**REAL-TIME ANALYTICS DASHBOARD**

**Team g : Podutur Lahari - DE126 & Tejaswini Gokanakonda - DE142**

**Date : 13-12-2024**

**Project Overview:**

This project involves building a real-time analytics dashboard using **Azure Databricks** and **PySpark** to handle and analyze streaming data. The solution includes ingesting real-time data from a streaming source such as Kafka, Event Hubs, or IoT devices into Databricks. The data is processed and transformed using **PySparkSQL** to derive actionable insights. The processed results are then visualized in real-time using Databricks Notebooks or integrated with visualization tools like **Power BI** or **Tableau** for advanced dashboards. This project showcases the seamless integration of big data processing with interactive visualizations to monitor trends, detect anomalies, and support decision-making in real-time.

**Real-time Analytics Dashboard Overview :**

A **Real-Time Analytics Dashboard** is an interactive tool designed to monitor, analyze, and visualize streaming data as it is generated, enabling immediate insights and decision-making. By leveraging technologies such as streaming platforms (e.g., Apache Kafka, Azure Event Hubs), big data processing frameworks (e.g., Apache Spark, PySpark), and visualization tools (e.g., Power BI, Tableau), real-time dashboards offer continuous updates on key metrics and events. They are widely used in domains like finance, healthcare, e-commerce, and IoT, where timely responses to data changes are critical. These dashboards integrate data ingestion, processing, and visualization to present live updates, detect trends, highlight anomalies, and improve operational efficiency.

**Dataset Overview :**

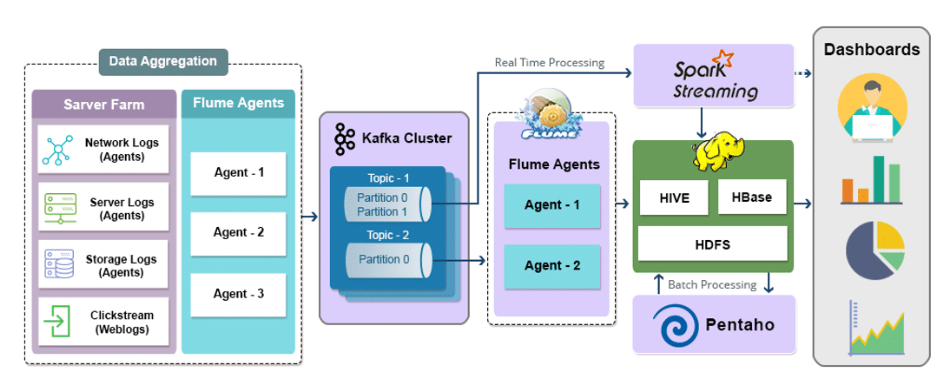
**Dataset Source :** Kaggle  
**Dataset Size :** 10,000 rows and 13 columns

**Description:**The dataset contains customer data for a credit card company, with fields relevant to customer demographics, financial details, and account activity. This data is ideal for analyzing customer behavior and trends in real-time.

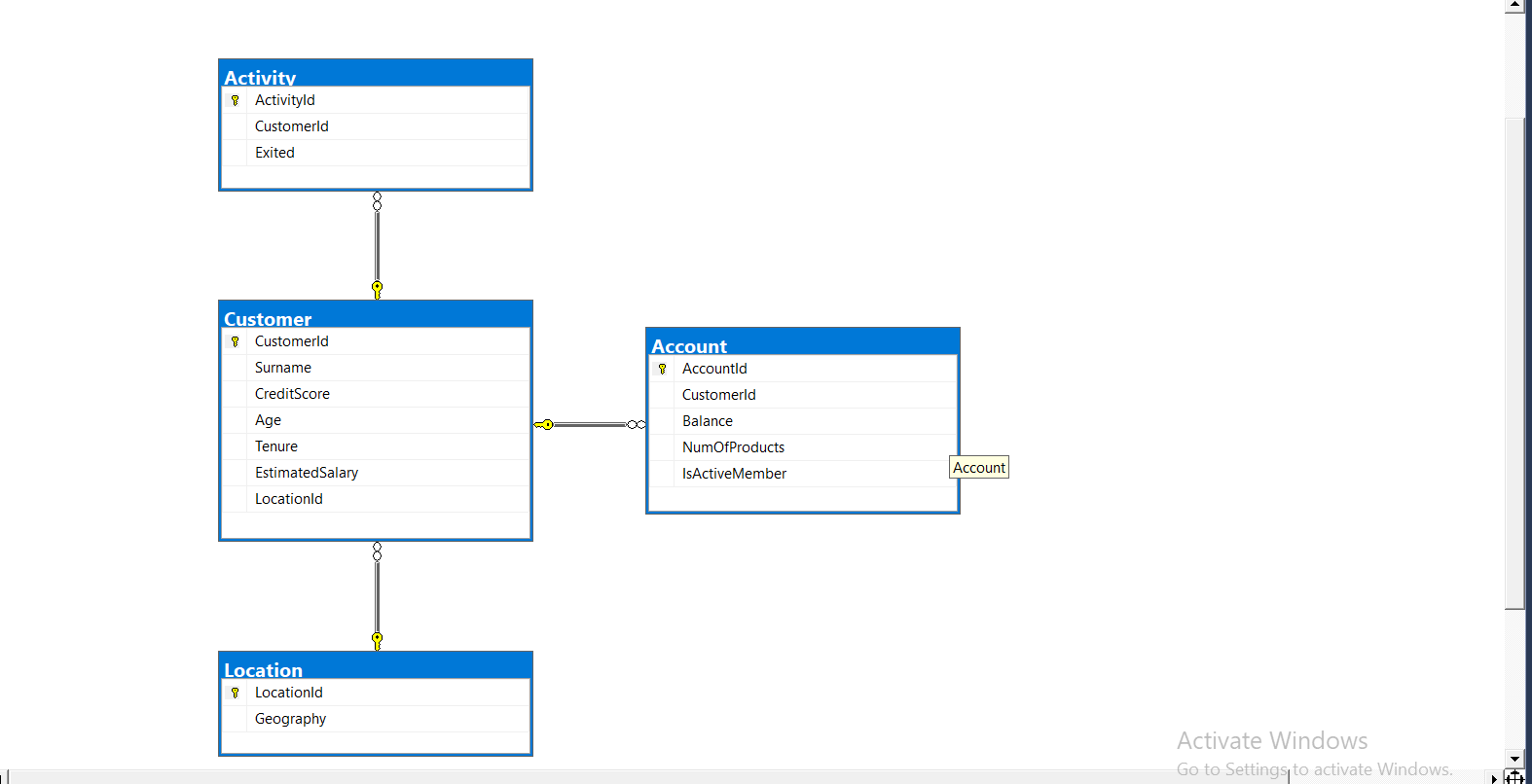
**Columns:**

* **RowNumber :** Sequential identifier for each row.
* **CustomerId :** Unique identifier for each customer.
* **Surname :** Customer's surname.
* **CreditScore :** Customer's credit score.
* **Geography :** Country of residence.
* **Gender :** Customer's gender.
* **Age :** Customer's age.
* **Tenure :** Number of years as a customer.
* **Balance :** Account balance.
* **Num Of Products :** Number of products the customer is using.
* **IsActiveMember :** Whether the customer is an active member (1: Yes, 0: No).
* **EstimatedSalary :** Estimated salary of the customer.
* **Exited :** Whether the customer has exited (1: Yes, 0: No).

**Architecture Diagram :**



**ER Diagram :**



**How it works :**

### **Source Data Files :**

We are referring to open-source data from Kaggle for the Real-Time Analytics Dashboard project. The data pertains to credit card customer activity and is structured as follows:

| **File Name** | **File Type** | **Description** |
| --- | --- | --- |
| CustomerDetails | CSV | Contains demographic information like customer ID, surname, and age. |
| CreditScores | CSV | Provides credit score details for each customer. |
| FinancialData | Single Line JSON | Includes customer balance and estimated salary information. |
| ActivityLogs | Single Line Nested JSON | Tracks customer activity status, including active membership and tenure. |
| Transactions | Multi Line JSON | Stores details about recent transactions and balance updates. |
| ChurnStatus | Split CSV Files | Indicates whether a customer has exited the service or remains active. |
| AnalyticalMetrics | Split Multi Line JSON Files | Summarizes processed insights like churn probabilities and product usage. |

This structured dataset provides the foundation for ingesting, processing, and visualizing data in real-time.

**Execution Overview :**

* The Real-Time Analytics Dashboard project ingests customer data (e.g., demographics, transactions, and credit scores) into Azure Databricks using streaming data sources like Azure Event Hubs or Kafka.
* PySpark processes and transforms this data, applying analytics to calculate key metrics, such as churn risk and account balance trends. Data is stored in Azure Blob Storage or Delta Lake for efficient querying.
* Real-time insights are visualized using Databricks Notebooks or external tools like Power BI. Monitoring and alerting features are set up to track key metrics, while the solution is optimized for scalability and performance using cluster-based processing.

**ETL Pipeline :**

**1. Ingestion Pipeline (Bronze Zone):**

* **Data Sources:** Raw data (CSV, JSON) is ingested, including customer details, transactions, and financial data.
* **Transformations:** Minimal transformations, such as dropping columns, renaming headers, and applying schema. Add audit columns like ingestion\_date, file\_source, and file\_date.
* **Output:** Processed data is stored in Delta Lake in the Bronze zone.

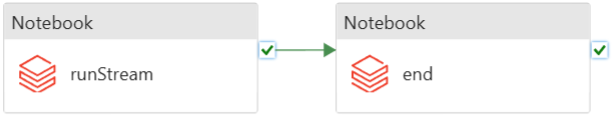
**2. Transformation Pipeline (Silver to Gold Zone):**

* **Input:** Preprocessed data from the Bronze zone.
* Transformations: Deduplication, joins, aggregations, and data enrichment to create final dimensional model tables in the Gold zone.
* **Output:** Transformed data is saved in Delta format.

**3. Automation and Scheduling:**

* The pipeline runs every Sunday at 10 PM using Azure Data Factory (ADF). If no new data is available, it skips execution.
* Tumbling Window Trigger executes both ingestion and transformation pipelines with file\_date as the parameter.

This ETL pipeline ensures smooth data flow from raw ingestion to real-time analytics.



### 

### **Azure Resources Used for Project :**

1. **Azure Databricks:**
   * Apache Spark is used for data processing and transformations.
   * Databricks SQL for querying and transforming data into the final dimensional model.
   * Notebooks for data visualization and real-time analytics.
2. **Azure Data Factory (ADF):**
   * Used to orchestrate and schedule the ETL pipelines (ingestion and transformation).
   * Tumbling Window Trigger is used for scheduling data processing.
   * Pipeline Scheduling: Runs every Sunday at 10 PM.
3. **Azure Blob Storage / Azure Data Lake Storage:**
   * Used to store raw and processed data in formats like CSV, JSON, and Delta.
   * Data is ingested into Blob Storage and stored in Delta Lake for further processing.
4. **Azure Event Hubs / Apache Kafka:**
   * For real-time data streaming to ingest data from various sources (e.g., customer transactions, activity logs).
   * Captures streaming data and feeds it into the Databricks environment.
5. **Azure SQL Database / Delta Lake:**
   * Stores processed data in a structured format (e.g., tables for dimensional models).
   * Delta Lake enables scalable storage and ACID transactions for data integrity.
6. **Power BI / Tableau:**
   * Used for data visualization and creating interactive dashboards that provide real-time insights from the processed data.
7. **Azure Monitor:**
   * Monitors pipeline execution, logs, and triggers real-time alerts for job failures or processing anomalies.

**Project Requirements :**

#### **1. Data Sources :**

* CSV and JSON files containing customer details, financial data, transaction logs, and activity status.
* Streaming data from platforms like Azure Event Hubs or Apache Kafka for real-time data ingestion.

#### **2. Azure Resources :**

* Azure Databricks for data processing using Apache Spark and Databricks SQL.
* Azure Data Factory (ADF) for orchestrating ETL workflows and scheduling pipeline execution.
* Azure Blob Storage or Azure Data Lake Storage for storing raw and processed data (in CSV, JSON, or Delta Lake format).
* Azure SQL Database for storing final processed data and dimensional tables.
* Power BI or Tableau for real-time data visualization and dashboard creation.
* Azure Monitor for pipeline monitoring and alerting.

#### **3. Data Processing & Transformation :**

* Apache Spark and PySpark for data ingestion, cleaning, and processing.
* Databricks SQL for transforming data into dimensional models and aggregating key metrics (e.g., churn prediction, transaction trends).
* Tumbling Window Trigger in ADF to automate pipeline execution on a weekly schedule.

#### **4. Automation & Scheduling :**

* Azure Data Factory (ADF) to automate ETL pipeline runs.
* Scheduled Triggers: Pipeline should run every Sunday at 10 PM and be designed to skip execution if no new data is available.

#### **5. Security & Compliance :**

* Ensure data encryption at rest and in transit using Azure Storage Encryption and Azure Key Vault.
* Implement role-based access control (RBAC) for Azure resources to ensure secure access to the data and pipelines.

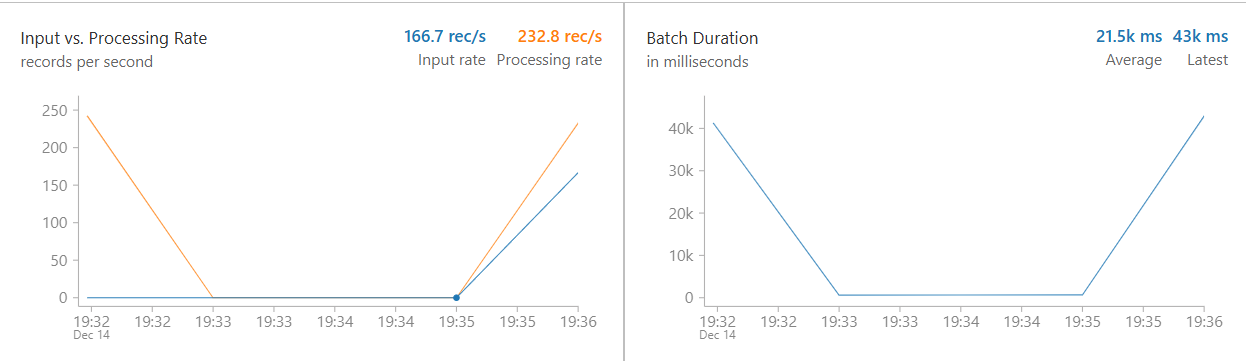
#### **6. Performance & Scalability :**

* Utilize Databricks clusters for scalable processing of large data volumes in real-time.
* Optimize Delta Lake for high-performance querying and storage of processed data.

#### **7. Visualization & Reporting :**

* Power BI or Tableau for building dynamic, real-time dashboards that visualize KPIs like churn risk, transaction trends, and balance analysis.
* Dashboards should update automatically as new data flows into the system.

**Analysis Results :**



**Tasks Performed :**

* Built a solution architecture for real-time analytics using Azure Databricks, Azure Data Lake Gen2, Azure Data Factory, and Power BI for visualization.
* Created and managed Azure Databricks service and the architecture within Azure, ensuring smooth integration for data processing.
* Worked with Databricks notebooks, using Databricks utilities, magic commands, and various transformations to process real-time data.
* Passed parameters between notebooks and created notebook workflows to automate data processing tasks.
* Created, configured, and monitored Databricks clusters, cluster pools, and jobs for scalable data processing and real-time analytics.
* Mounted Azure Storage in Databricks using secrets stored in Azure Key Vault for secure access to data.
* Worked with Databricks Tables and Databricks File System (DBFS) for data storage and processing.
* Used Delta Lake to implement a Lakehouse architecture for real-time data storage and processing, enabling fast queries and reliable data updates.
* Created dashboards in Power BI to visualize real-time analytics from Databricks tables, enabling business stakeholders to gain actionable insights.
* Connected Power BI to Azure Databricks tables for seamless data visualization and reporting.

### **Spark (PySpark and SQL) :**

* Spark architecture, Data Sources API, and Dataframe API for working with structured and unstructured data.
* PySpark - Ingested CSV and JSON files into the data lake as parquet tables for efficient querying.
* PySpark - Applied transformations such as Filter, Join, Simple Aggregations, GroupBy, Window functions, etc., to prepare the data for analytics.
* PySpark - Created global and temporary views for data access and transformation.
* Spark SQL - Created databases, tables, and views for structured data processing and querying.
* Spark SQL - Applied transformations such as Filter, Join, GroupBy, and Window functions to prepare data for reporting.
* Implemented full refresh and incremental load patterns using partitions for efficient data processing.

### **Delta Lake :**

* Performed Read, Write, Update, Delete, and Merge operations to Delta Lake using both PySpark and SQL for managing data consistency.
* Leveraged Delta Lake features like History, Time Travel, and Vacuum for managing historical data and optimizing storage.
* Converted Parquet files to Delta files for improved performance and flexibility in real-time data processing.
* Implemented incremental load patterns using Delta Lake to handle large data volumes efficiently.

### **Azure Data Factory :**

* Created pipelines in Azure Data Factory to execute Databricks notebooks, automating ETL tasks.
* Designed robust pipelines to handle unexpected scenarios, such as missing files, ensuring smooth data flow.
* Created dependencies between activities and pipelines to ensure the correct order of execution.
* Scheduled pipelines using ADF triggers to execute at regular intervals, ensuring timely data processing.
* Monitored triggers and pipelines to ensure successful execution and handle errors or issues promptly.

**Technologies/Tools Used:**

* Pyspark
* Spark SQL
* Delta Lake
* Azure Databricks
* Azure Data Factory
* Azure Date Lake Storage Gen2
* Azure Key vault
* Power BI (Optional)