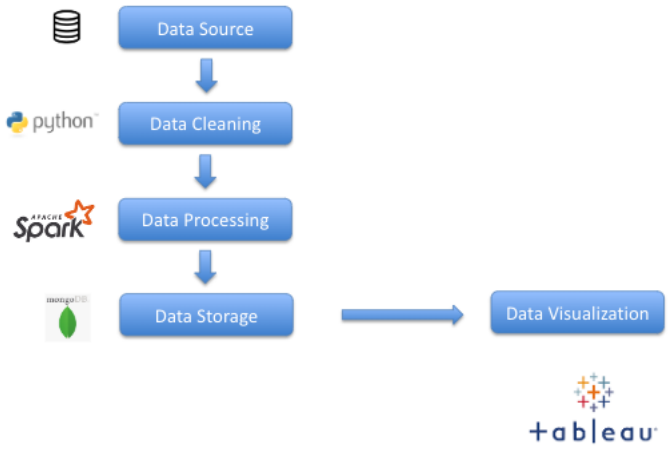
Flight Delay Analysis using Big Data Architecture

Introduction: A flight delay analysis system can have a significant impact on the aviation industry and passengers. By accurately analyzing flight delays, airlines can proactively manage operations and minimize delays, which can improve customer satisfaction and save costs. In this project, I explored the use of Apache Spark, MongoDB, and PySpark in flight delay analysis and how they can help analyze large amounts of flight data.

Sourced data: Sourced data from the Harvard Dataverse, which includes over 120 million records of flights primarily originating and terminating in the United States between 1987 and 2008. The compressed data occupies 1.6GB while the uncompressed data occupies 12GB.

Implemented Architecture:



Data Analysis with Pandas: Performing exploratory data analysis (EDA) on a small chunk of data using Pandas helped to understand the dataset's structure, identify any missing or inconsistent data, and gain insights into the data.

Data Cleaning with PySpark: After the EDA, the data cleaning process was extrapolated for four years of data using PySpark. PySpark's distributed computing framework allowed for the handling of the massive dataset with ease and quick performance of cleaning operations.

Data Storage with MongoDB: Once the data was cleaned, the clean data was written to a MongoDB collection. MongoDB's flexible schema and efficient querying capabilities enabled the storage and retrieval of data with ease.

Data Visualization with Tableau: MongoDB was connected to Tableau to perform data visualization to gain insights into the data. The visualization allowed for the spotting of patterns and relationships in the data that would be hard to identify otherwise.

1. Analysis of Flight Data: Delving deeper into the flight data revealed several interesting insights that could be useful in predicting flight delays and improving airport operations.
2. Identifying High Traffic Airports: Airports with high traffic experience more departure and arrival delays. Atlanta (ATL) was the busiest airport, followed by Chicago (ORD).
3. Identifying Airlines with the Most Delays: Southwest airlines had the most delays among the airlines analyzed.
4. Correlation between Arrival and Departure Delays: Departure delays were highly influenced by arrival delays, indicating a strong correlation between the two. This finding provided an opportunity to develop predictive models to forecast departure delays based on arrival delays.
5. Identifying Potential Bottlenecks: MLB, Florida, had the highest taxi-in time, despite having lower traffic. This information can be useful to identify potential bottlenecks in the airport's infrastructure and operations that could be improved to reduce taxi-in times.
6. Efficient Operations at Atlanta Airport: Despite heavy traffic at Atlanta (ATL), taxi-out time was nominal, indicating efficient operations. This finding provided an opportunity to study and identify the factors contributing to this efficiency, which could be replicated at other airports.
7. Seasonal Trends in Flight Delays: Flight delays were affected by the time of the year, with March and June having lower flight delays, and February and May having higher flight delays. This information helped to identify seasonal trends that could be incorporated into predictive models for flight delays.
8. Factors Contributing to Arrival Delays: There was no major dependency of arrival delay based on the distance. This finding provided an opportunity to investigate other factors contributing to arrival delays, such as weather conditions or airport congestion.

Further project considerations:

1. Move project to a cloud platform: Migrate the project infrastructure, application code, and data to a cloud-based environment to improve scalability, ease of management, and cost-effectiveness.
2. Integrate data from API with Apache Kafka: Use Apache Kafka to process and analyze data from multiple sources in real-time, which can help to identify patterns and insights.
3. Create a real-time analytical dashboard: Enhance the current dashboard to provide real-time insights to airport and airlines administration, which can help to make informed decisions.
4. Develop an ML algorithm to predict delays: Use historical and real-time data to develop an ML algorithm that can predict delays and provide proactive measures to minimize or prevent delays.
5. Build a Flask-based web app for public flight information and feedback: Develop a user-friendly web app using Flask that provides accurate and up-to-date information on flight status, delays, and feedback mechanisms to improve customer experience.

Conclusion: The use of Apache Spark, MongoDB, and PySpark can enable data analysts and data scientists to efficiently and effectively handle and analyze large amounts of flight data, leading to better insights into flight delays and the aviation industry as a whole.