Automated ETL and Data Analysis using Apache Airflow Data Pipeline

Jawed Siddique
Student ID: 22232915
Master of Science in Data Analytics
National College of Ireland
Dublin, Ireland
x22232915@student.ncirl.ie

Praphul Kumar Krishna Kumar Gowda
Student ID: 22187961
Master of Science in Data Analytics
National College of Ireland
Dublin, Ireland
x22187961@student.ncirl.ie

Saket Abhaykumar Kulkarni
Student ID: 23102381
Master of Science in Data Analytics
National College of Ireland
Dublin, Ireland
x23102381@student.ncirl.ie

Abstract – *Analysing or visualizing data, building machine learning models and deep learning require data that is continuously from multiple sources using different tools and techniques. This data however cannot be used in its raw state and requires transformation before it is stored in a destination source and can be used in the future for different business use. Handling this complete process manually is time-consuming and inefficient. Hence, data pipelines are created through Directed Acyclic graphs (DAGs) that can automate and efficiently perform the complete extraction, transformation and load operation i.e. ETL. This paper aims to explore the key aspects of creating data pipelines by creating tasks and nodes and understanding the key components required to build a complete data pipeline within a workflow scheduler, namely Apache Airflow. Furthermore, it also aims to thoroughly understand the ETL process and the different tools, namely Python programming language, in-built Python libraries, MongoDB database and PostgreSQL database, that can be used to perform the three key operations. Finally, this paper also explores the analysis and visualization of the chosen datasets to answer key questions and derive key insights from the data.*

Keywords – Data pipeline, Directed Acyclic Graphs (DAGs), Apache Airflow, MongoDB, PostgreSQL, ETL, Python, Visualization.

I. INTRODUCTION

In today's world, we generate an astounding 328 million terabytes of data every day [1]. Unlike in the past, where data was majorly used to review the effects of a business decision, a reactive approach, the significance of data has evolved today and is used by industries for proactive decision-making [2]. Data, in today's time, plays a critical role in business decision-making. It is used to build and train machine learning (ML) and deep learning (DL) models, generate reports and perform predictive analysis [3]. However, it is not possible to achieve all of this without the presence of high-quality data as flawed input data would result in faulty outputs. To create excellent data products and rely on the decision that is made based on the insights provided from the data, it is crucial to have high-quality data [4]. Unfortunately, the raw data; structured, semi-structured or un-structured; that is obtained from a wide range of sources such as APIs, SQL or NoSQL databases, web scrapping, files and others cannot be used in their raw form for analysis, model-building or other data visualization tasks. To make the data suitable for business use, the sourced data usually undergoes some sort of data processing including transformations, such as filtering, sorting, derivation, normalization, masking etc. Further, the raw sourced data or the clean and transformed data might not be of immediate use and is stored pre- or post-transformation in databases, data lakes or data warehouses, so that it can be used at a later stage for analysis, visualization or machine learning [5]. These processes are usually repetitive and time-consuming. According to a survey conducted by Anaconda, 45% of a data analyst or data scientist's time is spent in data preparation including extracting and transforming the data [6]. Consequently, to make the complete process i.e. ingesting raw data from various sources, transforming it into desired formats as per the business requirement and loading it into a desired database or data warehouse; more time and cost-efficient and to streamline the process flow, data pipelines are used [7].

Data pipeline is a process where one or more datasets are modified through a series of sequential steps, where the next steps take as input the output generated in the first step, creating a pipe-like structure and having a one-directional flow [8]. Mainly there are two types of data pipelines depending on how the data is processed within the data pipeline: batch processing and streaming data [9]. Additionally, data pipelines can be created using different tools and technologies such as Apache Airflow, Apache Spark, Dagster, Informatica etc. Apache Airflow is an open-source scheduling platform designed to automate and develop scalable, dynamic, highly integrated and extensible workflows using Python programming language [10]. This is achieved by the concept of Directed Acyclic Graphs (DAGs), which represent a workflow as a series of interconnected nodes with defined dependencies. Each node here represents a specific task within the workflow [11].

This project aims to develop data pipelines by using Apache Airflow. The objective of selecting this project is to automate the ETL operation i.e. extract raw data, transform it into a usable format, load it into a designated destination and reduce the manual effort required to perform this process. This process is crucial in handling and refining data before it can be used for analysis, and as mentioned above is usually a repetitive and time-consuming process. This project aims to

use Apache Airflow workflow scheduler to simplify this process through automation making it more efficient. Additionally, after the prepared and transformed data is stored in the destination database, the project aims to analyze the data and answer questions on various domains such as electric vehicles, public safety, and health.

II. Related Work

Several recent studies have been conducted in the fields of electric vehicles, public safety and health as these domains highly impact society and the climate which are major areas of concern for the modern world. The research conducted by Joel Jaeger titled *'Electric Mobility'* [12] et al. (2023), showed 10% of the passenger vehicles sold globally were all electric which is 10 times more than it was 5 years prior. In terms of global percentage sales, countries like Norway (80%), Iceland (41%), Sweden (32%), Netherlands (24%), and China (22%) are at the top in sales of passenger electric vehicles. A survey conducted in 2019 in Ireland by the Central Statistics Office focused on electric vehicles covering a vast age group came up with key findings like; hybrid cars were the most common type of EV where out of the population that already owns EVs, 57% own hybrid cars and that is the most preferred. 40% of the people would consider purchasing an EV and the reason influencing the purchase was better affordability to run [13]. A study conducted on the NYC crime and arrest data by Ali Bauman et. al. (2023) showed a decrease in the overall crime and arrests made in 2023 [14]. Additionally, ever since the COVID-19 pandemic hit the world, multiple studies have been conducted to understand the reasons and its impact. One such study conducted by P. Oboza et. al. (2023) on the main cause of death in patients due to COVID-19 showed thrombotic changes in the microcirculation [15]. Additionally, data pipeline development, deployment and management have also been a topic of multiple studies such as the study *'Data Pipeline Management in Practice: Challenges and Opportunities'* conducted by Aiswarya Raj Munappy, Jan Bosch, and Helena Homstr¨om Olsson discusses the challenges and opportunities in developing data pipelines [3].

III. Methodology

The architecture of data pipelines; which is a series of automated and sequential steps aimed at completing the ETL i.e. extract, transform and load operation; has five core components [5] as shown in Fig. 1 below:

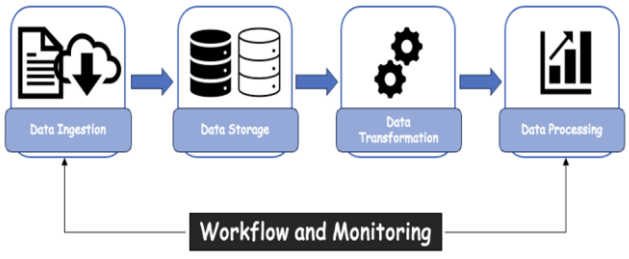

Figure 1: Components of Data pipeline

i. Data Ingestion: This is the process of gathering and collecting data from various sources.
ii. Data Storage: The raw data taken from the ingestion stage, is then stored in places like data warehouses, data lakes, databases or distributed file systems depending on various factors such as volume, scalability, cost and other considerations.
If not of immediate use, the clean data obtained after the transformation component can also be stored in these places.
iii. Data Transformation: This process includes tasks like cleaning, preparing and modifying the data for future analysis.
iv. Data Processing: During this stage, meaningful insights are derived from the transformed data.
v. Data Delivery: In this stage the processed data is delivered to the stakeholder as per the business requirement.

There are multiple tools and technologies available that can be used to build each component of the data pipeline. Additionally, the components can follow different orders depending on the business requirement, such as ETL or ELT processes. Highlighting the components, the below Fig. 2, shows an overview of the process flow followed and the tools used to develop the data pipeline for this project.

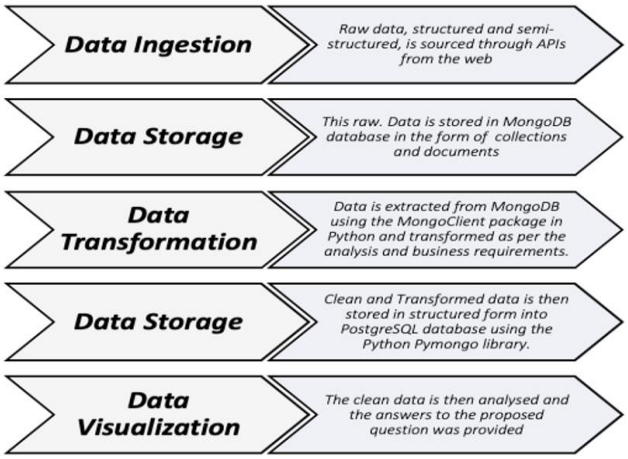

Figure 2: Tools used for each component in the Airflow data pipeline

In this project, one structured dataset in CSV format and two semi-structured in JSON format have been used to develop three unique data pipelines, giving different analyses and results. The following section discusses in detail the essential components of the developed data pipeline - ingestion, storage, transformation and processing – for each of the individual datasets. The project revolves around developing a data pipeline utilizing Apache Airflow to Automate the process. A method in which raw data is ingested from the web and then ported to the data store for analysis. Before data flows into a data repository, it usually undergoes some data processing [16]. By using Apache Airflow and calling the Python operator that is available in airflow, DAG (Directed Acyclic Graph) files can be written to call each piece of code together and make the process automated. The DAG files here are a key as it runs the collection of tasks assigned in the code.

A. Data Pipeline and ETL for Electric Vehicles population analysis:

Electric vehicles (EVs) are the game-changing component in this generation's vehicle industry. By saving fuel as well as money, EVs emerge as one of the best travel methods. They are by far the most efficient vehicles, and there are numerous reasons why one should own an electric vehicle. Many countries are actively pushing to become as electric as possible. Norway, currently leading in this transition, is cited

as an example [17]. Other countries, such as China, Iceland, Sweden, and the Netherlands, are also on the list for increasing EV usage. The objective of the research is to perform an analysis of the population of EVs and answer four research questions:

Q1. Which city in Washington state has the most population of Electric vehicles and the top 15?
Q2. Which are the top 15 models and which Model is the most in population?
Q3. Which are the top 3 Electric vehicle models for each year from 2014-2024?
Q4. Among the top 5 models, which model has the highest range?

The categories in focus are Battery Electric Vehicles (BEVs) which are solely operated on battery and no gas engine parts, and Plug-in Hybrid Electric Vehicles (PHEVs) which consist of both electric charging as well as gas engine parts as the name suggests it's a hybrid of both. These vehicles are currently registered through the Washington State Department of Licensing (DOL) [18].

Sourcing:

The data is fetched from the US government data website which is an open-source dataset platform containing all types of data related to the federal government of the United States [19]. The data contains important information such as the vehicle models, vehicle license ID and range of the electric vehicles, and all these vehicles are registered under the Washington State Department of Licensing. The dataset is in JSON format and is sourced by taking the API from the web and sourcing the file. It is called by using the "requests" library in Python, which is the most important tool in sourcing the JSON file via API by making HTTP requests for the data. The data contains "metadata" and the "data":

- *'Metadata'* consists of the view section where the data description, the ID for the data, the creation of the data, and other generic information about the dataset. It is important to note that parsing the metadata is required to be able to get some form of structured data for storage in a database.
- *'Data'* consists of the actual data that will be dealt with in the project with all the rows present in this section which is the main information needed for the entire process. This data is extracted by relating it with the columns in the metadata and extracting the entire file as one.

Storage:

Storage of the data after souring it from the API is done by pushing it into MongoDB, which is a NoSQL database that means Not only SQL. MongoDB stores the data in JSON-like files called BSON. It is scalable as it allows storage of very large files allowing good representation of complex data. While storing the data for this project, MongoDB was accessed using the Python programming language. The Python library *'Pymongo'* is used for the manipulation of MongoDB by establishing a connection between the Python environment and 'MongoClient' which is an attribute of Pymongo allowing the connection via the connection string that is provided after hosting MongoDB on the machine. After the connection is successful, a database is created manually by calling the client from Pymongo, and then a collection in the database is created which is where the data will be stored. Before storing the data, it must be converted to dictionary format and oriented as records so each element of the data will be separated and displayed as single records. Along with that, the 'Pandas' library in Python is used for data manipulation and storage into data frames which makes access to the data very convenient.

Transformation & Preprocessing:

After loading the data into MongoDB, the records are stored as individual elements. It is necessary to bring this data into a structure to be able to perform further analysis create visualizations according to the proposed questions and gain insights from the data. The same procedure of connecting to the MongoDB database is followed and then the database with the records is accessed to pull the data into a data frame using Pandas library. The data is extracted from the collection using the Pymongo library and its attributes which help in locating the data and which is in turn stored in a data frame after converting the raw data into a list. This data frame can be used to perform transformations on the data according to the requirements in the further parts of the analysis.

- **Column Transformation:** After extraction, the data frame containing all the records as a list is transformed, it was observed that there were unnecessary columns like the id columns and other metadata columns which are of no use in the analysis or even reading the data. This kind of extra data is usually information in the JSON file as it is semi-structured data and requires more relation between the elements in its formatting. While structuring the data, only the columns containing the most important information are required. In this case, the important columns in consideration are the Model, State, Electric Vehicle Range, and Model Year. Other than these some more columns have not been dropped as they do have a relation with the electric vehicles. A total of 28 columns were extracted from the raw data and after processing, the remaining columns are 16.
- **Data Type Conversion:** After the removal of unwanted columns and other transformations, it was observed that all the data types of the data had become object types which makes it difficult to manipulate. Therefore, after checking the columns with integer values, those columns are converted to integers.
- **Null Values:** The next step in the data transformation process is to remove null values. While performing this it came as a surprise that the column with the electric range of the vehicle had a lot of null values, therefore, those must be removed as it would create bad outputs. After removing these null values, the data contains an electric range for all the vehicles.
- **Model Year and Electric Rage Transformation:** In this case, the focus is to analyze the EV population the recent years, however, the data contains information about the models from the year 1997 to 2024. Based on the purpose of this analysis, it was decided to eliminate the years 1997 to 2013 which did not have a high number of records. Dropping these records would not result in a high data loss and will make the analysis objective focused on the population of EVs in the past 10 years.

Fig 3 below gives information about the transformed data and the effect of the transformation steps on the raw dataset as per the defined business and analysis needs.

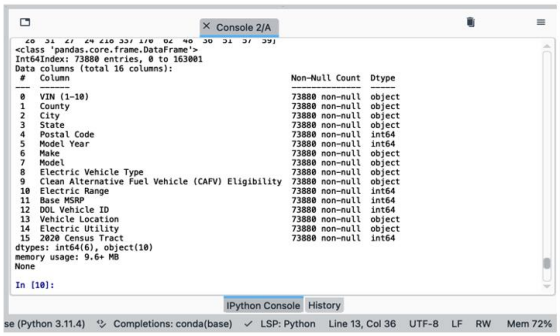

Figure 3: Information about the data after the transformation process

After the extraction and transformation of the data, it is well-structured and ready for any analysis that needs to be performed on it. To access the structured data, it needs to be stored in a database that accepts structured and formatted data. This was done using the PostgreSQL database, which is a relational database platform for structured data, and it stores the data in the form of tables in the created database. To perform this task, *'psycopg2'*, *'sqlalchemy'*, and *'urllib. Parse'* are the libraries required. These libraries act as the link between Python and PostgreSQL. After creating a PostgreSQL database, the credentials are used in the code to connect it to the Python environment using the connection string. Python authenticates the credentials, focusing mainly on the password, and then completes the connection, allowing the user to load data in the form of a table into the database. Finally, another data frame is used to declare the filtered data from the transformation section of the code. That data frame is converted to SQL format and a table name is given according to the user which in this case is *'EV_Clean_Transformed_Data'*. This table is now loaded into the PostgreSQL database and can be extracted using Python to perform the final analysis and visualization of the data it contains all the important information. This will be used to answer the proposed questions in the results and evaluation section (see section IV (A) below).

B. Data Pipeline and ETL for New York City YTD (2023) arrest analysis:

Ensuring public safety by keeping a secure environment is the key responsibility of law enforcement agencies. Considering the impact and importance of creating a safe and secure community, for this project an analysis is performed on the New York Police Department (NYPD) arrest data. The purpose of this study is to understand law enforcement patterns, identify trends, and potential areas of concern and answer the following questions:

Q1. For which category were the most arrests made?
Q2. Most arrests were made from which of the five New York City Boroughs.
Q3. Analysis of the age of the offenders.
Q4. Based on the available data (YTD) perform a time series analysis on the month and week of the arrest.

Sourcing:

The raw NYPD arrest data (year to date) in CSV format is sourced programmatically from the publicly available *'New York City Open Data'* repository that contains data covering a wide range of departments within New York City. The sourcing is done using the Socrata Open Data API (SODA) which is an easy platform to access public datasets. Further, to access the 'Socrata' library from Python, the 'sodapy' package has been used. The dataset contains a breakdown of all the arrests made in New York City by the NYPD during the current year. The dataset has approximately 170K records and each record is captured along 24 fields such as arrest date, age group, offence committed, arrest precinct etc.

Storage:

After the data is sourced, using another Python package 'Pymongo', this data is stored in MongoDB, which is a NoSQL database. In this database, the data is organized into collections and each record from the sourced data is stored separately in individual documents. The database can then be further connected using Python and the individual documents can then be retrieved for the next step in the workflow.

Transformation & Preprocessing:

The next step in the data pipeline is to convert the raw into a format that is suitable for analysis. By using Python programming language, the sourced raw dataset that was stored in the MongoDB database is extracted and the following transformations and preprocessing steps have been performed to address the data quality issue and ensure the data is clean and accurate before it is loaded to the destination system (in this case PostgreSQL database) that can be used further as per business requirements.

- **Renaming and Filtering Columns:** A few column names do not match the data dictionary and the header information available on the source website. Thus, to maintain consistency, these column names have been renamed. Further, columns *'_id', 'pd_cd' and 'pd_desc'* have repeated information and columns *'x_coord_cd', 'y_coord_cd', 'geocoded_column', 'Community_Dist ricts' 'LAW_CODE', 'Borough_Boundaries', 'City_Council_Di stricts' 'Zip_Code s', and 'Police_Precincts'* are not of interest for this analysis, thus these columns have been filtered out from the data.
- **Formatting:** The loaded data from the MongoDB database has an object datatype that can cause performance and operational challenges in the next transformation steps. To address this issue, in this step, the data type of the variable is converted according to the information present in the data dictionary and the type of data stored in it. For example, for column *'arrest_date'* the datatype is changed from object to datetime.
- **Sorting:** The data extracted from the source is not sorted and, in this step, sorting on the dataset is performed according to the *'arrest_date'* that will help during the analysis and visualization steps of this project.
- **Deduplication:** Duplicate records in the data can skew the output results and thus need to be removed from the data. During this transformation stage, a check for duplicate records was performed and no such records were found in our dataset.
- **Data Cleansing:** By conducting tests through Python missing values such as 'NaN', 'N/A', and blank entries, were found in the dataset. In the transformation process, these values have also been removed so that our analysis provides valid outputs.
- **Data Validation:** According to the official data dictionary, specific columns cannot accept arbitrary values and must follow pre-defined values. Based on the data review, certain values do not adhere to these values and thus have been removed from the dataset as part of

the data transformation process, before performing the analysis. Column ‘ky_cd' has values as zero, however, this column describes the offence code for which the arrest was made and having a value of 0 does not provide much context to our analysis and thus such records are dropped from the dataset. Offences that fall under the category '9' are not under the scope of our analysis and thus have been removed from the data.

- **Data Derivation:** For columns 'law_cat_cd', 'arrest_boro', ‘jurisdiction_code’ and ‘arrest_date’, meaningful codes were provided which would help in the analysis and visualization of the data. This would help in providing reasonable insights into the analysis questions proposed for this dataset.

Fig 4 below gives information about the transformed data and the effect of the transformation steps on the raw dataset as per the defined business and analysis needs.

```
[2023-12-16, 20:22:45 UTC] {logging_mixin.py:154} INFO - No of duplicated records = 0
[2023-12-16, 20:22:45 UTC] {logging_mixin.py:154} INFO - arrest_key              0
arrest_date              0
ky_cd                    0
ofns_desc                0
law_cat_cd            1257
arrest_boro              0
arrest_precinct          0
jurisdiction_code        0
age_group                0
perp_sex                 0
perp_race                0
latitude                 0
longitude                0
dtype: int64
[2023-12-16, 20:22:46 UTC] {logging_mixin.py:154} INFO - Index(['arrest_key', 'arrest_date', 'ky_cd', 'ofns_desc', 'law_cat_cd',
       'arrest_boro', 'arrest_precinct', 'jurisdiction_code', 'age_group',
       'perp_sex', 'perp_race', 'latitude', 'longitude', 'Law_Codes_Category',
       'Arrest_Boroughs_Category', 'Jurisdiction_Code',
       'JurisdictionCode_Category', 'Arrest_Month', 'Arrest_Week'],
      dtype='object')
[2023-12-16, 20:22:46 UTC] {logging_mixin.py:154} INFO - (168825, 19)
```

Figure 4: Data and column info after the transformation process

After the raw data is extracted from MongoDB and cleaned as well as transformed *‘NewYorkArrest_YTD_Clean_transformed_Data’* as per the requirements of this analysis, the data is stored in PostgreSQL from where it can be used for multiple purposes such as analysis, building machine learning models and visualization. PostgreSQL is a type of SQL database that is predominantly used to store relational and structured data. Post this, the data has been used to derive meaningful insights and answer the proposed questions which are discussed in the next section (see section IV (B) below).

C. Data Pipeline and ETL for death involving COVID-19 analysis:

The objective of this project centres around creating a data pipeline to extract, transform and store data for the analysis detailing health conditions and contributing causes associated with deaths involving COVID-19. The objective is to extract insights into COVID-19 mortality patterns, emphasizing age groups and jurisdictions of occurrence. By focusing on this dataset, the analysis aims to inform public health strategies and policies by uncovering trends and critical factors related to the pandemic. To achieve these goals, the analysis tackles two key questions:

Q1. What patterns emerge in the number of mentions related to COVID-19, and how do these mentions correlate with the actual reported COVID-19 deaths?
Q2. What insights can be gained from analyzing the reported COVID-19 deaths, particularly looking at conditions, state, year and age group in comparison?

The achievement of these goals is orchestrated through the coordination of code blocks that can be integrated into a directed acyclic graph (DAG) for automated execution. This strategic approach ensures efficiency in data processing and sets the stage for insightful analyses that can inform evidence-based decision-making in public health.

Sourcing and Storage:

During this stage, a series of operations for fetching, processing, and storing COVID-19-related data from the Centers for Disease Control and Prevention (CDC) using a Python script is performed. The initial steps involve making an API to retrieve raw data. The CDC API was chosen due to its direct relevance to COVID-19 data, providing a reliable and up-to-date data source for analysis. The retrieved data, which is in JSON format, is then converted into a Pandas data frame, allowing for convenient manipulation and analysis. This data frame is further subjected to preprocessing steps such as filtering out rows where the *‘Year’* column is NaN (not available) and excluding entries related to *‘Circulatory diseases’* or *‘Respiratory diseases’* under the *‘Condition Group’*. This refined dataset is subsequently converted into a list of dictionaries, with each dictionary representing a row of data. Later using Python code an interface was created with MongoDB, a NoSQL database, to establish a connection to the local MongoDB instance. The MongoDB database was selected for its flexibility in handling JSON-like documents and aligning well with the unstructured nature of the retrieved COVID-19 raw data. The specified or created database, named *‘Covid_Contribution’* becomes the target for storing the processed data. The MongoDB client is instantiated, the desired database is accessed, and the connected database is displayed. Finally, the processed data is inserted into the 'conditions' collection within the MongoDB database using the insert_many method. as shown in Fig. 6 below.

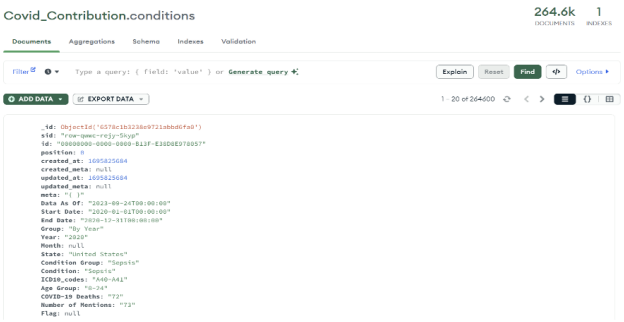

Figure 5: MongoDb database with raw stored data

Transformation & Preprocessing:

The goal of this step is to process and analyze the COVID-19 data that was previously sourced by interacting with two databases: one MongoDB and the other PostgreSQL. It first establishes a connection to a MongoDB database called *‘Covid_Contribution’*, accesses the database *‘conditions’* collection, and carries out several retrieval tasks. From the *‘conditions’* collection, it retrieves all documents and all documents stored in a list. Then, extra columns are removed, and these documents are transformed into a Pandas data frame. After certain columns are converted to integer data types and the data frame is inspected for missing values, it is displayed. The data frame is also altered to remove the '_id' column as it is not required for our analysis.

The transformed data is then ready to be stored in a PostgreSQL database which has robust relational data storage capabilities and full SQL support which is ideal for structured data storage in a database table called

'COVID19_Clean_Transformed_Data' from where the clean and transformed data is used to analysis and answer the proposed questions (see section IV(C) below).

IV. RESULTS & EVALUATION

Following the creation of Directed Acyclic Graphs (DAGs) to perform the task of extraction of the three sets of raw data using APIs, storing the data in its raw state in the MongoDB database, subsequently extracting and transforming this raw data using built-in Python packages and libraries, such as MongoClient, and loading the final clean and transformed data into PostgreSQL; the clean and transformed data was analyzed using additional python packages and provide answers to the proposed questions using visualization techniques. This section aims to answer and provide the results from the analysis and visualization.

A. Analysis of the Electric Vehicles population

Q1. Which city in Washington state has the most population of Electric vehicles and the top 15?

To answer this question, analysis and visualizations are done concerning the State column and the count of the states is taken, and the top 15 results are shown in Fig. 6 and Fig. 7 below.

```
Seattle          12833
Bellevue          3270
Vancouver         2712
Redmond           2355
Kirkland          2101
Olympia           1949
Renton            1924
Bothell           1909
Sammamish         1838
Tacoma            1615
Bellingham        1386
Tukwila           1332
Spokane           1181
Mercer Island     1026
Issaquah          1015
Name: City, dtype: int64
```

Figure 6: Top 15 states with EV population

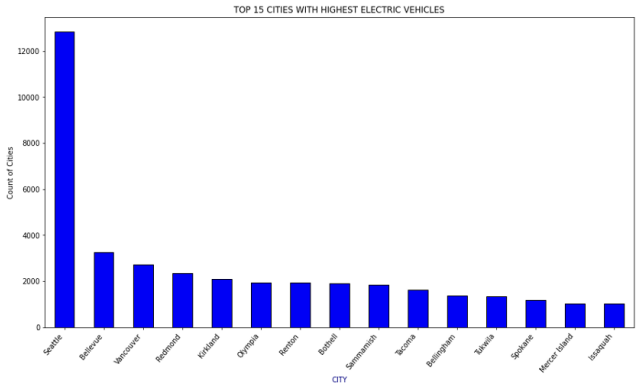

Figure 7: Bar chart of the top 15 states with the highest EV population

Based on the findings, Seattle leads the way in terms of EV population by having approximately 30% of the total EV population. The other 14 states also have a decent number from which it can be concluded that these states are pushing to grow in terms of EV use.

Q2. Which are the top 15 models and which Model is the most in population?

To find the top 15 models an analysis is performed on the Model column containing all the models in this population. Below are the top 15 models that belong to the total population.

```
MODEL 3          13887
LEAF              7749
MODEL S           5206
BOLT EV           3916
VOLT              3534
MODEL X           3288
WRANGLER          2900
PACIFICA          2859
PRIUS PRIME       2569
MODEL Y           2255
NIRO              1899
I3                1880
RAV4 PRIME        1706
X5                1679
FUSION            1622
Name: Model, dtype: int64
```

IPython Console History

(Python 3.11.4) Completions: conda(base) LSP: Python Line 74, Col 19

Figure 8: Top 15 electric vehicle models

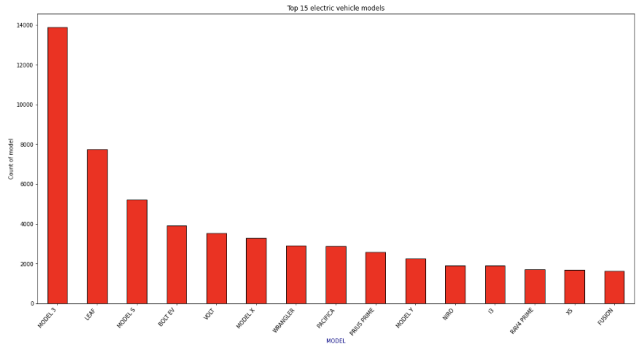

Figure 9: Bar chart of Top 15 electric vehicle models

According to Fig. 8 and Fig. 9 above, Model 3 has the highest number in the population. Looking at the visualization provided in Figure 4, it can be said that there is room for an increase in the number of EVs in the future as most of the models have an average number.

Q3. Which are the top 3 Electric vehicle models for each year from 2014-2024?

By grouping the Model year and the Model of the vehicles and then counting the elements to sort i.e. the year is sorted in ascending order and the model count is sorted in descending order. The output below shows a clear visual of the years, the models, and the number of models in each year.

```
count    73027.000000
mean      2018.769332
std          2.474866
min       2014.000000
25%       2017.000000
50%       2019.000000
75%       2020.000000
max       2024.000000
Name: Model Year, dtype: float64
     Model Year       Model  Count
13         2014         VOLT    664
10         2014      MODEL S    662
9          2014        LEAF     648
22         2015        LEAF    1813
23         2015      MODEL S    1089
26         2015         VOLT     369
38         2016        LEAF    1137
39         2016      MODEL S     970
40         2016      MODEL X     652
67         2017         VOLT    1425
48         2017      BOLT EV    1372
58         2017        LEAF     917
81         2018      MODEL 3    6104
80         2018        LEAF    1218
82         2018      MODEL S     906
110        2019      MODEL 3    3975
109        2019        LEAF    1386
97         2019      BOLT EV    1001
140        2020      MODEL 3    3741
143        2020      MODEL Y    2255
128        2020      BOLT EV    1051
184        2021   RAV4 PRIME     802
188        2021     WRANGLER     631
177        2021     PACIFICA     511
218        2022           X5     571
217        2022     WRANGLER     472
211        2022   RAV4 PRIME     396
246        2023     WRANGLER    1509
232        2023     PACIFICA     868
247        2023           X5     545
262        2024     WRANGLER     288
251        2024        CX-90     269
263        2024           X5     251
```

IPython Console History

(Python 3.11.4) Completions: conda(base) LSP: Python Line 82, Col 30

Figure 10: Bar chart of Top 15 electric vehicle models

From the output in Fig. 10, the following observations are derived:

- 2014 and 2015 have similar models in population. The numbers of the models vary by count.
- Between 2016 and 2020, Model 3, Model S and Leaf have increased in population.
- Between 2021 and 2024, some new models came into the frame like the Wrangler, X5, and RAV4 Prime.

Q4. Among the top 5 models, which model has the highest range?

To answer this question, the model and the range of the vehicle were related after sorting the data into descending order so that the highest ranges show up first. Also, the other columns are removed and only the model along with the range is kept. After this a comparison was done on the output and as seen from Fig. 11, model Y which was not part of the most popular electric vehicle turns out to be the one with the highest range, and the same applies to KONA.

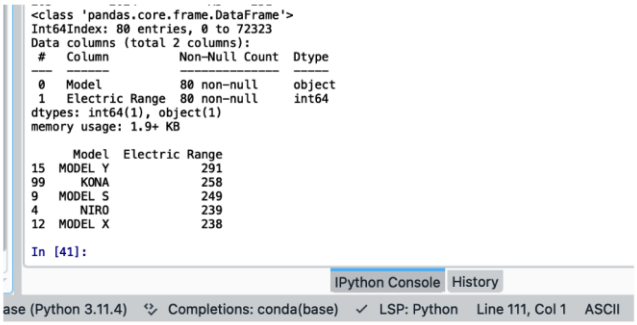

Figure 11: EV models with highest range

B. Analysis of the New York City YTD arrest data

Q1. For which category were the most arrests made?

Considering the 4 major categories for arrest, misdemeanour has the highest number of arrests with 93347 counts. The second highest number of arrests was for felonies having a count of 73853. These two offences make more than 99% of the arrests made in New York City YTD. Fig. 12 below shows a visual representation of these values.

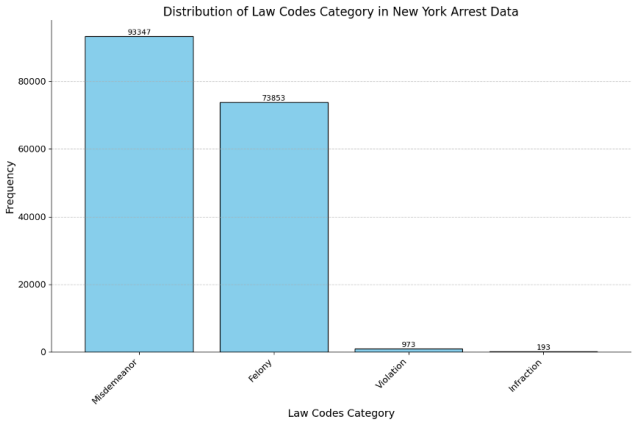

Figure 12: Category of most arrest

Q2. Most arrests were made from which of the five New York City Boroughs?

For the five boroughs in New York City, Brooklyn, Bronx, Manhattan, Queens, and Staten Island, the highest arrests were made in the borough of Brooklyn. A similar number of arrests were made in the Bronx, Manhattan and Queens. Fig. 13 below represents these values.

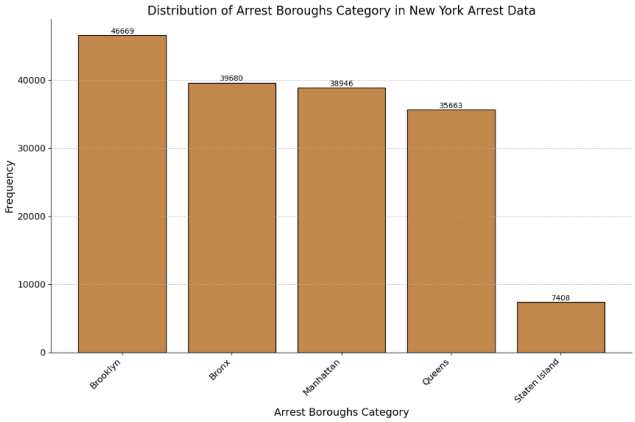

Figure 13: Arrests made by Boroghs

Q3. Analysis of the age of the offenders?

To understand if there is any relation between the arrests made and the age of the offender an analysis was performed. From the findings of the analysis, as can be seen below in Fig. 14, more than 55% of the offender's age was between the age group of 25-44 years.

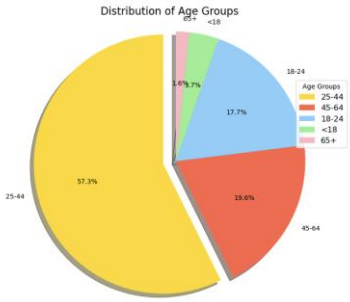

Figure 14: Offenders age group

Q4. Based on the available data (YTD) perform a time series analysis on the month and week of the arrest.

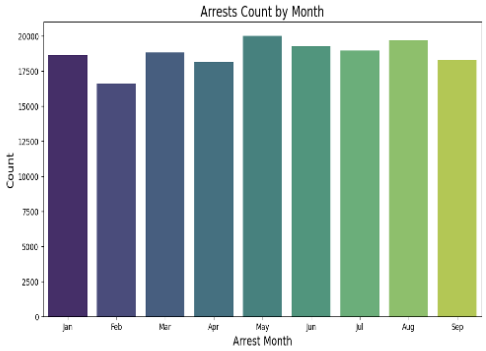

Figure 13: Arrests by months

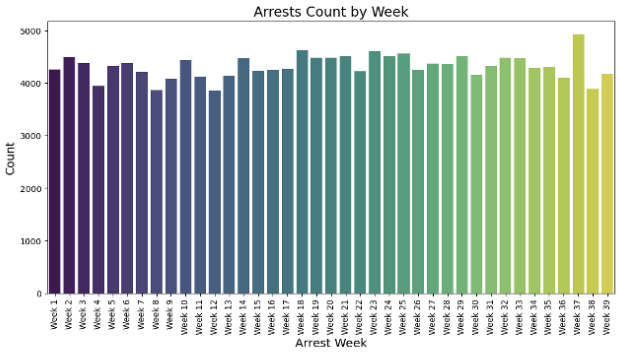

Figure 14: Arrests by weeks

From Fig. 13 and Fig. 14 above it can be noted that the number of arrests made has remained consistent month-on-month as well as week-on-week, with very minor variations. The highest number of arrests made is for March and August, while the highest number of arrests was made in Week 37.

C. Analysis of the COVID-19 deaths

Q1. What patterns emerge in the number of mentions related to COVID-19, and how do these mentions correlate with the actual reported COVID-19 deaths?

Q2. What insights can be gained from analyzing the reported COVID-19 deaths, particularly looking at conditions, state, year and age group in comparison?

The heatmap in Fig. 15 illustrates the total number of COVID-19 deaths for various conditions over the years.

- *Comparative Impact of Other Conditions:*
  'Malignant neoplasms' (cancer) consistently contributes to a substantial number of deaths each year, showcasing its persistent impact. 'All other conditions and causes (residual)' and 'Diabetes' also exhibit notable numbers of deaths each year.
- *Variation in Specific Conditions:*
  Conditions like 'Obesity', 'Renal failure', and 'Sepsis' show varying degrees of impact across the years, with fluctuations in death counts.
- *Emerging Trends:*
  'Alzheimer's disease' and 'Intentional and unintentional injury, poisoning, and other adverse events' exhibit relatively stable patterns with moderate numbers.

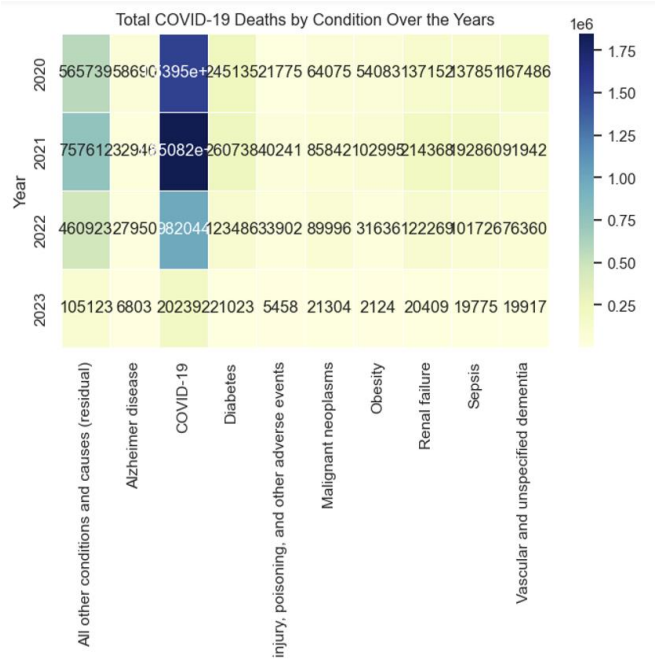

Figure 15: COVID-19 deaths by condition

Fig. 16 shows that there is a significant increase in COVID-19 deaths from 2020 to 2021, peaking in 2021, and a decrease in 2022 and 2023.

- *State-wise Analysis (2020-2023):*
  California, Texas, and Florida: These states consistently report the highest total COVID-19 deaths across all years, indicating a substantial impact of the pandemic.
  Alaska and Vermont: Smaller states like Alaska and Vermont consistently exhibit lower total death counts, reflecting potential differences in population density and healthcare infrastructure.
- *Yearly Trends:*
  2020: The initial year of the pandemic saw varying death counts across states, with larger states bearing a significant burden.
  2021: Total deaths increased in 2021, with larger states continuing to report higher numbers.
  2022: A decline in total deaths is observed in 2022, suggesting potential improvements or changes in the pandemic situation.
  2023: The trend of decreasing total deaths continues, indicating ongoing efforts to manage and control the impact of COVID-19.

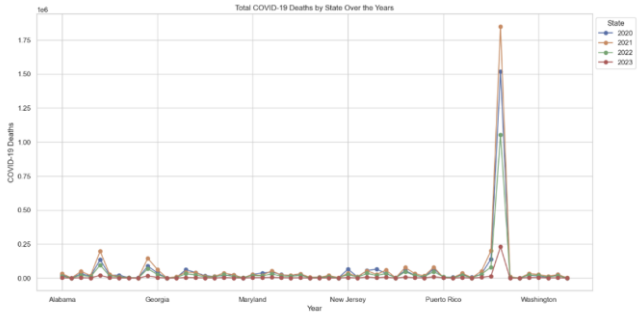

Figure 16: COVID-19 state-wise deaths by year

- *Considerations:*
  Population Disparities: Larger states with higher populations naturally report more deaths, but per capita rates might reveal different insights.
  Public Health Measures: Variations in state responses, healthcare systems, and public health measures may contribute to the observed trend

V. Conclusion & Future Works

In conclusion, through this project data pipelines were successfully created and implemented as DAGs in Apache Airflow workflow scheduler to automate the ETL process. Using the created workflow, the raw data in CSV and JSON format was extracted through APIs, stored in the NoSQL database, MongoDB, extracted again from MongoDB and transformed as per the requirements of the analysis and lastly was loaded to a PostgreSQL database. Further, after the creation and successful implementation of the Airflow pipeline, the transformed and clean data which was loaded to PostgreSQL was also used to analyse and provide answers to questions from the electric vehicles, public safety, and health domain. Although, the workflow was created successfully and the process did perform as intended, creating the workflow required several dependencies due to which Docker containers were used. Moreover, since the pipeline was running in a virtual environment several timeout and connection error was encountered. To mitigate these issues in future versions of the project, the recommendation would be to explore other workflow schedulers such as Dagster or Apache Spark and automate the ETL process. Also, a comparison between the performance of various workflows can provide a better idea of the actual implementation in a real-world business scenario.

REFERENCES

[1] F. Duarte, “Amount of Data created daily”, explodingtopics.com, 2023. [Online]. Available: https://explodingtopics.com/blog/data-generated-per-day [Accessed: Dec. 13, 2023]

[2] UNSW Sydney, “Descriptive, Predictive & Prescriptive Analytics: What are the differences?”. 2020. [Online]. Available: https://studyonline.unsw.edu.au/blog/descriptive-predictive-prescriptive-analytics, [Accessed: Dec. 14, 2023]

[3] [4] A. Munappy., J. Bosch, H. H. Olsson, “Data Pipeline Management in Practice: Challenges and Opportunities.”, pp 1-4 Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 12562: 168-184. http://dx.doi.org/10.1007/978-3-030-64148-1_11

[5] IBM, “What is a data pipeline?”, 2021, [Online]. Available: https://www.ibm.com/topics/data-pipeline, [Accessed: Dec. 15, 2023]

[6] A. Woodie, “Data Prep Still Dominates Data Scientists’ Time, Survey Finds”, datanami.com, Jul. 6, 2020 [Online]. Available: https://www.datanami.com/2020/07/06/data-prep-still-dominates-data-scientists-time-survey-finds/ [Accessed: Dec. 15, 2023]

[7] [8] Data Pipelines, “What is a Data Pipeline?”, datapipelines.com, Feb. 24, 2021 [Online]. Available: https://www.datapipelines.com/blog/what-is-a-data-pipeline/ [Accessed: Dec. 15, 2023]

[9] [10] Apache Airflow, [Online]. Available: https://airflow.apache.org/, [Accessed: Dec. 15, 2023]

[11] J. Jaeger, “These Countries Are Adopting Electric Vehicles the Fastest.” 2023. [Online]. Available: https://www.wri.org/ [Accessed on: Dec. 12, 2023].

[12] Central Statistics Office, “National Travel Survey 2019,” 2019. [Online]. Available: https://www.cso.ie/en/releasesandpublications/ep/p-nts/nationaltravelsurvey2019/electricvehicles/ [Accessed on: Dec. 14,2023].

[13] A. Bauman, ‘NYPD statistics show overall crime is down across New York City, but hate crimes are up’, [Online]. [Accessed on: Dec. 16, 2023].

[14] P. Oboza, N. Ogarek, M. Olszanecka-Glinianowicz, P. Kocelak, ‘The main causes of death in patients with COVID-19’, pp 1-4, PMID: 36930516 DOI: 10.26355/eurrev_202303_31589, [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/36930516/ [Accessed: Dec. 14, 2023]

[15] J.P. Pressley, ‘What Are Data Pipelines and How Do They Strengthen IT Infrastructure?’

[16] Data Catalog, “Electric Vehicle Population Data,” 2023. [Online]. Available: https://catalog.data.gov/dataset/electric-vehicle-population-data [Accessed on: Nov. 25, 2023]

[17] https://www.dol.wa.gov/, [Online]. [Accessed on: Dec. 14, 2023].

[18] https://data.gov/, [Online]. [Accessed on: Dec. 08, 2023].

[19] https://data.cityofnewyork.us/Public-Safety/NYPD-Arrest-Data-Year-to-Date-/uip8-fykc, [Online]. [Accessed on: Dec. 08, 2023].