# Solution - *Explain your solution here in a step by step manner.*

The healthcare insurance industry is increasingly turning to big data analytics to enhance customer engagement, optimize operational efficiency, and boost revenues. In this project, we aim to address a health insurance company’s challenge by developing a comprehensive data pipeline solution using Azure and Databricks. The primary goal is to analyze data from various sources to better understand customer behavior, optimize insurance offerings, and calculate royalties, thereby driving increased revenue.

The project involves processing data from competitors and various third-party sources, which includes customer information, claims data, hospital records, and more. By leveraging advanced data analytics, the company can identify trends, assess customer needs, and create targeted insurance policies. The solution is built using cloud technology, specifically Azure and Databricks, to ensure scalability, security, and efficiency in processing large datasets.

**Step-by-Step Solution**

1. **Data Ingestion:**
   * **Objective:** Ingest data from multiple sources including CSV and JSON files, databases, and APIs into Azure Data Lake Storage (ADLS).
   * **Steps:**
     + Utilize Azure Data Factory (ADF) to orchestrate the ingestion process.
     + Set up data connectors in ADF to pull data from different sources such as on-premise databases, third-party APIs, and raw files (CSV, JSON).
     + Store the ingested raw data in a structured format within ADLS, ensuring proper partitioning based on attributes like date, country, and data source.
2. **Data Cleaning and Transformation:**
   * **Objective:** Clean and transform the raw data into a structured, usable format.
   * **Steps:**
     + Load the raw data from ADLS into Azure Databricks for processing.
     + Use PySpark to perform data cleaning tasks, such as handling missing values, removing duplicates, and normalizing inconsistent data formats.
     + Apply data transformations to align with business logic, such as calculating age from birth dates, categorizing claims, and standardizing text fields (e.g., city names).
     + Transform data into meaningful datasets such as subscriber data, claim details, and disease profiles.
3. **Data Storage and Management:**
   * **Objective:** Store the cleaned and transformed data in a scalable and query-efficient format.
   * **Steps:**
     + Store the transformed data in Delta Lake tables within Azure Databricks, allowing for versioned and optimized storage.
     + Organize data into different layers (bronze, silver, gold) based on processing stages, where:
       - **Bronze Layer:** Raw ingested data.
       - **Silver Layer:** Cleaned and transformed data.
       - **Gold Layer:** Aggregated and enriched data ready for analysis and reporting.
4. **Data Analysis and Querying:**
   * **Objective:** Perform data analysis to extract insights that support business decisions.
   * **Steps:**
     + Utilize Databricks SQL to run analytical queries on the gold-layer datasets.
     + Implement queries to answer business questions, such as identifying the disease with the most claims, subscribers under 30 years old, and the profitability of different insurance groups.
     + Aggregate data to provide insights into customer behavior, claim rejection rates, hospital performance, and more.
5. **Reporting and Visualization:**
   * **Objective:** Present analytical results to business stakeholders in an understandable format.
   * **Steps:**
     + Integrate Azure Databricks with Power BI to create interactive dashboards and reports.
     + Develop dashboards to track key metrics like subscriber demographics, claim trends, disease prevalence, and financial performance.
     + Ensure reports are dynamic, allowing stakeholders to filter data by different dimensions such as time, geography, and disease type.
6. **Pipeline Orchestration and Automation:**
   * **Objective:** Automate the end-to-end data processing pipeline to ensure timely and consistent data updates.
   * **Steps:**
     + Use Azure Data Factory to schedule and orchestrate the data pipeline, ensuring that data ingestion, processing, and reporting are performed at regular intervals (e.g., daily, weekly).
     + Implement error handling and monitoring to track the pipeline’s performance and automatically retry or alert on failures.
     + Set up notifications via Azure Monitor or custom scripts to inform stakeholders of pipeline status, such as completion or failure.
7. **Security and Compliance:**
   * **Objective:** Ensure that the data pipeline adheres to security and regulatory requirements.
   * **Steps:**
     + Implement data encryption both at rest (in ADLS and Databricks) and in transit.
     + Apply role-based access controls (RBAC) within Azure to restrict access to sensitive data and ensure that only authorized personnel can access specific data assets.
     + Maintain audit logs of data access and processing activities to comply with healthcare regulations like HIPAA.
8. **Testing and Validation:**
   * **Objective:** Validate that the data pipeline functions correctly and meets business requirements.
   * **Steps:**
     + Perform unit testing on individual components, such as data transformation scripts and analytical queries.
     + Conduct integration testing to ensure all pipeline components work together seamlessly.
     + Validate the accuracy of the processed data by comparing results against known benchmarks or manually calculated values.
     + Test the performance of the pipeline under different data loads to ensure it can scale effectively.
9. **Deployment and Monitoring:**
   * **Objective:** Deploy the data pipeline into a production environment and monitor its performance.
   * **Steps:**
     + Deploy the pipeline to a production Azure Databricks workspace, using Azure DevOps for continuous integration and deployment (CI/CD).
     + Set up monitoring dashboards in Azure Monitor to track pipeline performance metrics such as runtime, data volume, and error rates.
     + Continuously monitor the pipeline and adjust resources (e.g., scaling compute clusters) to maintain performance and cost efficiency.
10. **Iteration and Improvement:**
    * **Objective:** Continuously improve the data pipeline based on feedback and evolving business needs.
    * **Steps:**
      + Collect feedback from business users regarding the accuracy and usability of the reports and dashboards.
      + Iterate on the data models and processing logic to refine the insights provided.
      + Update the pipeline to accommodate new data sources, changes in business logic, or improved processing techniques.

# Use Cases - *List down all the use cases on which this solution will be applicable.*

This solution can be applied to a wide range of use cases, each aimed at driving specific business outcomes for the healthcare insurance company. Below are 10 use cases categorized into relevant sections, with detailed descriptions of each.

**Section 1: Customer Behavior and Profiling**

**Use Case 1:** Identifying High-Risk Customers

* + **Description:** Analyze the historical claims data and medical records to identify customers who are at high risk of developing chronic diseases. By understanding these customers' profiles, the company can proactively offer customized insurance plans that cater to their specific needs.
  + **Objective:** Improve customer retention by providing tailored insurance packages that address specific health risks.
  + **Outcome:** Increased customer satisfaction and reduced churn rates.

**Use Case 2:** Customer Segmentation for Targeted Marketing

* + **Description:** Segment the customer base based on demographics, medical history, and past interactions with the company. This segmentation will help in identifying groups of customers who are likely to be interested in specific insurance products or services.
  + **Objective:** Enhance marketing efficiency by targeting the right customers with the most relevant offers.
  + **Outcome:** Improved conversion rates and higher sales of insurance policies.

**Use Case 3:** Predictive Modeling for Future Claims

* + **Description:** Use predictive analytics to forecast the likelihood of customers filing future claims based on their current health status, lifestyle, and past claim history. This model can be used to adjust premium rates or suggest preventive health measures to customers.
  + **Objective:** Mitigate risk by anticipating future claims and setting premiums accordingly.
  + **Outcome:** Optimized premium pricing and reduced financial risk for the company.

**Section 2: Claims Management and Optimization**

**Use Case 4:** Streamlining Claims Processing

* + **Description:** Automate the processing of claims by categorizing them based on predefined criteria such as disease type, claim amount, and patient history. This will enable faster processing of claims, especially for routine or low-risk cases.
  + **Objective:** Reduce the turnaround time for claims processing.
  + **Outcome:** Improved customer satisfaction due to quicker claim settlements.

**Use Case 5:** Fraud Detection in Claims

* + **Description:** Implement machine learning models to detect patterns of fraudulent claims by analyzing inconsistencies in claim data, such as unusually high claim amounts or repeated claims for the same treatment. Flag suspicious claims for further investigation.
  + **Objective:** Minimize losses due to fraudulent claims.
  + **Outcome:** Increased profitability and reduced fraud-related expenses.

**Use Case 6:** Analyzing Rejected Claims

* + **Description:** Analyze the reasons for claim rejections, such as missing information, policy exclusions, or incorrect claim submissions. Provide insights to improve the claims submission process and reduce rejection rates.
  + **Objective:** Lower the percentage of rejected claims by identifying and addressing common issues.
  + **Outcome:** Increased claim approval rates and improved customer trust.

**Section 3: Financial Performance and Profitability**

**Use Case 7:** Profitability Analysis by Policy Group

* + **Description:** Evaluate the profitability of different policy groups (e.g., government vs. private) by analyzing the premiums collected, claims paid, and administrative costs associated with each group. Identify which groups are most profitable and why.
  + **Objective:** Optimize the portfolio of insurance products to focus on the most profitable segments.
  + **Outcome:** Higher overall profitability for the company.

**Use Case 8:** Tracking Premium Payment Patterns

* + **Description:** Monitor the payment patterns of subscribers to identify trends, such as late payments or defaults. Use this data to predict future payment behaviors and implement measures to encourage timely payments.
  + **Objective:** Reduce the incidence of payment defaults and ensure a steady cash flow.
  + **Outcome:** Improved financial stability and reduced operational risk.

**Section 4: Operational Efficiency and Strategic Planning**

**Use Case 9:** Optimizing Hospital Network

* + **Description:** Analyze the data related to hospital services, patient admissions, and treatment outcomes to identify which hospitals provide the best care at the lowest cost. Use this information to optimize the network of hospitals included in the insurance plans.
  + **Objective:** Ensure that subscribers have access to high-quality healthcare while controlling costs.
  + **Outcome:** Enhanced quality of service and reduced healthcare costs for the company.

**Use Case 10:** Enhancing Strategic Decision-Making

* + **Description:** Provide business leaders with comprehensive dashboards that include key metrics such as claim rates, customer demographics, and financial performance. These dashboards will support data-driven decision-making and help in setting long-term business strategies.
  + **Objective:** Align business strategies with data-driven insights to achieve company goals.
  + **Outcome:** Improved strategic planning and business outcomes.

1. **Database Design - List down all possible db(Azure Synapse Analytics SQL pool) tables here**

For this healthcare insurance project, we will design the database tables in Azure Synapse Analytics SQL Pool, considering the necessary relationships and key constraints. Below is the detailed table metadata, including primary keys (PK) and foreign keys (FK), along with the appropriate relational database datatypes.

**a. Tables Metadata Info with PK/FK Relationship**

1. **disease\_tbl**
   * **Columns:**
     + SubGrpID - VARCHAR(50)
     + Disease\_ID - INT (Primary Key)
     + Disease\_name - VARCHAR(100)
   * **Primary Key:** Disease\_ID
   * **Foreign Keys:** SubGrpID references subgrp\_tbl(SubGrp\_id)
2. **group\_tbl**
   * **Columns:**
     + Country - VARCHAR(50)
     + premium\_written - FLOAT
     + zipcode - VARCHAR(10)
     + Grp\_Id - VARCHAR(50) (Primary Key)
     + Grp\_Name - VARCHAR(100)
     + Grp\_Type - VARCHAR(50)
     + city - VARCHAR(50)
     + year - INT
   * **Primary Key:** Grp\_Id
3. **group\_sub\_group\_tbl**
   * **Columns:**
     + SubGrp\_ID - VARCHAR(50) (Primary Key)
     + Grp\_Id - VARCHAR(50) (Primary Key)
   * **Primary Keys:** SubGrp\_ID, Grp\_Id
   * **Foreign Keys:**
     + SubGrp\_ID references subgrp\_tbl(SubGrp\_id)
     + Grp\_Id references group\_tbl(Grp\_Id)
4. **hospital\_tbl**
   * **Columns:**
     + Hospital\_id - VARCHAR(50) (Primary Key)
     + Hospital\_name - VARCHAR(100)
     + city - VARCHAR(50)
     + state - VARCHAR(50)
     + country - VARCHAR(50)
   * **Primary Key:** Hospital\_id
5. **patient\_tbl**
   * **Columns:**
     + Patient\_id - INT (Primary Key)
     + Patient\_name - VARCHAR(100)
     + patient\_gender - VARCHAR(10)
     + patient\_birth\_date - DATE
     + patient\_phone - VARCHAR(15)
     + disease\_name - VARCHAR(100)
     + city - VARCHAR(50)
     + hospital\_id - VARCHAR(50)
   * **Primary Key:** Patient\_id
   * **Foreign Keys:** hospital\_id references hospital\_tbl(Hospital\_id)
6. **subgrp\_tbl**
   * **Columns:**
     + SubGrp\_id - VARCHAR(50) (Primary Key)
     + SubGrp\_Name - VARCHAR(100)
     + Monthly\_Premium - FLOAT
   * **Primary Key:** SubGrp\_id
7. **subscriber\_tbl**
   * **Columns:**
     + sub\_id - VARCHAR(50) (Primary Key)
     + first\_name - VARCHAR(50)
     + last\_name - VARCHAR(50)
     + Street - VARCHAR(100)
     + Birth\_date - DATE
     + Gender - VARCHAR(10)
     + Phone - VARCHAR(15)
     + Country - VARCHAR(50)
     + City - VARCHAR(50)
     + ZipCode - VARCHAR(10)
     + Subgrp\_id - VARCHAR(50)
     + Elig\_ind - BIT
     + eff\_date - DATE
     + term\_date - DATE
   * **Primary Key:** sub\_id
   * **Foreign Keys:** Subgrp\_id references subgrp\_tbl(SubGrp\_id)
8. **claim\_tbl**
   * **Columns:**
     + claim\_id - INT (Primary Key)
     + patient\_id - INT
     + disease\_name - VARCHAR(100)
     + SUB\_ID - VARCHAR(50)
     + Claim\_Or\_Rejected - VARCHAR(10)
     + claim\_type - VARCHAR(50)
     + claim\_amount - FLOAT
     + claim\_date - DATE
   * **Primary Key:** claim\_id
   * **Foreign Keys:**
     + patient\_id references patient\_tbl(Patient\_id)
     + SUB\_ID references subscriber\_tbl(sub\_id)

**b. ER Diagram (Optional)**

The Entity-Relationship (ER) diagram for this database would depict the relationships between the tables, including primary and foreign key constraints. Although this step is optional, an ER diagram can be a valuable tool for visualizing how the tables are connected, especially when working with complex datasets.

This detailed database design ensures that all relevant aspects of the healthcare insurance company's data are organized efficiently, allowing for complex queries and analysis in Azure Synapse Analytics.

A screenshot of a computer screen

Description automatically generated

**Output Tables**

**1. disease\_max\_claims\_tbl**

* **Purpose:** List the disease with the maximum number of claims.
* **Columns:**
  + Disease\_ID - INT (Primary Key)
  + Disease\_name - VARCHAR(100)
  + Claim\_Count - INT
* **Primary Key:** Disease\_ID

**2. subscribers\_under\_30\_tbl**

* **Purpose:** Find subscribers under the age of 30 who are subscribed to any subgroup.
* **Columns:**
  + sub\_id - VARCHAR(50) (Primary Key)
  + first\_name - VARCHAR(50)
  + last\_name - VARCHAR(50)
  + Birth\_date - DATE
  + Subgrp\_id - VARCHAR(50)
* **Primary Key:** sub\_id
* **Foreign Keys:** Subgrp\_id references subgrp\_tbl(SubGrp\_id)

**3. group\_max\_subgroups\_tbl**

* **Purpose:** Find out which group has the maximum number of subgroups.
* **Columns:**
  + Grp\_Id - VARCHAR(50) (Primary Key)
  + Grp\_Name - VARCHAR(100)
  + Subgroup\_Count - INT
* **Primary Key:** Grp\_Id
* **Foreign Keys:** None

**4. hospital\_most\_patients\_tbl**

* **Purpose:** Find the hospital that serves the most number of patients.
* **Columns:**
  + Hospital\_id - VARCHAR(50) (Primary Key)
  + Hospital\_name - VARCHAR(100)
  + Patient\_Count - INT
* **Primary Key:** Hospital\_id
* **Foreign Keys:** None

**5. subgroups\_most\_subscribed\_tbl**

* **Purpose:** Find out which subgroups are subscribed to the most number of times.
* **Columns:**
  + SubGrp\_id - VARCHAR(50) (Primary Key)
  + SubGrp\_Name - VARCHAR(100)
  + Subscription\_Count - INT
* **Primary Key:** SubGrp\_id
* **Foreign Keys:** None

**6. rejected\_claims\_count\_tbl**

* **Purpose:** Find out the total number of claims which were rejected.
* **Columns:**
  + Rejected\_Claim\_Count - INT
* **Primary Key:** None (Aggregate result)

**7. claims\_by\_city\_tbl**

* **Purpose:** Identify the city from where most claims are originating.
* **Columns:**
  + City - VARCHAR(50) (Primary Key)
  + Claim\_Count - INT
* **Primary Key:** City
* **Foreign Keys:** None

**8. policy\_type\_subscriptions\_tbl**

* **Purpose:** Determine whether subscribers are more likely to subscribe to government or private policies.
* **Columns:**
  + Policy\_Type - VARCHAR(50) (Primary Key)
  + Subscription\_Count - INT
* **Primary Key:** Policy\_Type
* **Foreign Keys:** None

**9. average\_monthly\_premium\_tbl**

* **Purpose:** Calculate the average monthly premium paid by subscribers.
* **Columns:**
  + Average\_Monthly\_Premium - FLOAT
* **Primary Key:** None (Aggregate result)

**10. most\_profitable\_group\_tbl**

* **Purpose:** Identify the most profitable group.
* **Columns:**
  + Grp\_Id - VARCHAR(50) (Primary Key)
  + Grp\_Name - VARCHAR(100)
  + Profit - FLOAT
* **Primary Key:** Grp\_Id
* **Foreign Keys:** None

**11. patients\_under\_18\_cancer\_tbl**

* **Purpose:** List all the patients under the age of 18 who have been admitted for cancer.
* **Columns:**
  + Patient\_id - INT (Primary Key)
  + Patient\_name - VARCHAR(100)
  + Birth\_date - DATE
  + disease\_name - VARCHAR(100)
* **Primary Key:** Patient\_id
* **Foreign Keys:** None

**12. cashless\_patients\_high\_charges\_tbl**

* **Purpose:** List patients who have cashless insurance and have total charges greater than or equal to Rs. 50,000.
* **Columns:**
  + Patient\_id - INT (Primary Key)
  + Patient\_name - VARCHAR(100)
  + Total\_Charges - FLOAT
* **Primary Key:** Patient\_id
* **Foreign Keys:** None

**13. female\_patients\_knee\_surgery\_tbl**

* **Purpose:** List female patients over the age of 40 who have undergone knee surgery in the past year.
* **Columns:**
  + Patient\_id - INT (Primary Key)
  + Patient\_name - VARCHAR(100)
  + Birth\_date - DATE
  + Surgery\_Date - DATE
* **Primary Key:** Patient\_id
* **Foreign Keys:** None

**Summary**

Each output table is designed to address a specific requirement from the project. The columns and datatypes are tailored to capture the necessary information, while primary keys ensure uniqueness. Foreign keys are included where relationships between tables are relevant for ensuring data integrity.

# Technologies and Platforms to be used in this solution -*List down list of technologies like spark, aws and databricks etc.*

To successfully implement and manage the healthcare insurance project, various technologies and platforms will be leveraged. Each technology plays a specific role in ensuring efficient data processing, storage, and project management. Below is a detailed list of the technologies and platforms to be used, along with their purposes and benefits.

**1. Azure**

**Azure** is a comprehensive cloud computing platform by Microsoft that provides a wide range of services for building, deploying, and managing applications. For this project, Azure services will be crucial for data storage, data processing, and overall infrastructure management.

* **Azure Blob Storage**
  + **Purpose:** Store large amounts of unstructured data, including raw data files such as CSVs and JSONs.
  + **Benefits:** Scalable, secure, and cost-effective storage solution for big data.
* **Azure Data Factory**
  + **Purpose:** Orchestrate and automate data workflows, including data ingestion, data transformation, and data loading.
  + **Benefits:** Enables the creation and management of data pipelines, integrating data from various sources.
* **Azure Key Vault**
  + **Purpose:** Manage and safeguard sensitive information such as API keys, passwords, and certificates.
  + **Benefits:** Ensures secure storage and access to sensitive data used within the project.

**2. Azure Synapse Analytics**

**Azure Synapse Analytics** (formerly known as Azure SQL Data Warehouse) is an integrated analytics service that combines big data and data warehousing. It provides capabilities for data integration, data warehousing, and big data analytics.

* **Purpose:** Store structured and semi-structured data, perform large-scale data processing, and execute complex queries.
* **Benefits:** Offers a unified experience for data integration and analytics, allowing for scalable and performant data processing.
* **SQL Pools**
  + **Purpose:** Use SQL-based queries to perform large-scale data analytics and reporting.
  + **Benefits:** Optimized for high-performance analytics on large datasets.

**3. Azure Databricks**

**Azure Databricks** is an Apache Spark-based analytics platform optimized for the Azure cloud. It provides a collaborative environment for data engineers, data scientists, and analysts.

* **Purpose:** Process and analyze large datasets using Spark-based analytics, create data pipelines, and perform machine learning tasks.
* **Benefits:** Enhances productivity with interactive notebooks, integrates seamlessly with Azure services, and scales efficiently for big data processing.
* **Databricks Notebooks**
  + **Purpose:** Develop and execute code for data processing and analysis in a collaborative environment.
  + **Benefits:** Supports multiple languages (Python, Scala, SQL), making it versatile for various analytical tasks.

**4. Jira**

**Jira** is a project management tool by Atlassian that helps teams plan, track, and manage agile projects.

* **Purpose:** Manage project tasks, track progress, and collaborate with team members.
* **Benefits:** Provides detailed project tracking, issue management, and reporting capabilities.
* **Use Case:** Track the progress of data pipeline development, monitor task completion, and manage project milestones.

**5. GitHub**

**GitHub** is a platform for version control and collaboration, allowing teams to manage code repositories and track changes.

* **Purpose:** Store and manage source code, collaborate on development, and track changes to codebases.
* **Benefits:** Facilitates code versioning, collaboration, and continuous integration/continuous deployment (CI/CD) practices.
* **Use Case:** Version control for code related to data processing, maintain documentation, and manage changes across different components of the solution.

**6. Power BI**

**Power BI** is a business analytics tool by Microsoft that provides interactive visualizations and business intelligence capabilities.

* **Purpose:** Create interactive reports and dashboards to visualize data insights and trends.
* **Benefits:** Provides intuitive data visualization and reporting, enabling data-driven decision-making.
* **Use Case:** Develop dashboards to visualize key metrics such as claim statistics, customer demographics, and financial performance.

**7. Apache Airflow**

**Apache Airflow** is an open-source platform to programmatically author, schedule, and monitor workflows.

* **Purpose:** Manage and orchestrate complex data workflows and ETL processes.
* **Benefits:** Provides robust scheduling and monitoring of data pipelines and workflows.
* **Use Case:** Schedule and monitor ETL jobs, ensuring timely and accurate data processing.

**Summary**

The technologies and platforms listed above will work together to build a robust and scalable solution for processing healthcare data. Azure and its services will provide the necessary infrastructure and data management capabilities, while Databricks will handle data processing and analytics. Jira and GitHub will support project management and version control, respectively, while Power BI will offer advanced data visualization and reporting. Apache Airflow will be used for managing and orchestrating data workflows, ensuring efficient and reliable data processing.