# Financial Fraud Detection: Credit Card Transaction Analysis for SecureGuard

SecureGuard Financial Solutions specialises in delivering innovative, real-time solutions to detect and prevent fraud in the financial sector. With the rise of digital payments, rapid fraud detection is vital to reduce risk, maintain customer trust, and protect valuable assets. This report presents a comprehensive workflow for analyzing credit card transaction data using Python, Excel, SQL, and Tableau, supporting SecureGuard’s mission to identify anomalous spending and fraudulent activity.

## 1. Data Pre-processing in Python

### 1.1. Package Installation

pip install pandas

### 1.2. Filtering, Selecting, and Stratified Sampling

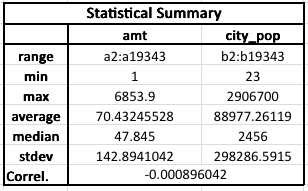
import pandas as pd  
  
# Load the dataset  
df = pd.read\_csv('your\_file.csv')  
  
# Keep necessary columns  
cols\_needed = ['amt', 'city\_pop', 'is\_fraud', 'gender', 'category', 'state', 'job']  
df = df[cols\_needed]  
  
# Clean data  
df = df[df['amt'] > 0]  
df = df[df['gender'].notnull()]  
df = df.drop\_duplicates()  
  
# Stratified sample: 5% from each 'category', retaining small groups  
optimum\_frac = 0.05  
stratified\_sample = df.groupby('category', group\_keys=False).apply(  
 lambda x: x.sample(frac=optimum\_frac, random\_state=42) if len(x) > 20 else x  
).reset\_index(drop=True)  
  
# Export for Excel analysis  
stratified\_sample.to\_csv('stratified\_sample.csv', index=False)

## 2. Data Exploration & Analysis in Excel Online

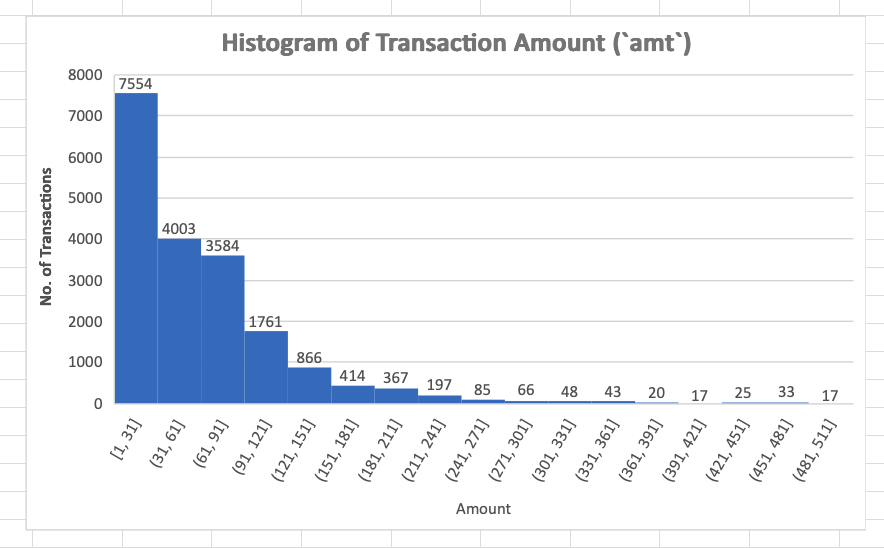
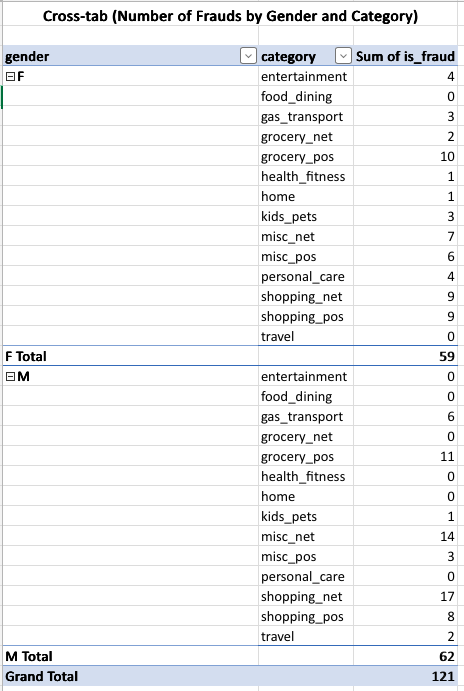
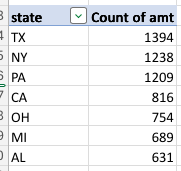
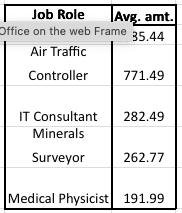
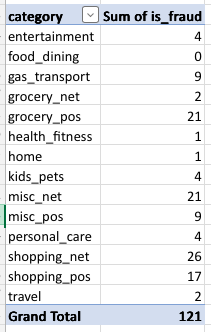
### 2.1. Import Cleaned Data

* Upload stratified\_sample.csv to Excel Online for exploration.

### 2.2. Statistical Overview

* Calculate min, max, average, median, and stdev for amt and city\_pop.
* Add formulae such as =MAX(A2:A1000) as needed. 

### 2.3. Visual Exploration

* Plot histogram for amt. 
* Use pivot tables for:
  + Fraud count by gender and category
  + 
  + Screenshot 2025-08-18 at 11.09.00.png
  + Top 3 states by transaction count
  + 
  + Screenshot 2025-08-24 at 11.16.59.png
  + Average transaction amount by job
  + 
  + Screenshot 2025-08-24 at 11.17.26.png
  + Fraud count by category
  + 
  + Screenshot 2025-08-24 at 11.17.12.png

### 2.4. Insights

* Most transactions are low-value with few outliers.
* Fraud most common in shopping and grocery categories.
* Certain professions have higher average amounts.
* Texas, New York, Pennsylvania lead in transaction count.

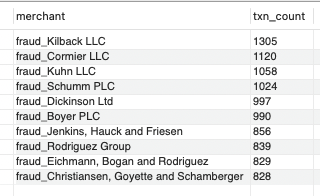
## 3. Data Analysis with SQL

### 3.1. Schema and Loading

CREATE SCHEMA finance;  
USE finance;  
-- Create cc\_data and location\_data tables matching CSVs

### 3.2. Key SQL Queries

-- 1. Total transactions  
SELECT COUNT(\*) FROM cc\_data;  
  
-- 2. Top merchants  
SELECT merchant, COUNT(\*) AS txn\_count FROM cc\_data GROUP BY merchant ORDER BY txn\_count DESC LIMIT 10;  
  
-- 3. Avg. transaction by category  
SELECT category, AVG(amt) FROM cc\_data GROUP BY category;  
  
-- 4. Count & % fraud  
SELECT COUNT(\*) AS total, SUM(is\_fraud), ROUND(100.0 \* SUM(is\_fraud) / COUNT(\*), 2) FROM cc\_data;

Screenshot 2025-08-19 at 12.16.10.png Screenshot 2025-08-19 at 11.31.01.png

* Join cc\_data & location\_data to get Geo-coordinates for mapping.

SELECT  
 cc.trans\_num,  
 cc.cc\_num,  
 cc.city,  
 cc.state,  
 loc.lat,  
 loc.long AS long\_  
FROM cc\_data cc  
LEFT JOIN location\_data loc  
 ON cc.cc\_num = loc.cc\_num  
WHERE loc.lat IS NOT NULL AND loc.long IS NOT NULL;



Screenshot 2025-08-19 at 11.50.20.png

* Find city with highest population and transaction activity.

SELECT city, state, city\_pop  
FROM cc\_data  
ORDER BY city\_pop DESC  
LIMIT 1;  
  
SELECT city, state, COUNT(\*) AS txn\_count  
FROM cc\_data  
GROUP BY city, state  
ORDER BY txn\_count DESC  
LIMIT 1;

Screenshot 2025-08-19 at 11.51.51.pngScreenshot 2025-08-19 at 11.53.59.png

* Extract earliest/latest transaction dates.

SELECT  
 MIN(trans\_date\_trans\_time) AS earliest\_txn,  
 MAX(trans\_date\_trans\_time) AS latest\_txn  
FROM cc\_data;

Screenshot 2025-08-19 at 11.54.17.png

Screenshot 2025-08-19 at 11.54.17.png

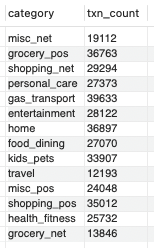
* Summarise transaction total, category counts, and average amount by gender or day of week.

SELECT SUM(amt) AS total\_spent FROM cc\_data;

Screenshot 2025-08-19 at 11.54.35.png

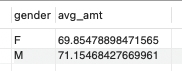
Screenshot 2025-08-19 at 11.54.35.png

SELECT category, COUNT(\*) AS txn\_count  
FROM cc\_data  
GROUP BY category;



Screenshot 2025-08-19 at 11.54.47.png

SELECT gender, AVG(amt) AS avg\_amt  
FROM cc\_data  
GROUP BY gender;



Screenshot 2025-08-19 at 11.55.01.png

SELECT  
 DAYNAME(STR\_TO\_DATE(trans\_date\_trans\_time, '%d-%m-%Y %H:%i')) AS day\_of\_week,  
 AVG(amt) AS avg\_amt  
FROM cc\_data  
GROUP BY day\_of\_week  
ORDER BY avg\_amt DESC  
LIMIT 1;

Screenshot 2025-08-19 at 11.57.16.png

Screenshot 2025-08-19 at 11.57.16.png

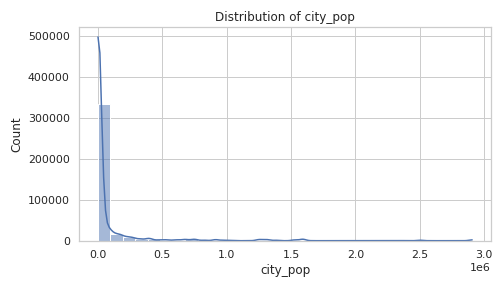
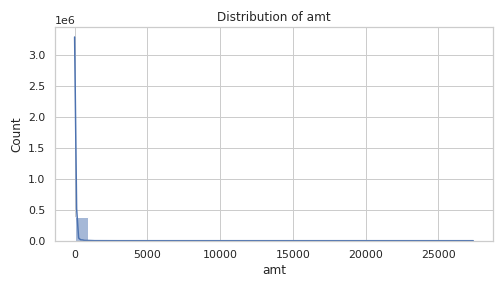
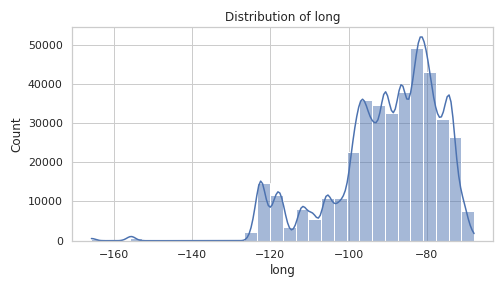
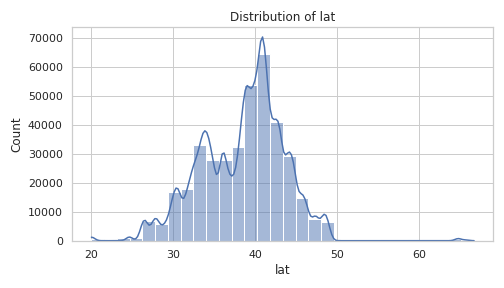
## 4. Exploratory Data Analysis (Python / Jupyter)

### 4.1. Dataset Dimensions

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
import os  
  
# Create directory for saving plots  
os.makedirs('plots', exist\_ok=True)  
  
# Visualization style  
sns.set(style="whitegrid")  
  
# Load dataset  
df = pd.read\_csv('cc\_data.csv')  
  
print('Rows, columns:', df.shape)

### 4.2. Unique Categorical Values & Distribution Plots

for col in df.select\_dtypes(include=['object', 'category']):  
 print(f"{col}: {df[col].nunique()} unique")  
# Histograms, boxplots, KDE for numerical columns  
num\_cols = df.select\_dtypes(include=[np.number]).columns  
df[num\_cols].hist(bins=30, figsize=(15, 10))  
plt.suptitle('Numerical Feature Distributions')  
plt.tight\_layout()  
plt.show()  
  
# Focused distributions with KDE and save plots  
num\_plot\_cols = ['amt', 'city\_pop', 'lat', 'long']  
for col in num\_plot\_cols:  
 plt.figure(figsize=(7,4))  
 sns.histplot(df[col], bins=30, kde=True)  
 plt.title(f'Distribution of {col}')  
 plt.xlabel(col)  
 plt.ylabel("Count")  
 plt.tight\_layout()  
 plt.savefig(f'plots/{col}\_distribution.png')  
 plt.show()

trans\_date\_trans\_time: 293627 unique values merchant: 693 unique values category: 14 unique values first: 352 unique values last: 481 unique values gender: 2 unique values street: 979 unique values city: 890 unique values state: 51 unique values job: 492 unique values dob: 964 unique values trans\_num: 389002 unique values.  

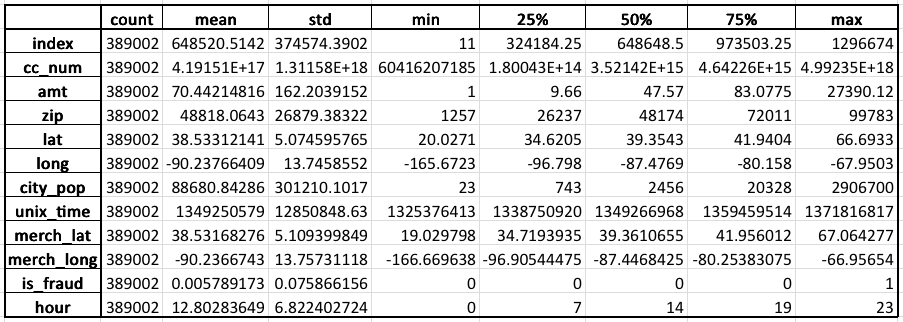
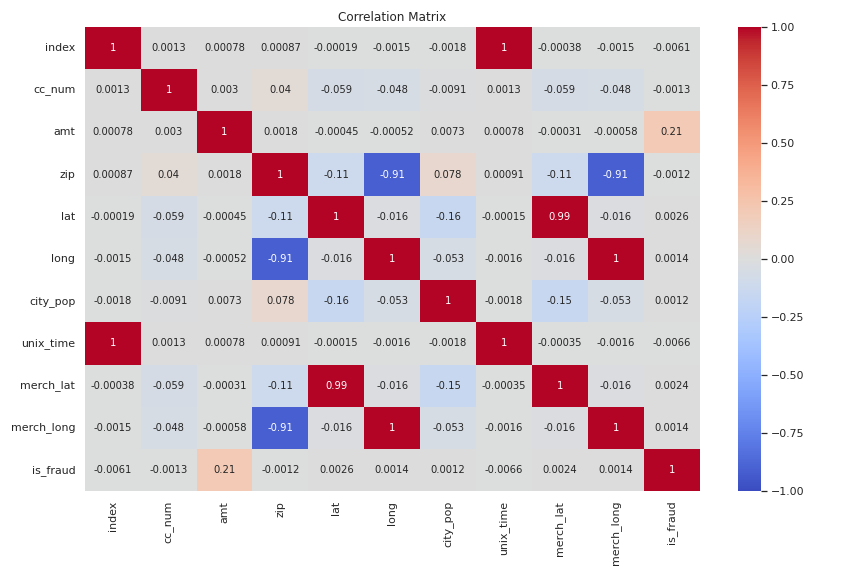
* Visualize and check for missing values, outliers, and skew-ness.

print(df.isnull().sum())  
# No missing values detected. If needed, we could impute or drop missing data here.

index 0 trans\_date\_trans\_time 0 cc\_num 0 merchant 0 category 0 amt 0 first 0 last 0 gender 0 street 0 city 0 state 0 zip 0 lat 0 long 0 city\_pop 0 job 0 dob 0 trans\_num 0 unix\_time 0 merch\_lat 0 merch\_long 0 is\_fraud 0 dtype: int64.

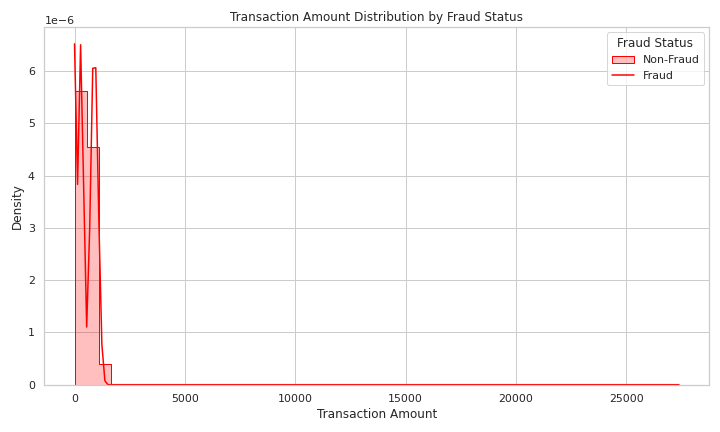
* Compute summary stats and correlation matrix.

display(df.describe().T)  
  
plt.figure(figsize=(12,8))  
sns.heatmap(df[num\_cols].corr(numeric\_only=True), annot=True, cmap='coolwarm', vmin=-1, vmax=1)  
plt.title("Correlation Matrix")  
plt.tight\_layout()  
plt.savefig('plots/correlation\_matrix.png')  
plt.show()  
# Correlation values closer to +1 or -1 indicate strong relationships.

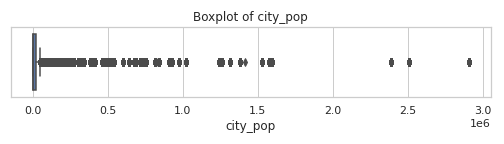
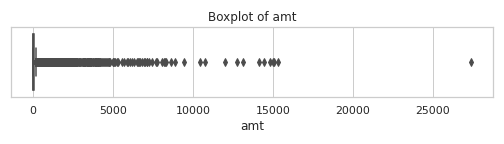
 

* Split/compare distributions by is\_fraud, gender, or category.

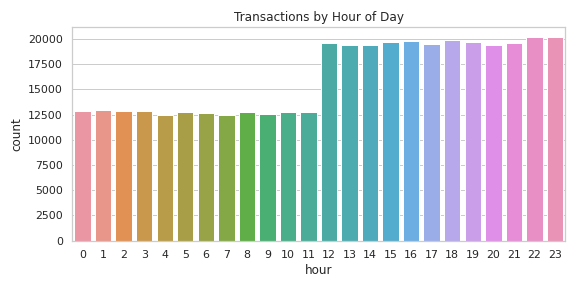
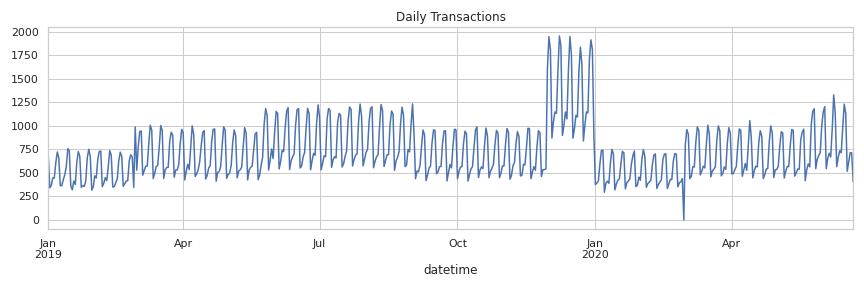
plt.figure(figsize=(10,6))  
sns.histplot(  
 data=df,  
 x='amt',  
 hue='is\_fraud',  
 hue\_order=[1],  
 bins=50,  
 kde=True,  
 element='step',  
 stat='density',  
 palette={0: 'blue', 1: 'red'}  
)  
plt.title('Transaction Amount Distribution by Fraud Status')  
plt.xlabel("Transaction Amount")  
plt.ylabel("Density")  
plt.legend(title="Fraud Status", labels=["Non-Fraud", "Fraud"])  
plt.tight\_layout()  
plt.savefig('plots/amt\_fraud\_hist.png')  
plt.show()

 - Identify and count outliers.

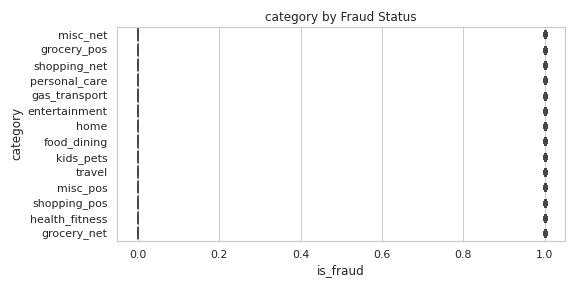
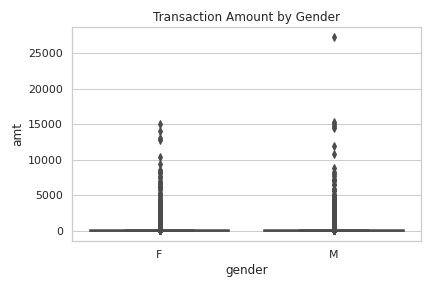
for col in ['amt', 'city\_pop']:  
 plt.figure(figsize=(7,2))  
 sns.boxplot(x=df[col])  
 plt.title(f'Boxplot of {col}')  
 plt.tight\_layout()  
 plt.savefig(f'plots/{col}\_boxplot.png')  
 plt.show()  
  
Q1 = df['amt'].quantile(0.25)  
Q3 = df['amt'].quantile(0.75)  
IQR = Q3 - Q1  
outliers = df[(df['amt'] < (Q1 - 1.5\*IQR)) | (df['amt'] > (Q3 + 1.5\*IQR))]  
print(f"{len(outliers)} outlier(s) detected in amt")

 -Analyse trends over time (daily/hourly/monthly).

# Convert to datetime  
df['datetime'] = pd.to\_datetime(df['trans\_date\_trans\_time'], format='%d-%m-%Y %H:%M')  
  
# Daily transaction counts  
plt.figure(figsize=(12,4))  
df.set\_index('datetime').resample('D').size().plot(title="Daily Transactions")  
plt.tight\_layout()  
plt.savefig('plots/daily\_transactions.png')  
plt.show()  
  
# Extract hour and plot hourly transaction counts  
df['hour'] = df['datetime'].dt.hour  
plt.figure(figsize=(8,4))  
sns.countplot(x='hour', data=df)  
plt.title("Transactions by Hour of Day")  
plt.tight\_layout()  
plt.savefig('plots/hour\_transactions.png')  
plt.show()

 - Segment-wise (e.g. by job or location) comparisons.

# Transaction amount by gender  
plt.figure(figsize=(6,4))  
sns.boxplot(x='gender', y='amt', data=df)  
plt.title("Transaction Amount by Gender")  
plt.tight\_layout()  
plt.savefig('plots/gender\_amt\_boxplot.png')  
plt.show()  
  
# Fraud rate by category  
plt.figure(figsize=(10,4))  
fraud\_rates = df.groupby('category')['is\_fraud'].mean().sort\_values(ascending=False)  
fraud\_rates.plot(kind='bar', color='crimson', title='Fraud Rate by Transaction Category')  
plt.ylabel("Fraud Rate")  
plt.tight\_layout()  
plt.savefig('plots/category\_fraud\_rate.png')  
plt.show()



### 4.3. Key EDA Insights

* Transaction values are highly skewed; a few large outliers.
* Fraud concentrated in specific categories.
* Geographic, time, and group-based analysis reveal patterns useful for fraud detection.

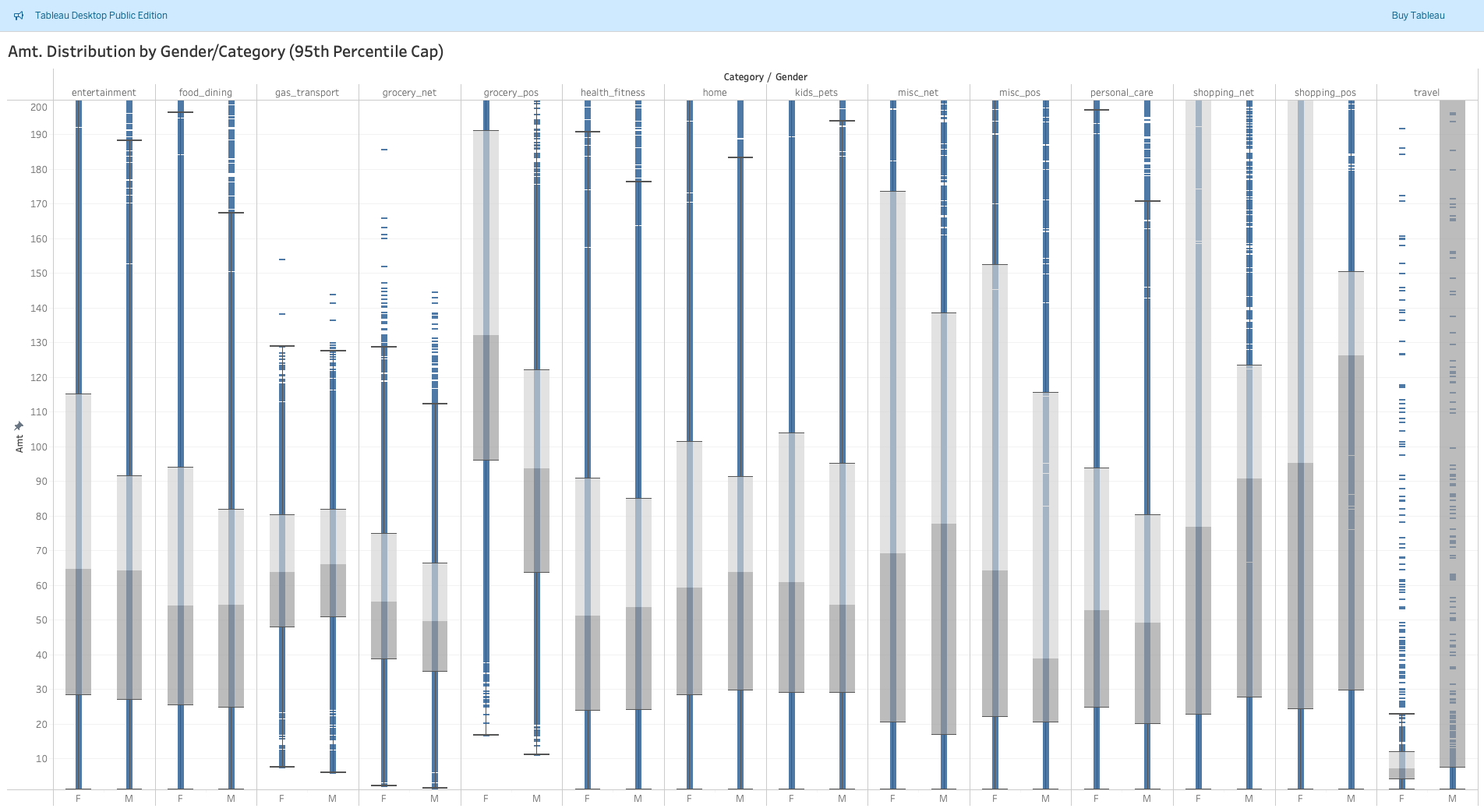
## 5. Major Visual Data Insights & Interactive Reporting (Tableau)

### 5.1 Workflow Overview

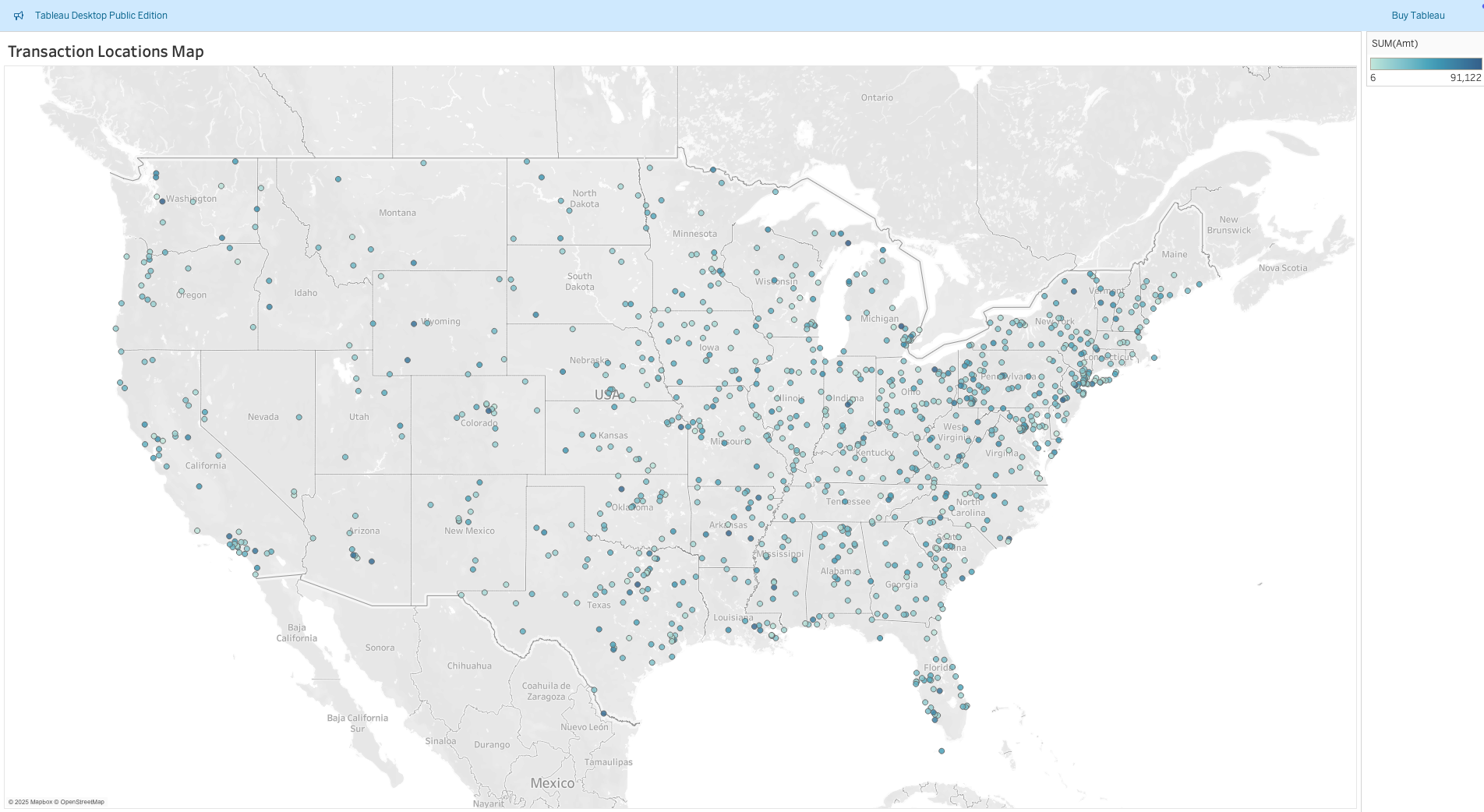
Interactive Tableau dashboards are created for in-depth fraud analysis and transparency, enabling dynamic data slicing and stakeholder exploration.

### 5.2 Step-by-Step Tableau Implementation

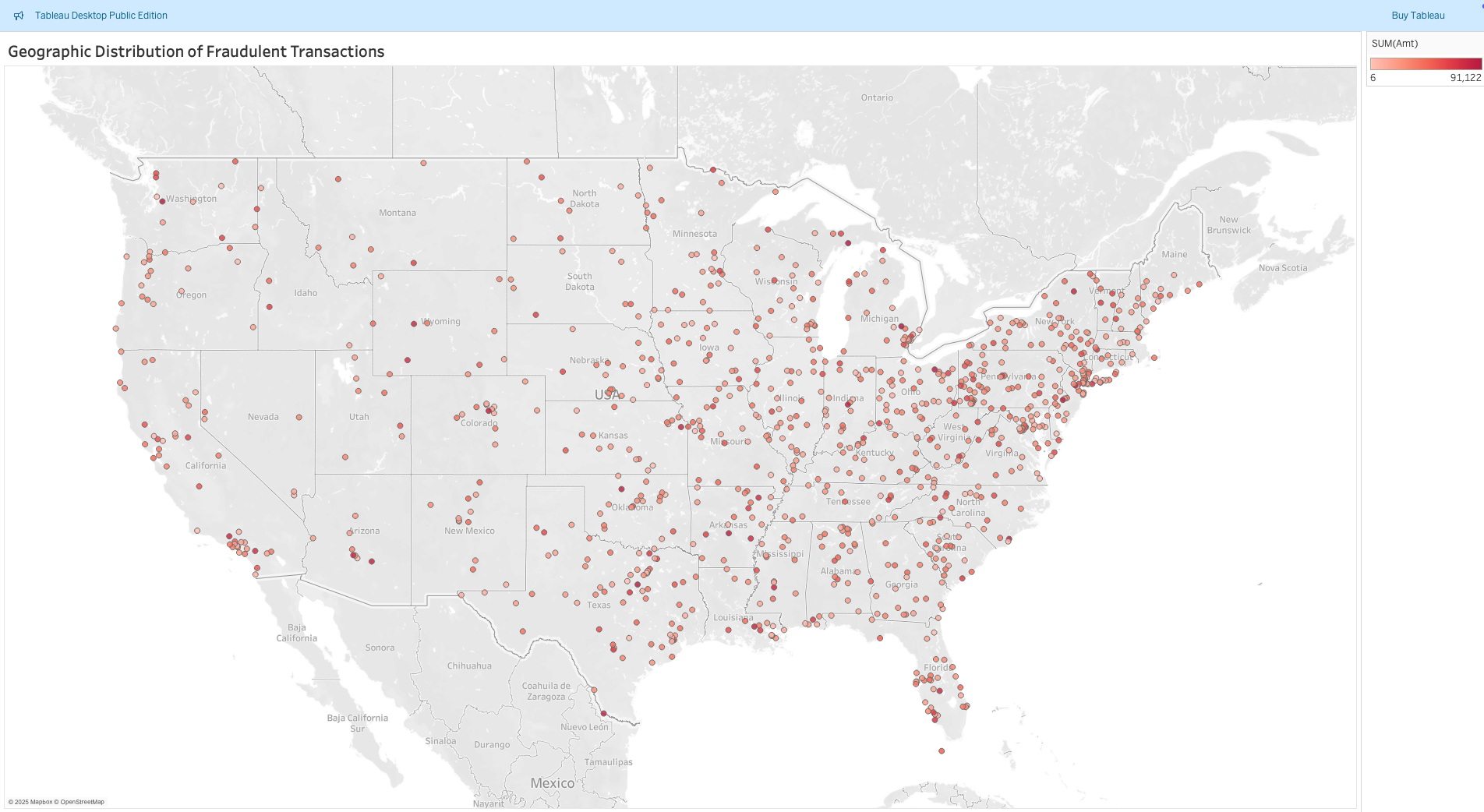
#### A. Box & Whisker Plot: Transaction Amount by Gender & Category

1. New worksheet: drag category (Columns), amt (Rows), and gender (Color/Columns).
2. Select “Box-and-Whisker Plot” in Show Me.
3. Edit tooltips, labels, and titles for clarity. 

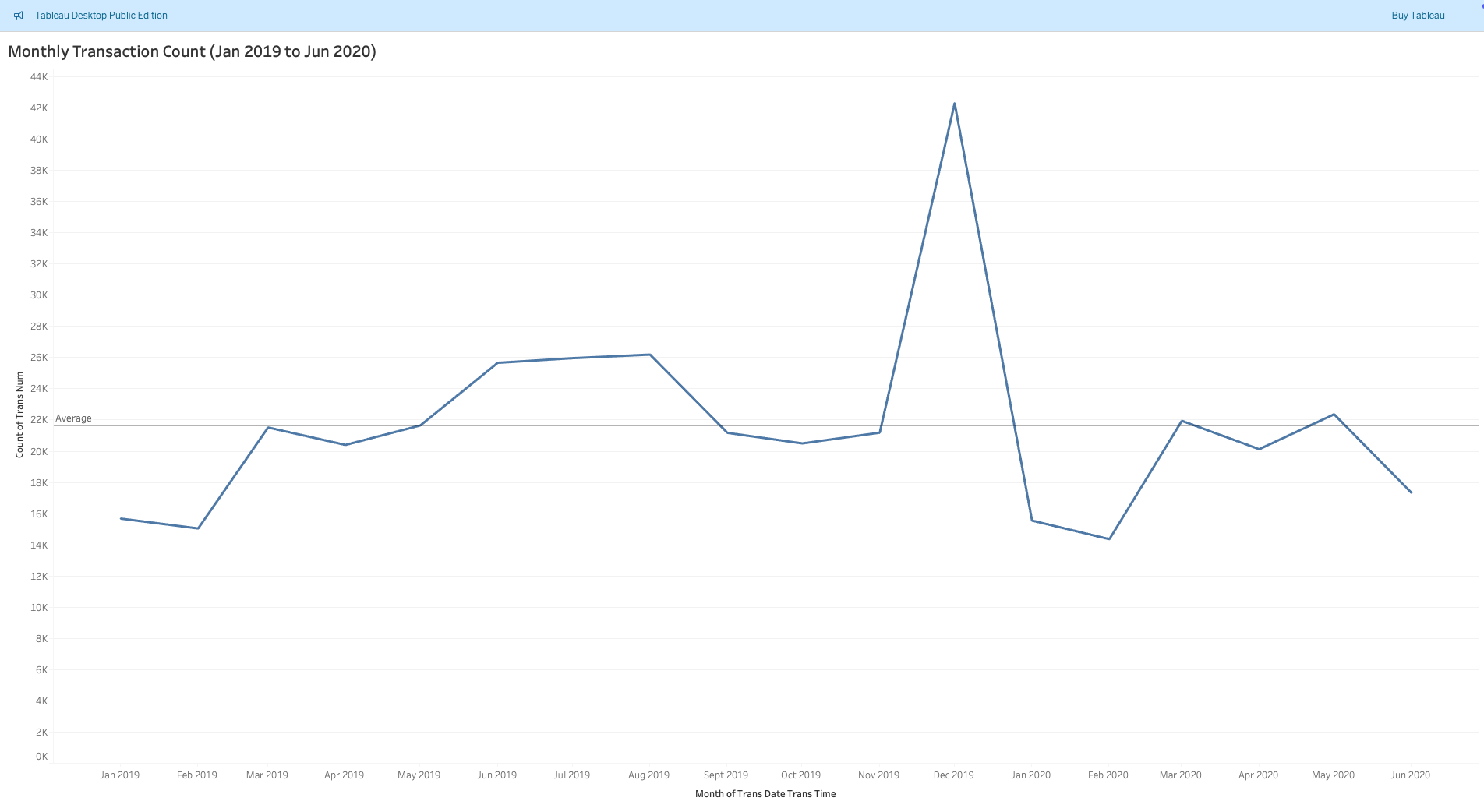
#### B. Map Visualization: All Transaction Locations

1. Ensure lat and long are geographic.
2. Drag lat (Rows) and long (Columns) for map.
3. Show all transactions; optionally size/color by amt. 

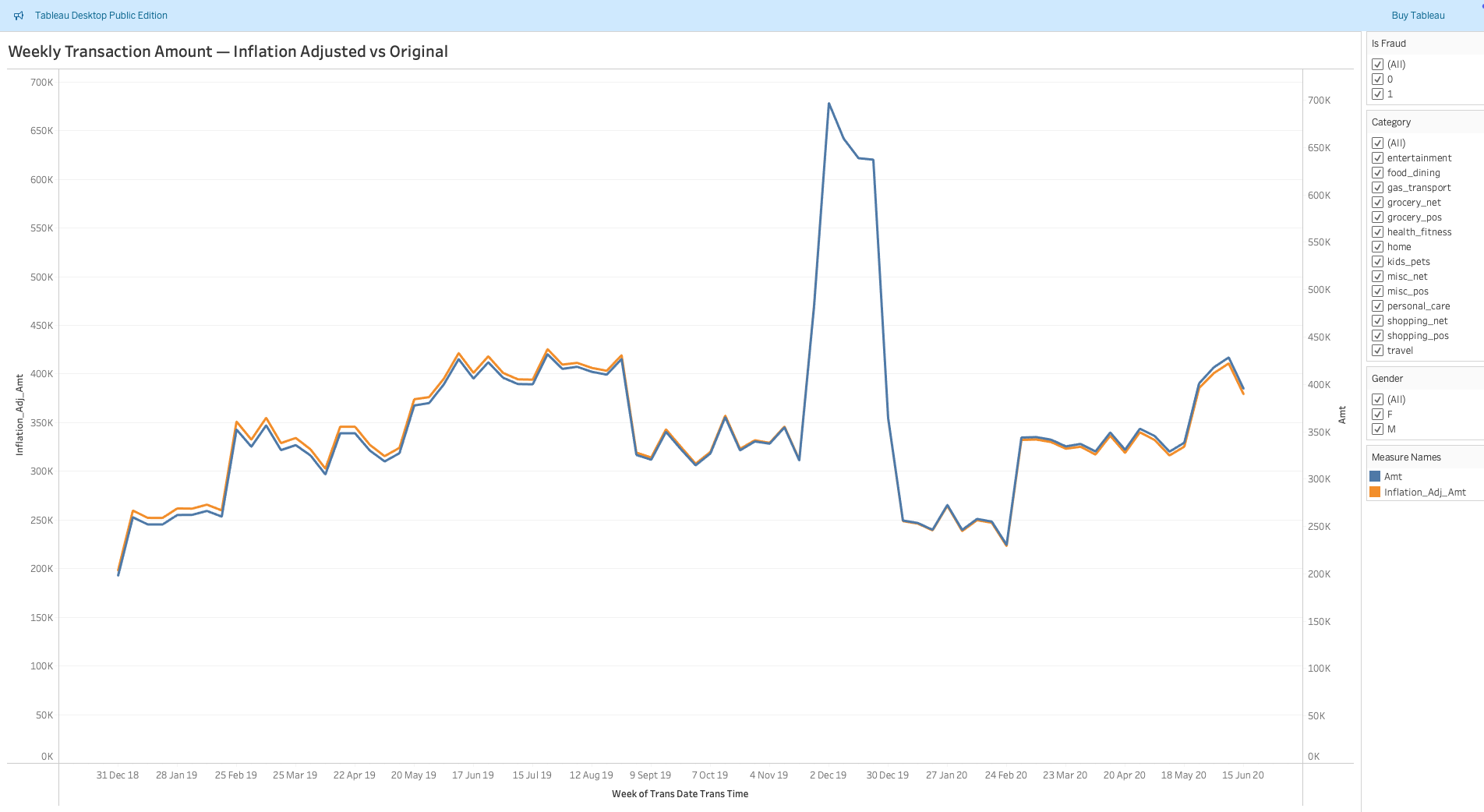
#### C. Fraud Map

1. Duplicate location map.
2. Filter by is\_fraud = 1.
3. Use red for points; enhance tooltips. 

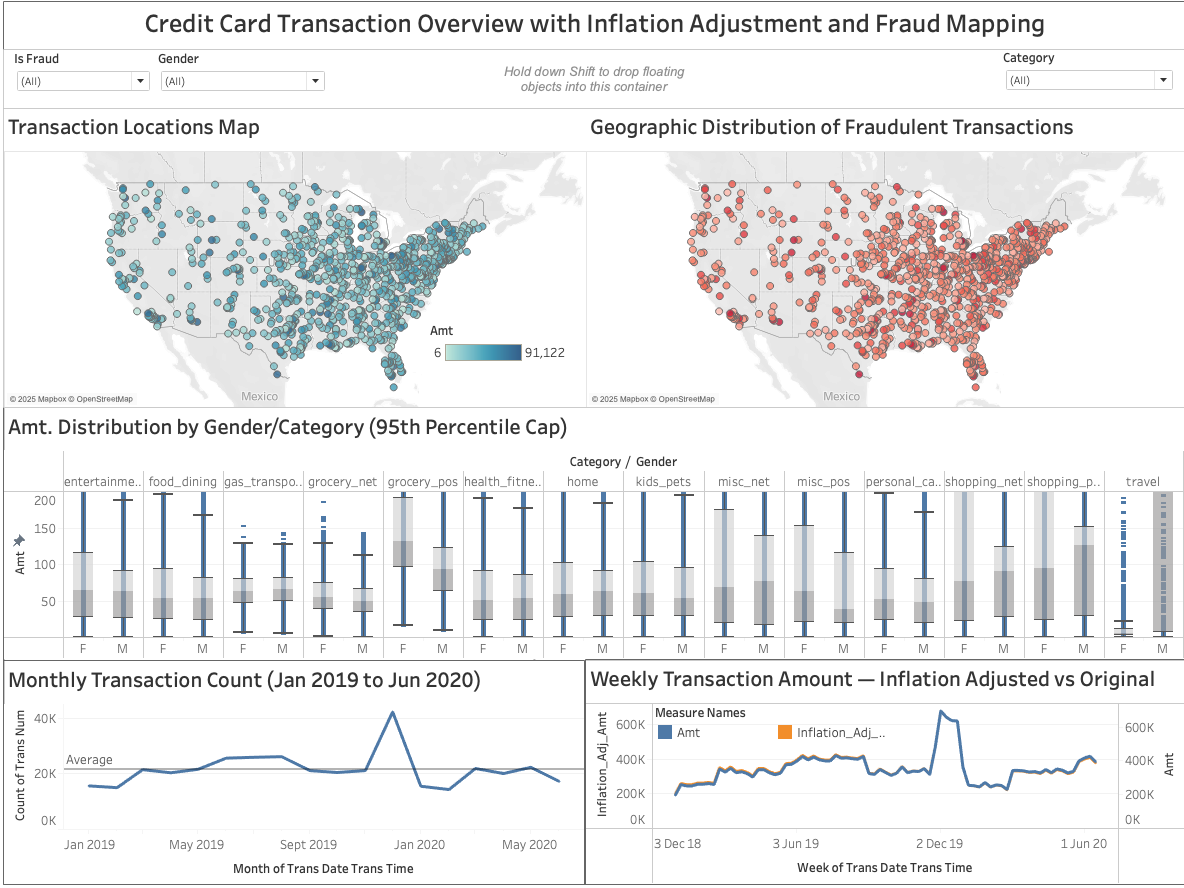
#### D. Monthly Trend Chart

1. Drag trans\_date\_trans\_time (Columns), set to Month.
2. Drag trans\_num (Rows, as COUNT).
3. Use Line chart, label axes. 

#### E. Inflation Adjustment Analysis

1. Create Inflation\_Adj\_Amt field: [Amt] / POWER(1 + 0.03, DATEDIFF('month', DATE("2019-01-01"), DATETRUNC('month', [Trans Date Trans Time])) / 12)
2. Chart by week: use line(s) for SUM(adjusted) and SUM(original) amounts. 

#### F. Interactive Dashboard Assembly

1. Create new Dashboard.
2. Drag and arrange all key worksheets.
3. Add slicers for Gender, Category, Is\_Fraud:
   * Show filter for each field.
   * Use “Apply to all worksheets using this data source.”
4. Add titles, descriptions, and adjust for clarity.
5. Format tooltips; add project/author info.
6. Publish via “Save to Tableau Public As…” 

## 6. Summary and Recommendations

The exploratory data analysis and visualisation efforts have yielded a comprehensive understanding of the credit card transaction dataset and the patterns indicative of fraudulent behavior. Below is a detailed summary of key findings and actionable recommendations aimed at strengthening SecureGuard’s fraud detection capabilities.

### Key Findings

* **Skewed Transaction Amounts:** Transaction values are heavily right-skewed, with most transactions being low-value and a small number of high-value outliers. These extreme values have significant implications for modeling and anomaly detection.
* **Fraud Concentration:** Fraudulent transactions constitute less than 1% of all transactions, predominantly occurring in categories such as online shopping (shopping\_net), grocery POS, and miscellaneous networks. The gender distribution among fraud cases is nearly even, but crime patterns vary subtly by category and gender.
* **Geographical Clustering:** Spatial analysis revealed clusters of fraudulent activity, especially in high-transaction volume states such as Texas, New York, and Pennsylvania. Geographic insights can inform region-specific monitoring and intervention strategies.
* **Temporal Patterns:** Volume fluctuations across time demonstrate seasonality and trend effects. Inflation adjustments reveal genuine changes in transaction values beyond economic inflation, enhancing temporal models.
* **Predictive Feature Identification:** Correlations and distributional differences highlight features such as transaction amount, transaction timestamp (hour), geographic coordinates, and category as valuable inputs for fraud predictive models.

### Recommendations

1. **Outlier Treatment:** Develop strategies—such as transformations or capping—to mitigate the influence of extreme transaction and city population values in predictive modeling and reporting.
2. **Focus on High-Risk Categories:** Allocate analytical and monitoring resources disproportionately to categories with elevated fraud risk (e.g., online shopping), employing adaptive rules and machine learning models attuned to these segments.
3. **Leverage Geographic Insights:** Integrate spatial fraud patterns into real-time monitoring systems, enabling SecureGuard to deploy localised alerts, investigations, and possibly enhanced verification in fraud hot-spots.
4. **Incorporate Inflation and Time Trends:** Adjust for inflation and seasonal effects within fraud detection frameworks to ensure historical comparisons and trends reflect true transactional risk changes.
5. **Deploy Interactive Dashboards:** Utilise the Tableau dashboards designed in this pipeline to empower analysts with dynamic filters for gender, category, and fraud status, improving anomaly investigation turnaround and transparency.
6. **Further Enhancements:**
   * Expand data sources to include merchant risk profiles, customer demographics, and behavioural signals.
   * Explore and validate advanced machine learning models leveraging the identified predictive features.
   * Establish ongoing feedback loops from fraud investigations to continuously refine detection rules and model accuracy.