Real-Time Accident Data Processing

Digital Egypt Pioneers

Agenda

- Project Overview
- Architecture
- Batch Data
- ETL for batch data
- Real-Time Data
- Streaming Data from Kafka to Spark
- Processing Data with PySpark
- Analysis
- Huawei Cloud MRS

Project Overview

The project showcases a real-time and batch data pipeline using **Docker**, **PySpark**, **Kafka**, **MySQL**, and **Jupyter**.

Key components include:

- **Docker-Compose setup** for a PySpark Jupyter environment, Zookeeper, Kafka, and MySQL.
- Simulated streaming data source for real-time ingestion via **Kafka**.
- Batch data processed in a Jupyter notebook with **Pandas**.
- An ETL pipeline for transforming and loading data into **MySQL**.
- **Power BI** Dashboards for real-time data visualization.

The project's goal is to demonstrate building a scalable pipeline for processing both real-time and batch data.

Architecture

Kafka ingests real-time streaming data; Zookeeper manages its configuration.

KAFKA AND ZOOKEEPER

PySpark processes both the real-time streaming data from Kafka and batch data.

PYSPARK

The transformed data is loaded and stored in MySQL

MYSQL

Jupyter notebooks are Used for batch processing with Pandas and SQLAlchemy.

JUPYTER NOTEBOOKS

Visualizes data from MySQL, creating dashboards for insights into accident patterns and trends.

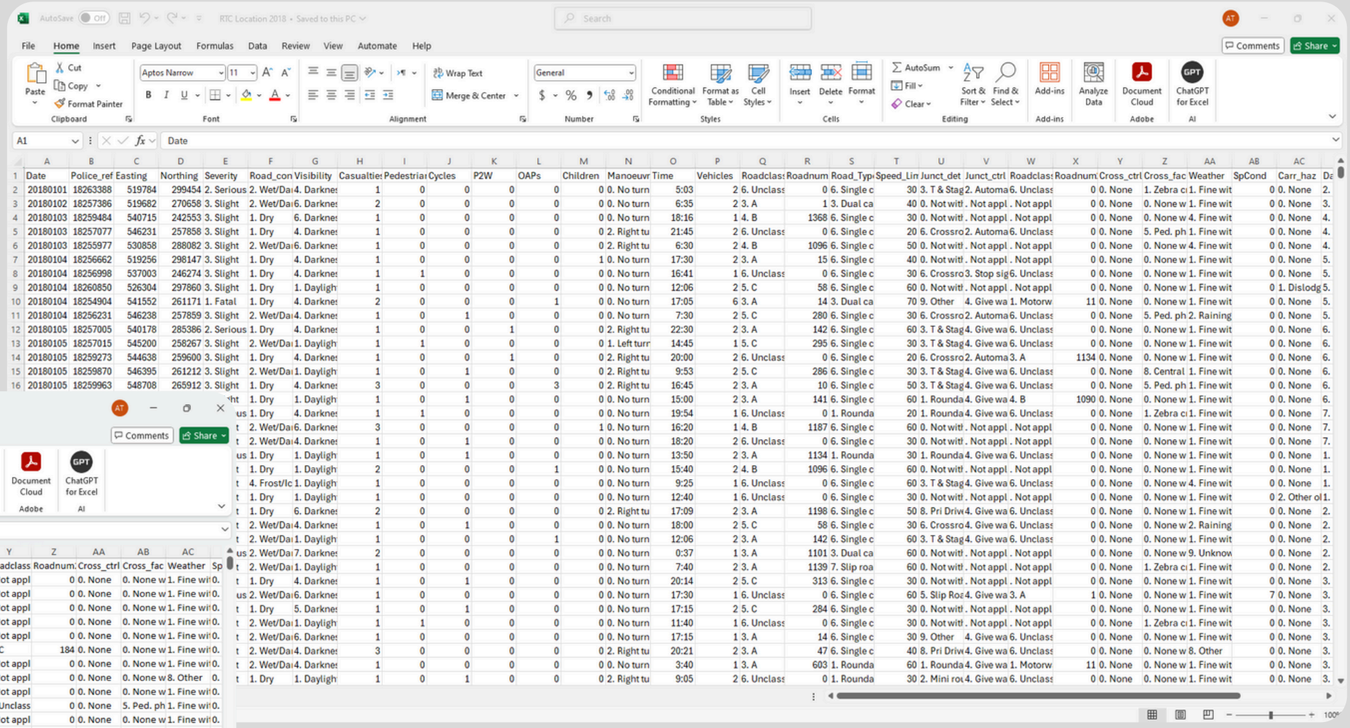
POWER BI

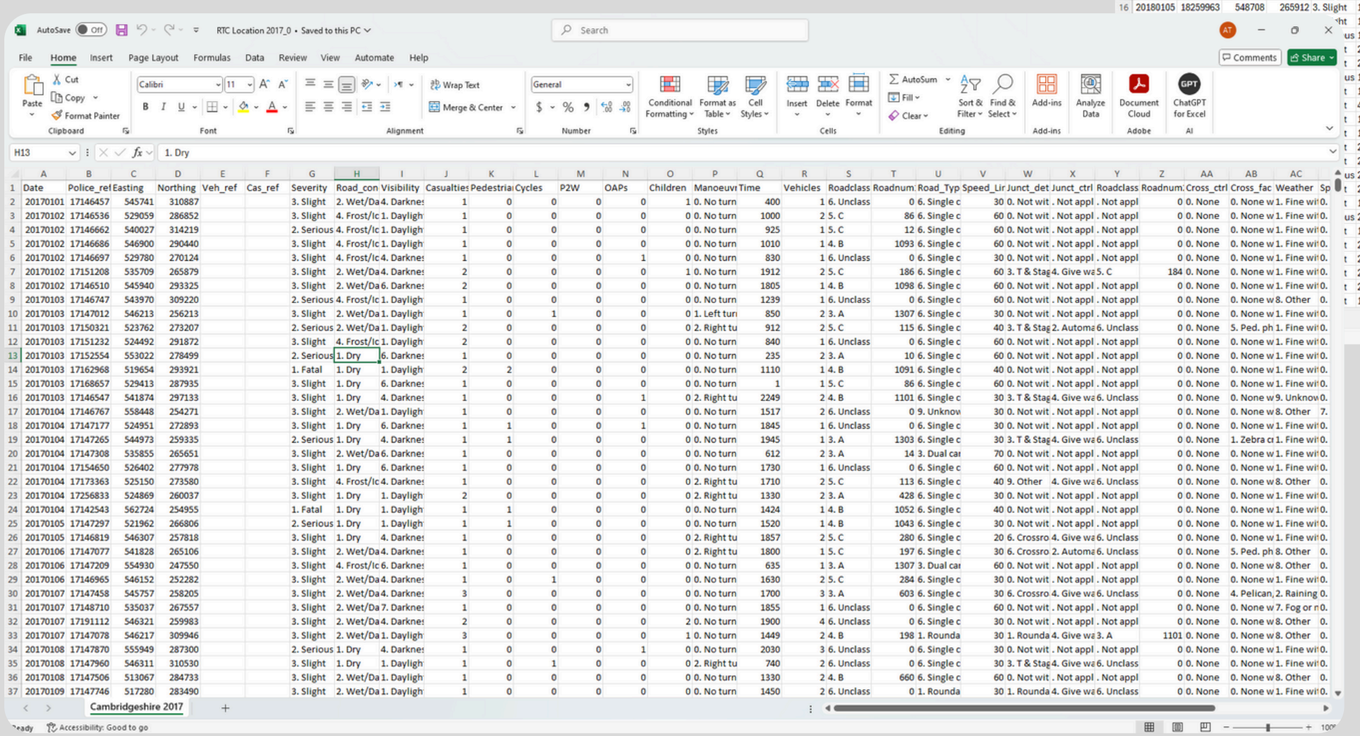
Batch Data

Digital Egypt Pioneers

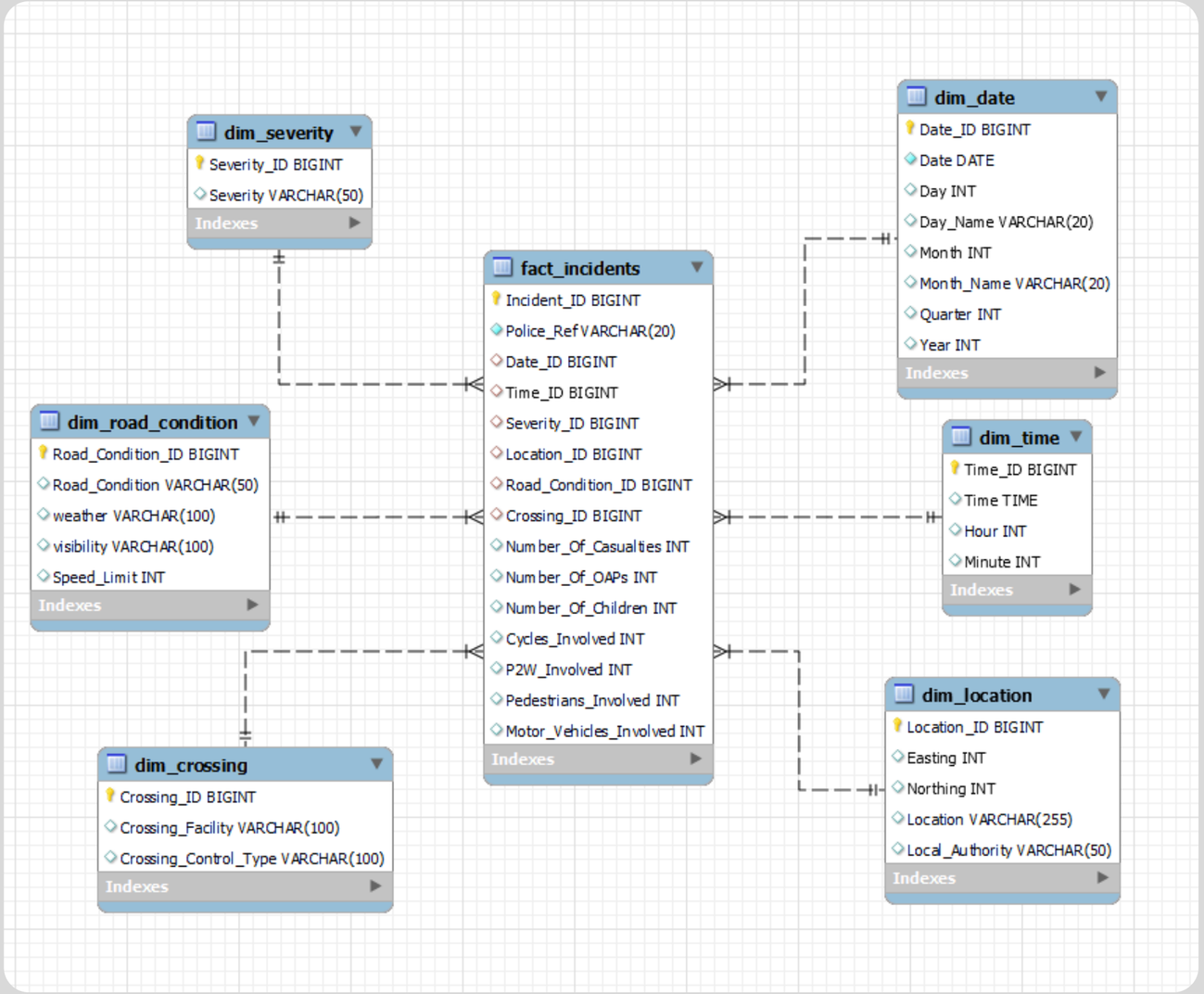
Road Traffic Collisions

Cambridgeshire County Council





Data Warehouse Schema



Why MySQL ?

- MySQL is the most widely adopted **open-source** database
- MySQL’s **fast write performance** ,especially for simple transactional operations, which is key for real-time data.
- MySQL tends to use **fewer system resources**, making it ideal for environments with limited resources

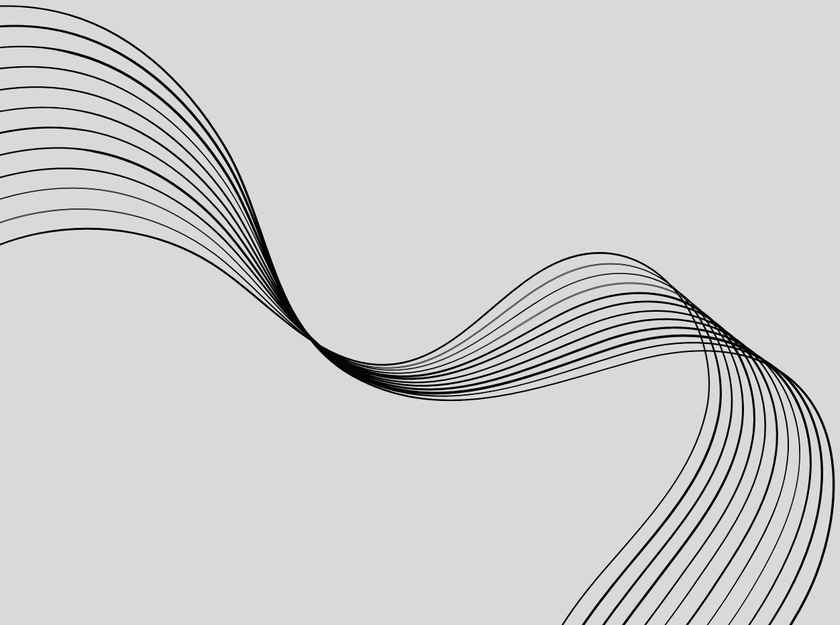
PostgreSQL can be an excellent choice too

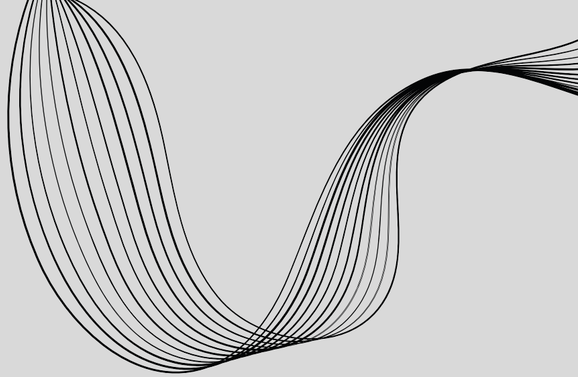
PostgreSQL

ETL For Batch Data

- **Extract**: Data is loaded from static sources using **Pandas**.
- **Transform**: Data is cleaned and preprocessed in the Jupyter notebook.
- **Load**: The batch data is inserted into MySQL using **SQLAlchemy**.

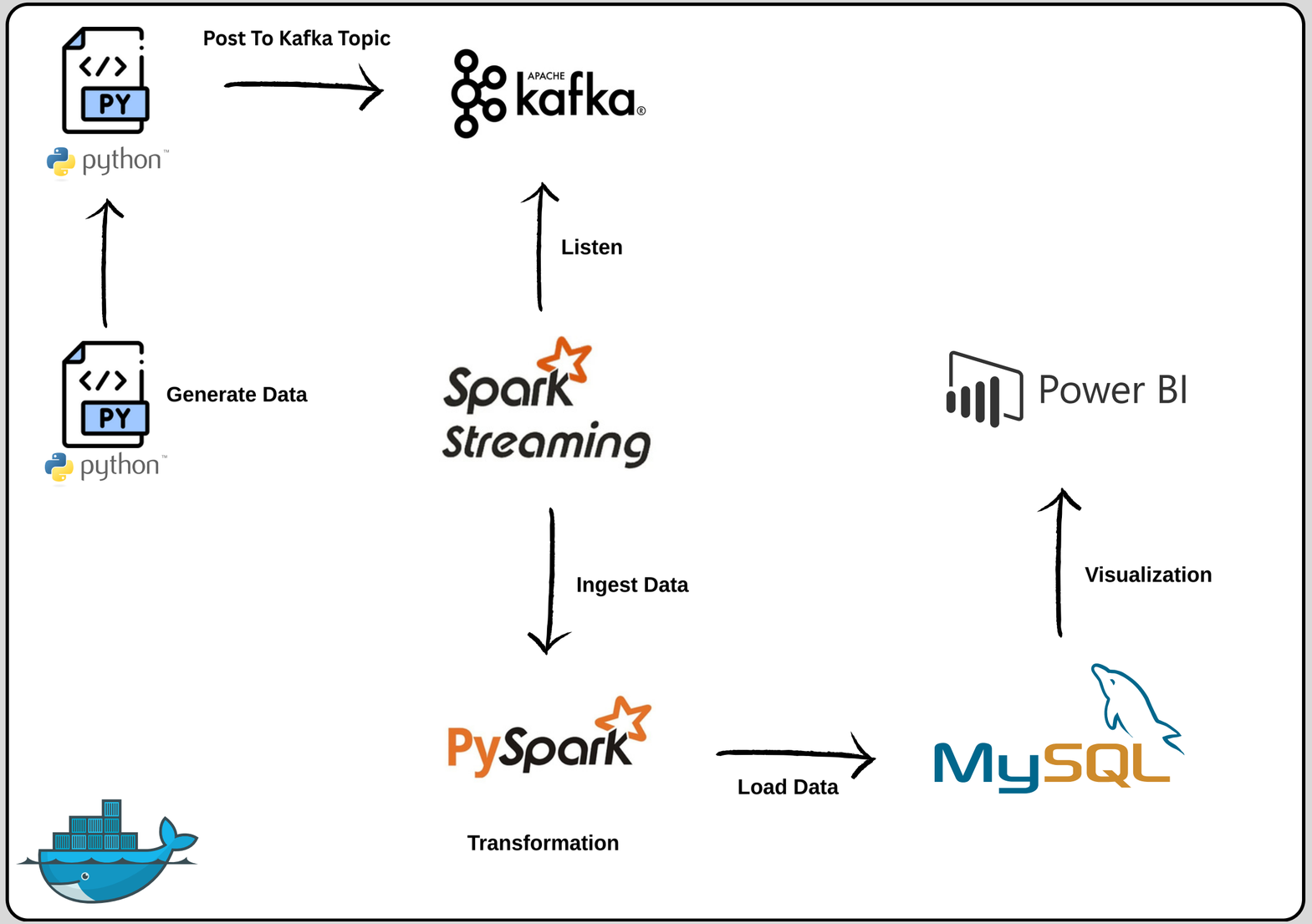
Real-Time Data





Digital Egypt Pioneers

Pipeline


Post To Kafka Topic
Listen
Generate Data
Ingest Data
Visualization
Load Data
Transformation

Streaming Data from Kafka to Spark

- Once the data is generated and posted to Kafka, our pipeline kicks into action.
- Spark Streaming seamlessly ingests real-time data from the Kafka topic.
- This allows us to process large volumes of data as they arrive, ensuring efficient data flow without latency.

Processing Data with PySpark

- With the data ingested, **PySpark** takes over, transforming the raw streams into structured, actionable data.

- We apply business rules, clean the dataset, and prepare it for storage, enabling the data to meet all analysis and reporting requirements.

Loading Data into MySQL

- After transformation, the next step is to persist the data. **PySpark** loads the processed data into a **MySQL** database.

- This provides a reliable and scalable storage solution, where the data can be queried and accessed efficiently for further operations.

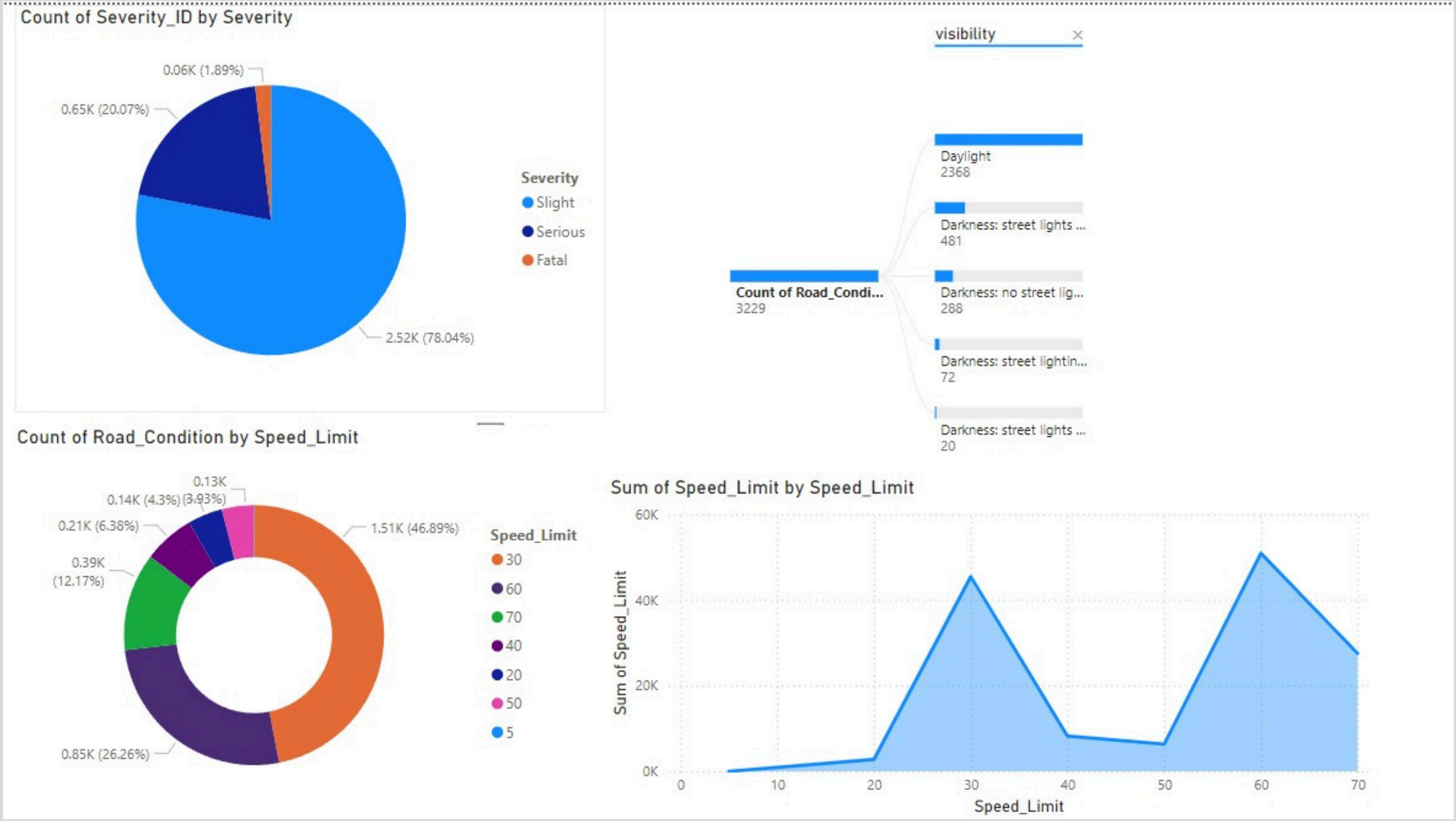
Ensuring Data Integrity and Availability

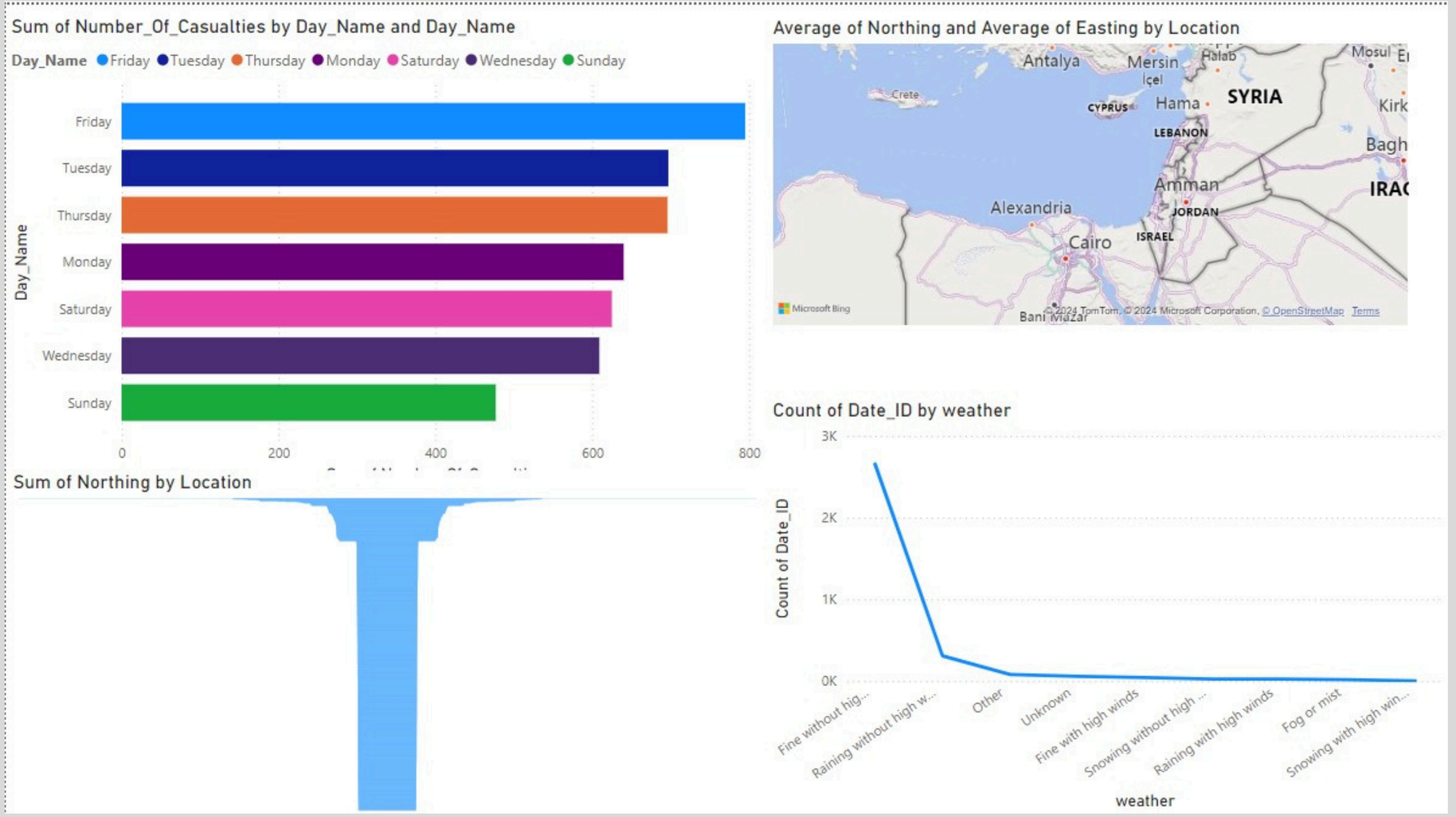
- Throughout the process, **Spark** ensures that data integrity is maintained.

- Real-time error handling and monitoring are incorporated, guaranteeing that only high-quality, clean data reaches the database.

Analysis

Digital Egypt Pioneers


Count of Severity_ID by Severity
0.06K (1.89%)
0.65K (20.07%)
2.52K (78.04%)
Severity
Slight
Serious
Fatal
visibility
Count of Road_Condi...
3229
Daylight
2368
Darkness: street lights ...
481
Darkness: no street lig...
288
Darkness: street lightin...
72
Darkness: street lights ...
20
Count of Road_Condition by Speed_Limit
0.13K
0.14K (4.3%)
0.21K (6.38%)
0.39K (12.17%)
0.85K (26.26%)
1.51K (46.89%)
Speed_Limit
30
60
70
40
20
50
5
Sum of Speed_Limit by Speed_Limit
60K
40K
20K
0K
Sum of Speed_Limit
0
10
20
30
40
50
60
70
Speed_Limit


Sum of Number_Of_Casualties by Day_Name and Day_Name
Day_Name Friday Tuesday Thursday Monday Saturday Wednesday Sunday
Friday
Tuesday
Thursday
Monday
Saturday
Wednesday
Sunday
Day_Name
0
200
400
600
800
Average of Northing and Average of Easting by Location
Antalya
Mersin
Halab
Mosul
Crete
CYPRUS
Hama
SYRIA
LEBANON
Amman
JORDAN
ISRAEL
Alexandria
Cairo
Microsoft Bing
© 2024 TomTom, © 2024 Microsoft Corporation, © OpenStreetMap Terms
Sum of Northing by Location
Count of Date_ID by weather
3K
2K
1K
0K
Count of Date_ID
Fine without hig...
Raining without high w...
Other
Unknown
Fine with high winds
Snowing without high ...
Raining with high winds
Fog or mist
Snowing with high win...
weather

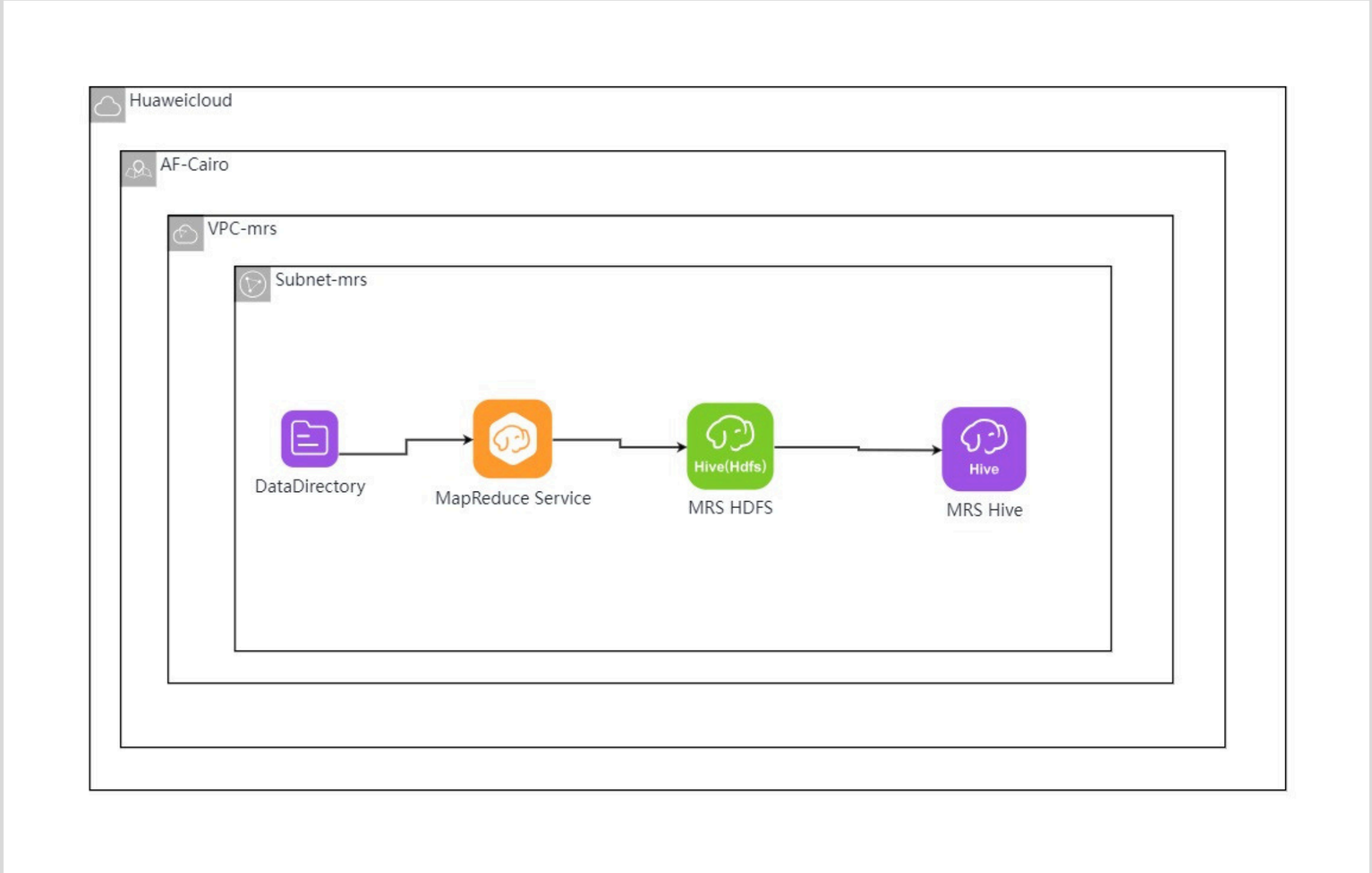
Huawei Cloud MRS

Digital Egypt Pioneers

Huawei Cloud MRS

- Huawei Cloud MRS stands for **MapReduce Service**, which provides a cloud-based big data platform.
- It’s built on top of **Hadoop** and supports tools like HDFS, Hive, Spark, Kafka, and more.
- It simplifies data storage and processing by managing everything in the cloud, which is especially useful for large-scale projects.
- With MRS, we don’t need to manage servers directly. it handles configuration, monitoring, and scaling automatically.

Architecture



- **Ingest data into HDFS**: We uploaded the raw CSV files into HDFS for distributed storage.
- **Create Hive tables**: We designed a Hive database and created multiple tables to store the raw data.
- **Load data into Hive tables**: Using LOAD DATA commands, we imported the data from HDFS into the Hive tables.
- **Build the analysis layer**: We wrote complex SQL queries to create summary tables for analysis, using the CTAS method (Create Table As Select).
- **Export to RDS**: Finally, we exported the results from the analysis layer in Hive to an RDS database on Huawei Cloud for reporting and further analysis.

MEET THE TEAM

- **Yousof Hatem**
- **Ahmed Tarek**
- **Martin Amgad**
- **Omar Akram**

Thank you!

Digital Egypt Pioneers