The dataset you have from Kaggle is well-suited for your **Supply Chain Anomaly Detection Dashboard** project. Here's how it can be structured and used effectively:

**1. Feasibility Analysis**

The dataset includes the following tables:

1. **Orders Table**: Order metadata (timestamps, delivery status, etc.).
2. **Order Items Table**: Details of items in each order (product, seller, price, shipping).
3. **Customers Table**: Customer details (location, demographics).
4. **Payments Table**: Payment details (type, value, installments).
5. **Products Table**: Product metadata (category, dimensions, weight).

**Key Insights for Supply Chain Anomaly Detection**:

* **Delivery Performance**: Compare actual delivery timestamps with estimated delivery dates.
* **Cost Analysis**: Analyze shipping charges, product prices, and payment values.
* **Customer Behavior**: Evaluate order frequencies, locations, and spending patterns.
* **Product Trends**: Analyze category-wise product sales and anomalies in demand.

This dataset provides enough granularity and breadth for advanced analysis, predictive modeling, and visualization.

**2. Fact and Dimension Tables**

To design the data model (for ETL and Power BI), divide the dataset into **fact** and **dimension** tables:

**Fact Table**

* **Fact\_Order\_Details**:
  + **Key Columns**: order\_id, order\_item\_id.
  + **Metrics**: price, shipping\_charges, payment\_value.
  + **Time Metrics**: order\_purchase\_timestamp, order\_delivered\_timestamp, order\_estimated\_delivery\_date.

**Dimension Tables**

1. **Dim\_Customers**:
   * **Key Column**: customer\_id.
   * **Attributes**: customer\_zip\_code\_prefix, customer\_city, customer\_state.
2. **Dim\_Products**:
   * **Key Column**: product\_id.
   * **Attributes**: product\_category\_name, product\_weight\_g, product\_length\_cm, product\_height\_cm, product\_width\_cm.
3. **Dim\_Sellers**:
   * **Key Column**: seller\_id.
   * **Attributes**: Aggregated performance metrics like total orders, delivery delays, etc. (to be derived).
4. **Dim\_Orders**:
   * **Key Column**: order\_id.
   * **Attributes**: order\_status, order\_approved\_at.
5. **Dim\_Payment\_Types**:
   * **Key Column**: payment\_type.
   * **Attributes**: Installments, frequency, average value (to be derived).

**3. Data Usage for Your Project**

**a. ETL Workflow**

* **Extraction**:
  + Load raw tables into a database (MySQL or PostgreSQL).
* **Transformation**:
  + Clean and preprocess:
    - Fill missing delivery timestamps for delayed orders.
    - Standardize product dimensions (e.g., convert grams to kilograms).
  + Derive new columns:
    - **Delivery\_Delay** = order\_delivered\_timestamp - order\_estimated\_delivery\_date.
    - **Profit\_Margin** = price - (cost + shipping\_charges).
* **Loading**:
  + Create fact and dimension tables in a clean schema.

**b. Predictive Modeling**

* **Goal**: Predict delays, unusual demand, and shipping cost anomalies.
* **Steps**:
  + Feature engineering:
    - Aggregate delivery performance by seller\_id and product\_category\_name.
    - Extract time-based features (day of the week, month, holiday).
  + Model selection:
    - **Delay Prediction**: Use classification models (e.g., Random Forest).
    - **Anomaly Detection**: Use unsupervised models like Isolation Forest.

**c. Power BI Dashboard**

1. **Visualizations**:
   * KPIs: Total orders, delayed orders, delivery delay average.
   * Trend Charts: Delivery performance over time.
   * Heatmaps: Delay trends by region and product category.
   * Drill-Down: Seller performance and product sales.
2. **Python Integration**:
   * Use Python to display anomaly detection results dynamically.

**4. Steps for Predictive Modeling**

1. **Prepare Data**:
   * Use Python (pandas) for cleaning and feature engineering.
   * Handle missing values (fillna) and create derived features.
2. **Train Models**:
   * **Delay Prediction**:

python

Copy code

from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()

model.fit(X\_train, y\_train)

* + **Anomaly Detection**:

python

Copy code

from sklearn.ensemble import IsolationForest

model = IsolationForest()

model.fit(X)

anomalies = model.predict(X)

1. **Evaluate**:
   * Use metrics like F1-score for classification or silhouette score for clustering.

**5. Generative AI Integration**

* Use GPT to analyze and summarize patterns in anomalies:
  + Example Prompt: “Summarize the delays and trends in delivery data.”
* Output: Display text-based insights alongside Power BI visuals.