed data facilitating data queries with enabling ease and seamless connection to other BI tools.

4.1. Overview of HDFS

Hadoop Distributed File System (HDFS) is employed for this project to store and manage security for large datasets safeguarding scalability and fault tolerance.

The data file in the Ubuntu project directory is transferred to the master container directory to make data accessible for the HDFS environment.

To access HDFS it is secured with Kerberos authentication, this system requests the user to provide valid login credentials by generating a ticket upon successful login with a time validity of 24 hours. Then create a directory in HDFS to store incoming raw data files streamlining access and processing.

Raw data file from the master container directory is uploaded to the raw directory in HDFS making it accessible for processing. Once data is uploaded needs to be verified to confirm data is distributed across all the nodes available without errors.

```
[root@master /]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: alice@EXAMPLE.COM

Valid starting    Expires          Service principal
12/11/2024 23:45:58 12/12/2024 23:45:58  krbtgt/EXAMPLE.COM@EXAMPLE.COM
[root@master /]# hdfs dfs -ls /user/alice/
Found 4 items
drwxr-xr-x  - alice supergroup          0 2024-12-05 05:58 /user/alice/.skein
drwxr-xr-x  - alice supergroup          0 2024-11-29 00:41 /user/alice/.sparkStaging
drwxr-xr-x  - alice supergroup          0 2024-12-05 04:59 /user/alice/notebooks
drwxr-xr-x  - alice supergroup          0 2024-11-29 00:56 /user/alice/raw
```

Fig 1.3

4.2. Database Management with SSMS

Processed data from the ETL process will be saved in a Microsoft SQL Server database. This data can be easily arranged and stored in distinct tables inside SSMS based on specific use cases by assigning appropriate data types to each feature, data integrity and query efficiency can be improved.

To access and analyze this data, users must log into the SQL Server, which provides a strong security layer. Data access can be regulated at the granular level using the user roles set in SSMS. SQL Server connects with a variety of business intelligence and visualization tools allowing smooth integration across sectors.

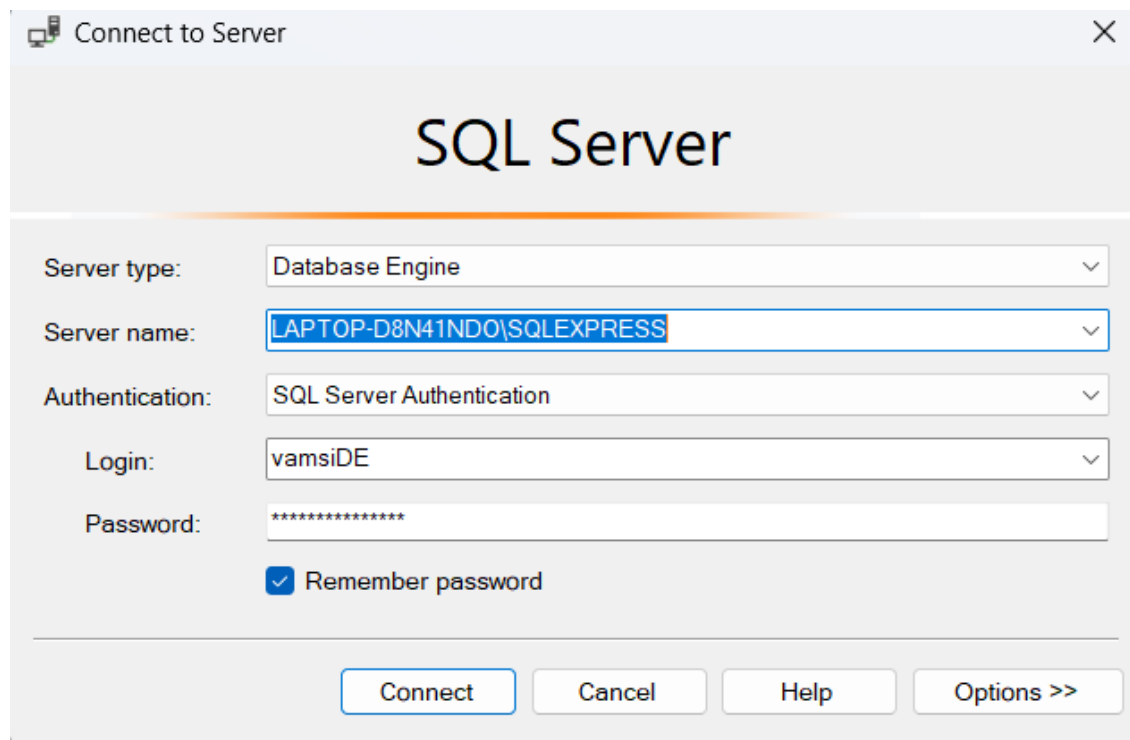


Fig 1.4

5. Distributed Systems

Crime Data analytics system is a distributed data system by differentiates tasks between the components and the containers to optimize the performance and reduce latency.

5.1. Apache Spark

Apache Spark is a distributed computing component in the Hadoop cluster performing data processing tasks. It provides a more efficient alternative to MapReduce for running data pipelines.

Worker nodes In the cluster mostly use Spark to process and transform huge datasets. Spark's in-memory computing capabilities and efficient execution engine make it ideal for ETL processes, which require rapid data intake, transformation, and loading into data warehouses.

5.2. Spark with python

Spark provides a Python API called PySpark to make Python-based data processing easier within the Hadoop cluster. Users can use Python to execute data pipelines and perform data analysis operations directly on the cluster by importing the required spark modules and creating a spark session in Jupyter Notebook. This connection allows you to write and execute spark tasks with familiar Python syntax in Jupyter Notebook.

```
In [1]: import pyspark

In [2]: from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("ETLpipeline").getOrCreate()
spark

Out[2]: SparkSession - in-memory
SparkContext

Spark UI
Version
v2.4.0-cdh6.2.0
Master
yarn
AppName
ETLpipeline
```

Fig 1.5

6. Interactive systems

6.1. JupyterHub

JupyterHub is an interactive environment component used to run data pipelines. It is not the only choice for offering users a notebook environment with integration, but it does have certain advantages for fitting into the data system.

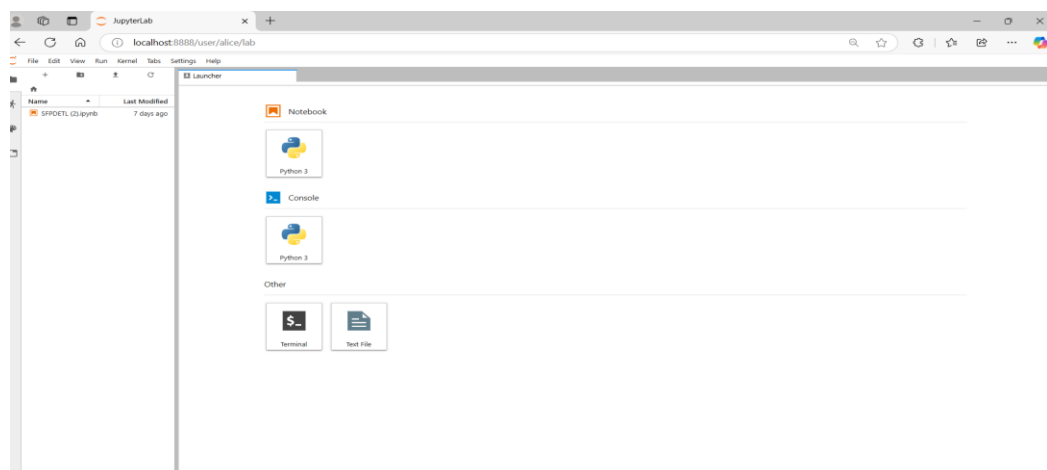
JupyterHub offers the same familiar Jupyter UI. It integrates effectively into the existing data science ecosystem and is widely used in both the business and public sectors. With JupyterHub, every user has their environment operating in their containers. This decreases the load on a single node while allowing resource utilization to scale dynamically with the number of users. Spark and Dask work naturally with no additional overhead when dealing with big amounts of data.

JupyterHub is adaptable and does not rely on a single cluster management. It performs well on both Hadoop clusters and single workstations. This implies that if infrastructure changes in the future, JupyterHub still can be used as the same.

For this data system image, the creator has provided three logins to get into JupyterHub. With these credentials can log login to JupyterHub to use the services.

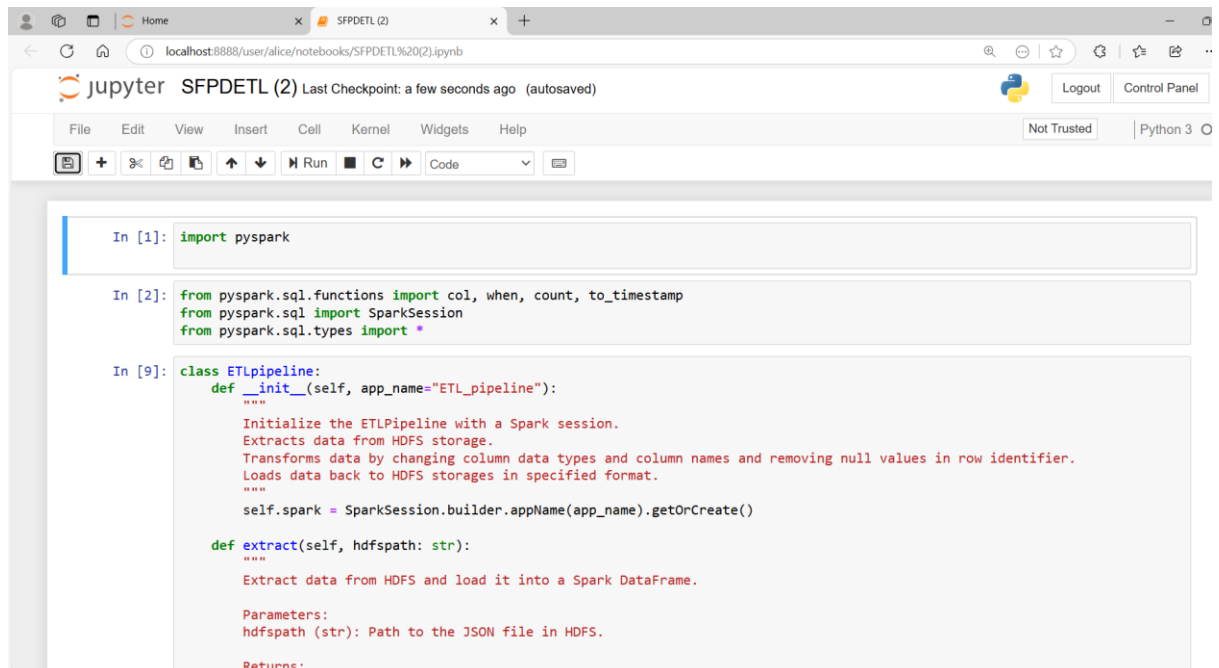
usernames: alice, Bob, Alex

Password: test pass



6.2. Jupyter Notebooks

Jupyter Notebooks is the place where I performed ETL pipelines with PySpark jobs. Notebooks are user-friendly and a user can write code and report information within the notebook same time.



The screenshot shows a Jupyter Notebook titled 'SFPDETL (2)' running on a local host. The interface includes a top bar with the Jupyter logo, the notebook title, and a 'Last Checkpoint' status. Below the top bar is a menu bar with options like File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A toolbar with icons for running, saving, and other actions is also present. The main area displays three code cells:

```
In [1]: import pyspark

In [2]: from pyspark.sql.functions import col, when, count, to_timestamp
        from pyspark.sql import SparkSession
        from pyspark.sql.types import *

In [9]: class ETLpipeline:
        def __init__(self, app_name="ETL_pipeline"):
            """
            Initialize the ETLpipeline with a Spark session.
            Extracts data from HDFS storage.
            Transforms data by changing column data types and column names and removing null values in row identifier.
            Loads data back to HDFS storages in specified format.
            """
            self.spark = SparkSession.builder.appName(app_name).getOrCreate()

        def extract(self, hdfspath: str):
            """
            Extract data from HDFS and load it into a Spark DataFrame.

            Parameters:
            hdfspath (str): Path to the JSON file in HDFS.

            Returns:
```

Fig 1.6

6.3. Querying Distributed Data

After data is transformed and loaded, it is stored in a data warehouse to facilitate quick and efficient access. This processed data becomes ready for querying and analysis. SQL queries can be written and executed directly in the Jupyter notebooks, providing an integrated environment for analysis. For enhanced user experience and a more robust interface SQL Server Management Studio is used.

SQL query:

Analyze incident category distribution by providing a count of recorded incident categories from highest to lowest.

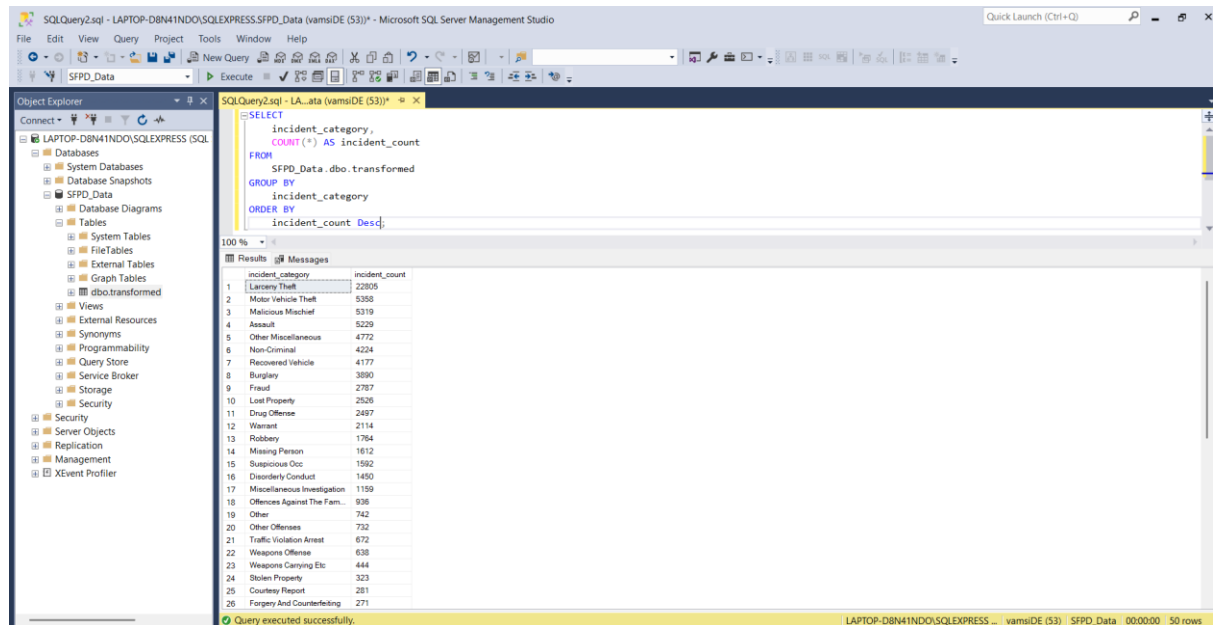


Fig 1.7

7. Data Flow

7.1. Data ingestion

This process involves importing raw data into the desired location that can be accessed while performing the ETL pipeline. In our case, the end location of data ingestion is the HDFS directory.

Data ingestion looks like

From open data portal to ubuntu project directory using curl command in the terminal.

From there data is copied to the docker container directory and then put into the hdfs directory with the Hadoop put command.

This ingestion flow guarantees the seamless transfer of raw data to its destination folder which is a distributed storage system.

Commands are as follows:

```
curl -G -o data.json "https://data.sfgov.org/resource/wg3w-h783.json" --  
data-urlencode '$limit=80000'  
docker cp data.json master:/rawdata_SFPD.json  
hdfs dfs -put /rawdata_SFPD.json /user/alice/raw/
```

7.2. ETL Data Pipeline

Extract, Transform, and Load pipeline is applied to support batch processing, safeguarding the effective handling of large datasets. ETL process is done using PySpark jobs within in Jupyter Notebook.

Extract

Data is extracted from the HDFS layer using PySpark which efficiently handles raw data.

Performed initial data exploration

Data types of features are inspected.

Data validation is conducted to check inconsistencies because of a large dataset.

Dimensions of the dataset are identified

To test the connection between PySpark and hdfs

```
] : spark.read.text("hdfs:/user/alice/raw").show()  
# tests connection between spark and hdfs]
```

Transformation

Data was cleaned and ready for further analysis.

Features with incorrect data types are resolved using column and cast methods.

Attributes names are renamed removing the special characters between words ensuring consistency.

The Null value count of each feature is printed for transparency.

Row ID is an attribute that uniquely identifies rows with a number, is checked for null values and if found is removed from the dataset.

Load

Processed data is saved in the desired format which in this case is CSV. This format is compatible with downstream tools and systems. Processed data is then loaded back into the HDFS directory making it accessible for querying and analysis.

8. Dashboarding

8.1. Tableau Integration

Tableau is used as the primary visualization tool in this project for creating interactive dashboards and reports.

Tableau supports integration with various data sources including on-premised systems, cloud platforms, servers, and local files. It also accepts a wider range of data formats.

SQL server database can be connected to Tableau using valid SQL server credentials. This connection allows Tableau to fetch the processed data efficiently and in real time. Tableau provides an intuitive drag-and-drop interface, enabling users to experiment more without any coding or prior knowledge of dashboarding.

Tableau's powerful features make it easy to create customized dashboards tailored to stakeholders' needs.

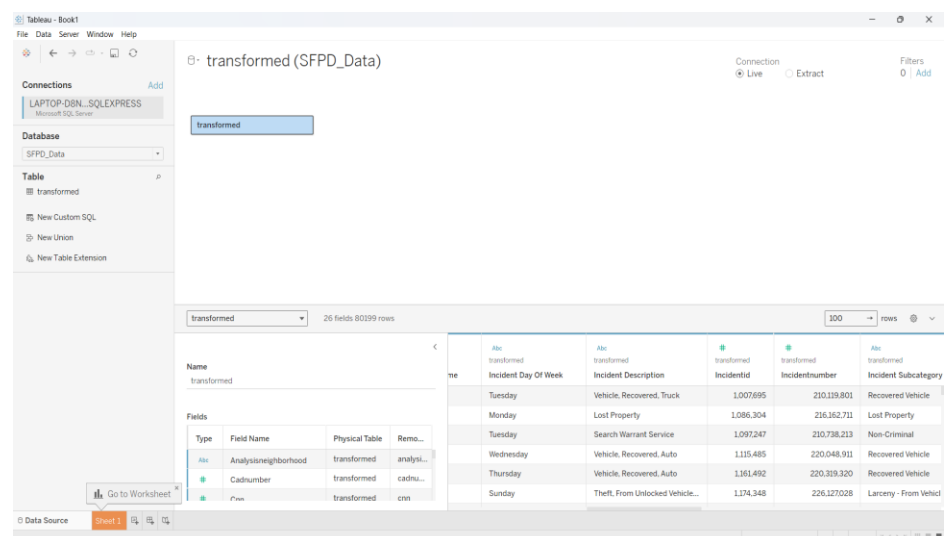


Fig 1.8

8.2. Executive Summary Dashboard

Structured data is now ready to provide insights through visualizations in Tableau.

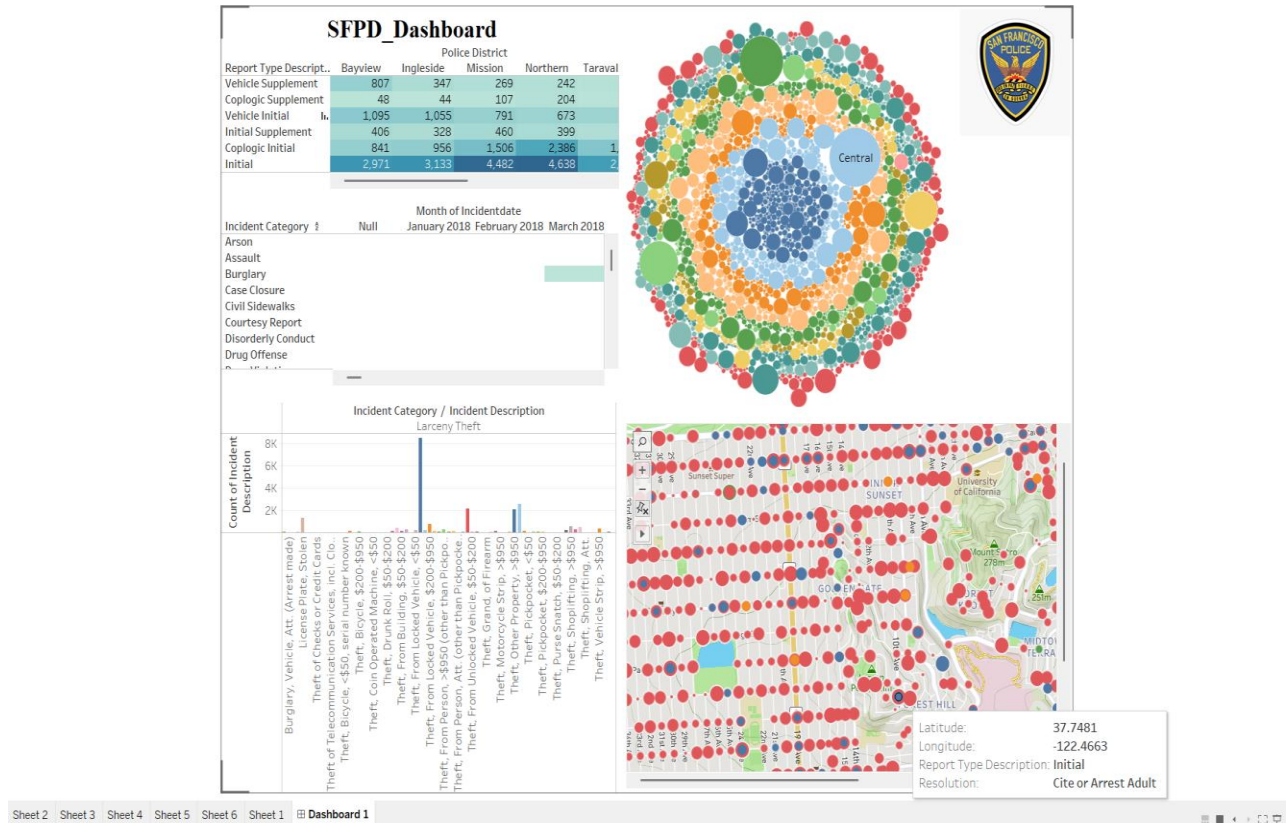


Fig 1.9

The San Francisco police department dashboard gives critical insights into crime trends across SFO's police districts. This information is an invaluable resource for law enforcement agencies and policymakers, allowing them to build successful strategies and regulations to improve public safety.

Larceny theft is the most reported criminal category in all police districts. The dashboards help stakeholders identify high-crime regions and prioritize resources accordingly. Provides actionable insights to help improve safety measures and provide a safe atmosphere for the public.

9. References

(Joe Reis, 2022)

(White, 2009)

(Ubuntu and Canonical are registered trademarks of Canonical Ltd., n.d.)

(crist, 2019)

(Ferdous, 2021)

(Parmar, 2022)

(Geeksforgeeks, 2024)

https://youtu.be/_C8kWso4ne4?feature=shared

<https://youtu.be/exUMINshRAs?feature=shared>

https://data.sfgov.org/Public-Safety/Police-Department-Incident-Reports-2018-to-Present/wg3w-h783/about_data

[https://aws.amazon.com/what-](https://aws.amazon.com/what-is/etl/#:~:text=Extract%2C%20transform%2C%20and%20load%20(,and%20machine%20learning%20(ML).)

[is/etl/#:~:text=Extract%2C%20transform%2C%20and%20load%20\(,and%20machine%20learning%20\(ML\).](https://aws.amazon.com/what-is/etl/#:~:text=Extract%2C%20transform%2C%20and%20load%20(,and%20machine%20learning%20(ML).)

<https://scaleyourapp.com/distributed-data-processing-101-the-only-guide-youll-ever-need/>

<https://www.restack.io/p/ai-data-management-certification-answer-distributed-data-processing>