# **Solution -**

1. Upload Data to S3: Competitor data is uploaded to an AWS S3 bucket for storage and processing.
2. Create Connection (Access Key): An IAM role is created in AWS with the necessary permissions, and access keys are generated for establishing a connection to S3.
3. DataFrame in Spark: Using PySpark, a dataframe is created from the data stored in S3 for further processing and analysis.
4. Data Cleaning: The data undergoes a cleaning process to remove duplicates, handle missing values, and ensure data consistency.
5. Data Validation: The dataframe undergoes data validation checks to ensure data integrity and quality.
6. Data Analysis: PySpark is used to analyze the data, identify trends in customer behavior, and generate insights.
7. Offer Customization: Based on the analysis results, offers are customized for customers to increase policy sales.
8. Royalty Calculation: Royalties are calculated for customers who have purchased policies in the past, incentivizing repeat purchases.
9. Write to Redshift: The validated and analyzed data is written to an AWS Redshift table for storage and further analysis.
10. Username and Password: The Redshift username and password are securely stored and used to establish the connection for writing to Redshift.

# Use Cases –

* 1. Identify Disease with Maximum Claims

Description: Analyze claims data to determine the disease with the highest number of claims.

Preconditions: Claims data is uploaded to S3 and cleaned.

Steps:

* + Load the claims data into a PySpark dataframe.
  + Group the data by disease and count the number of claims for each disease.
  + Identify the disease with the maximum number of claims.
  + Generate a report or visualization to present the findings.
  1. Identify Subscribers Under Age 30 with Subgroup Subscription

Description: Identify subscribers under the age of 30 who have subscribed to any subgroup.

Preconditions: Subscriber data is uploaded to S3 and cleaned.

Steps:

* + Load the subscriber data into a PySpark dataframe.
  + Filter the data to include only subscribers under the age of 30.
  + Check if these subscribers have subscribed to any subgroup.
  + Generate a report or visualization to show the findings.
  1. Identify Group with Maximum Subgroups

Description: Determine the group with the maximum number of subgroups.

Preconditions: Group data is uploaded to S3 and cleaned.

Steps:

* + Load the group data into a PySpark dataframe.
  + Count the number of subgroups for each group.
  + Identify the group with the maximum number of subgroups.
  + Generate a report or visualization to present the findings.
  1. Identify Hospital Serving Most Patients

Description: Identify the hospital that serves the most number of patients.

Preconditions: Hospital data is uploaded to S3 and cleaned.

Steps:

* + Load the hospital data into a PySpark dataframe.
  + Count the number of patients served by each hospital.
  + Identify the hospital with the highest count of patients.
  + Generate a report or visualization to show the findings.
  1. Identify Subgroups with Most Subscriptions

Description: Identify the subgroups that have the highest number of subscriptions.

Preconditions: Subscription data is uploaded to S3 and cleaned.

Steps:

* + Load the subscription data into a PySpark dataframe.
  + Group the data by subgroup and count the number of subscriptions for each subgroup.
  + Identify the subgroups with the highest number of subscriptions.
  + Generate a report or visualization to present the findings.

1. Database Design - List down all possible db(Redshift) tables here

## Tables Metadata Info with Pk/FK relationship –

Based on the provided use cases, here is a list of possible Redshift tables along with their primary key (PK) and foreign key (FK) relationships:

1. Claims Table:

Columns: Claim\_id (PK), Disease\_ID, SubGrp\_ID, Hospital\_id, Claim\_amount, Claim\_date

1. Patients Table:

Columns: Patient\_id (PK), Patient\_name, patient\_gender, patient\_birth\_date, patient\_phone, city, zipcode

1. Subscribers Table:

Columns: Sub\_id (PK), Subgrp\_id (FK), first\_name, last\_name, city, country, zipcode, Birth\_date, Gender, Phone, eff\_date, term\_date

1. Groups Table:

Columns: Grp\_Id (PK), Grp\_Name, Grp\_Type

1. Subgroup Table:

Columns: Subgrp\_id (PK), Grp\_Id (FK), Subgrp\_Name, Monthly\_Premium

1. Hospitals Table:

Columns: Hospital\_id (PK), Hospital\_name, city, state, country

1. Subgroup\_Subscription Table:

Columns: Sub\_id (FK), Subgrp\_id (FK), eff\_date, term\_date

1. Rejected\_Claims Table:

Columns: Claim\_id (FK), Rejection\_reason

1. Patient\_Disease Table:

Columns: Patient\_id (FK), Disease\_ID, Disease\_name

1. Premium\_Payment Table:

Columns: Sub\_id (FK), Payment\_date, Payment\_amount

1. City Table:

Columns: City\_id (PK), City\_name

1. Country Table:

Columns: Country\_id (PK), Country\_name

1. State Table:

Columns: State\_id (PK), State\_name

## ER diagram –

A screenshot of a computer

Description automatically generated

# Technologies and Platforms to be used in this solution –

The following technologies and platforms will be used in this solution:

1. Apache Spark: For data processing and analysis.
2. AWS (Amazon Web Services): For cloud storage, data warehousing, and computing resources.
3. AWS Redshift: For data warehousing and analytics.
4. Databricks: For collaborative big data analytics.
5. Jira: For project management and issue tracking.
6. GitHub: For version control and collaboration on code.
7. PySpark: For Python API for Spark.
8. AWS S3: For cloud storage of data.
9. AWS EMR (Elastic MapReduce) Studio: For data processing using Apache Hadoop and Spark.

## 