**Synthetic Insurance Data Generation Script Documentation**

**Overview**

This Python script generates synthetic data for an insurance scenario that simulates a 360° view of customers for an insurance company. The data is distributed across four tables—**Customers**, **Policies**, **Claims**, and **Communications**—and is designed for use in BI tools or to be consolidated into a master table in a SQL Server database. Key features include:

* **Realistic Data Generation:**
  + Customers are assigned to Kenyan counties with a higher probability of major urban centers (Nairobi, Mombasa, Kisumu, Eldoret, Nakuru) while others are randomly selected from a broader list.
  + Policies, claims, and communications are generated with realistic date ranges, monetary values, and status fields.
* **Interlinked Tables:**
  + Each table uses unique identifiers (IDs) to establish relationships between records.
  + For example, the customer\_id in the **Customers** table links to the **Policies**, **Claims**, and **Communications** tables.
  + The **Claims** table includes both a policy\_id and a customer\_id to ensure direct mapping to the customer.
* **Enhanced Communications:**
  + The **Communications** table captures various interaction channels (including Chatbot) and categorizes each interaction as a Complaint, Feedback, or Call Log.
  + It also optionally links communications to specific claims (through related\_claim\_id), simulating realistic follow-up scenarios.
* **CSV Outputs:**
  + The script outputs four CSV files: customers.csv, policies.csv, claims.csv, and communications.csv.
  + These files can be imported into SQL Server or any BI tool for further analysis, reporting, or the creation of a consolidated master table.

**Explanation of the Code**

1. **Configuration & Parameter Setup**  
   The script begins by setting up global parameters such as the number of customers (NUM\_CUSTOMERS), available policy types, claim statuses, and communication channels. It also defines lists of major Kenyan counties and other counties to create a realistic geographical distribution for customers.
2. **Helper Functions**  
   A helper function choose\_county() is used to assign a Kenyan county to a customer, with a 70% chance of choosing a major urban county.  
   Another helper function, generate\_message\_and\_sentiment(), returns a random reason category (Complaint, Feedback, or Call Log), a sample message text, and a corresponding sentiment label (Positive, Neutral, or Negative). These are used for populating the communications details.
3. **Generating Customers**  
   A loop creates a specified number of customers, each with:
   * A unique customer\_id.
   * Randomly generated first and last names.
   * A randomly assigned age, gender, and region (using the Kenyan county distribution).
   * A VIP status indicator.

All customer records are stored in a DataFrame and later exported as customers.csv.

1. **Generating Policies**  
   For each customer, the script randomly creates between 1 to 3 policies. Each policy includes:
   * A unique policy\_id.
   * A link to the customer via customer\_id.
   * A policy type, premium amount, sum insured, and a range of dates defining the policy’s active period.

Policies are stored in a DataFrame and exported as policies.csv.

1. **Generating Claims**  
   The script iterates through each policy to generate 0 to 3 claims per policy. Each claim record includes:
   * A unique claim\_id.
   * The policy\_id and customer\_id to link it directly to both the policy and the customer.
   * A claim date, claim amount (capped at a fraction of the sum insured), and a claim status.

Claims are stored in a DataFrame and exported as claims.csv.

1. **Generating Communications**  
   Communications are generated for each customer, with each customer having 1 to 5 communication records. Each record includes:
   * A unique communication\_id.
   * The customer\_id linking it to the customer.
   * A timestamp, communication channel, and reason category (with associated message text and sentiment).
   * An outcome status and an optional related\_claim\_id that may link the communication to one of the customer’s claims (particularly if the communication is a Complaint or Call Log).

These records are stored in a DataFrame and exported as communications.csv.

1. **CSV File Output**  
   Finally, the script writes all DataFrames to CSV files. These CSV files can be used to build a master table in SQL Server or imported into any BI tool for further analysis.

**Column Explanations**

**Customers Table**

* **customer\_id**  
  A unique identifier for each customer. This numeric key is used to link customer records to policies, claims, and communications.
* **first\_name**  
  The customer's given name, used for personalization and identification.
* **last\_name**  
  The customer's family name, which together with the first name helps identify the customer.
* **age**  
  The customer's age at the time of data generation. Useful for segmentation, risk analysis, and demographic reporting.
* **gender**  
  The customer’s gender (e.g., 'M' or 'F'), which can be leveraged for demographic breakdowns and trend analysis.
* **region**  
  The geographic location where the customer resides. In this script, it represents a Kenyan county—with a higher probability for major urban centers (Nairobi, Mombasa, Kisumu, Eldoret, Nakuru) and a smaller chance for other counties. This field is key for location-based segmentation and regional analysis.
* **vip\_status**  
  A boolean flag indicating if the customer is considered a VIP. This affects prioritization in service, eligibility for premium services, or targeted marketing strategies.

**Policies Table**

* **policy\_id**  
  A unique identifier for each insurance policy. It differentiates individual policies and is used to link policy records with claims and communications.
* **customer\_id**  
  The identifier of the customer who owns the policy. It connects the policy back to the corresponding customer in the Customers table.
* **policy\_type**  
  The type of insurance policy (e.g., Health, Motor, Life). This column helps categorize policies and tailor analysis and reporting to specific segments.
* **premium**  
  The amount that the customer pays for the policy. It is important for revenue calculations and risk assessments.
* **sum\_insured**  
  The total coverage amount provided by the policy. This indicates the maximum financial liability of the insurance provider in case of a claim and is used in premium and risk calculations.
* **start\_date**  
  The effective date when the policy begins, used to track policy durations, renewal dates, and eligibility periods for claims.
* **end\_date**  
  The date when the policy expires. Combined with the start\_date, it defines the policy’s active period and can be used to trigger renewals or expirations in analyses.

**Claims Table**

* **claim\_id**  
  A unique identifier for each claim record, used to distinguish individual claims filed against a policy.
* **policy\_id**  
  The identifier of the policy associated with the claim. This links a claim directly to the policy it’s filed under.
* **customer\_id**  
  The identifier of the customer who filed the claim. This direct linkage simplifies customer-level analyses without having to always join through the policy.
* **claim\_date**  
  The date on which the claim was filed. This is used to track claim trends over time and assess policy performance.
* **claim\_amount**  
  The monetary value being claimed, which is a key metric for assessing the financial impact of claims and managing risk.
* **claim\_status**  
  The current status of the claim (e.g., Approved, Rejected, Pending), which helps track the progress and resolution of claims, essential for operational performance monitoring.

**Communications Table**

* **communication\_id**  
  A unique identifier for each communication record. This field is used to track individual interactions between the customer and the insurer.
* **customer\_id**  
  The identifier of the customer who initiated the communication, linking the record back to the customer for a comprehensive view of interactions.
* **timestamp**  
  The date and time when the communication occurred, vital for time-series analysis, measuring response times, and tracking communication history.
* **channel**  
  The medium through which the communication took place (e.g., Phone, Email, In-Person, Chatbot, Social Media). This helps analyze customer preferences for communication channels.
* **reason\_category**  
  The broad category defining the purpose of the communication. Options include:
  + **Complaint:** Issues or dissatisfaction expressed by the customer.
  + **Feedback:** Suggestions, compliments, or general comments about service or products.
  + **Call Log:** Interactions that mimic traditional phone call records, which may also include chatbot interactions when the conversation is transactional.

Using “Call Log” for a Chatbot interaction acknowledges that many chatbot exchanges serve the same function as a call log.

* **message\_text**  
  A short descriptive text capturing the essence of the communication (e.g., “Billing error in latest statement”). This is used for further analysis, including sentiment analysis or natural language processing.
* **sentiment**  
  A basic sentiment indicator (e.g., Positive, Neutral, Negative) assigned to the message\_text. In a production environment, this would typically be derived from an NLP model.
* **outcome**  
  The result or resolution of the communication (e.g., Resolved, Escalated, Pending Info, Acknowledged), indicating how the issue was handled.
* **related\_claim\_id**  
  An optional field linking the communication to a specific claim. This is particularly useful when a customer’s communication directly references an existing claim, ensuring all related interactions are easily traceable.

**Conclusion**

This documentation covers the purpose of the script, the flow of data generation, and a detailed breakdown of every column in each table. The synthetic data generated by this script is designed to help simulate a realistic insurance environment for analysis and dashboarding, enabling a comprehensive 360° view of customer interactions, policies, claims, and communications.

Feel free to adjust the parameters or expand on the logic to match your real-world requirements.