# High-level overview

Financial data is processed in comma separated rows which each row representing 1 event e.g. buying groceries would be counted as one event or transaction and buying gas on the way home another event, similar to what you see in your banking app.

Usually these are in CSV (comma separated value) files so it's easy to parse data through programs, ML, or rule engines.

# What is included in financial fraud data?

Each row of financial data typically has identifiers (IDs) as follows:

- **transaction_id**: Unique transaction identifier (string)
- **timestamp**: UTC timestamp in ISO 8601 (e.g., 2025-10-03T12:34:56Z)
- **account_id**: Account receiving/holding funds (internal id)
- **payer_id**: Entity initiating payment (could equal account holder)
- **payee_id**: Counterparty receiving payment (merchant or person)
- **amount**: Numeric amount (decimal) in listed currency
- **currency**: ISO currency code (USD, EUR, etc.)
- **merchant_category**: High-level merchant category (grocery, wire, airfare, etc.)
- **country**: 2-letter country code where transaction originated or payee is located
- **channel**: How the tx was made (app, web, mobile, branch, atm, wire)
- **device_id**: Device identifier (hashed) used for the transaction
- **ip_hash**: Hashed IP or token representing client network
- **balance_before**: Account balance immediately before the transaction
- **balance_after**: Balance immediately after the transaction
- **label**: Ground-truth label for testing (0=clean, 1=suspicious, 2=fraud)
- **notes**: Short human-readable note (**NOT used in real systems, but for our sanity, added so we can tell what should be flagged as financial fraud to test accuracy of the system**)

*ISO 20022 source here
The ISO 20022 is the international standard for how institutions structure electronic financial data. It will be beyond the scope of this project but I was able to cherry pick the above data IDs from it.

# What fraud might look like

1. **Large amounts**: The example Sarah gave us was $10,000 which is instantly reported to the IRS as potentially fraud, especially if done in succession. Account transaction history is usually weighed on here so if you're very wealthy or a business owner and regularly make large transactions you are less likely to flag suspicion.
2. **Transaction volume spikes**: Spending/receiving many transactions in a short period of time (e.g. 20 small transactions in a couple minutes ($1, $2, or $5)
3. **Sending to new payee**: Older people fall for this all the time, super easy to check and flag if even remotely suspicious.

## More complicated/difficult to track forms of fraud

1. **High-risk countries**: Prevent scam via call centers and other common scam techniques. Check IP addresses of the sender and payee?
2. **Money Laundering**: refund then withdrawal patterns, technically a non-problem because it will be very obvious retrospectively once balances are settled but very helpful to prevent from happening in the first place.

## Other potential fraud to include

1. **Impossible balances**: Things like negative balances or checking for many double spends beyond what normally might happen (accidently getting charged twice for one item)
2. **Suspicious IP/device behavior**: Very hard to do without being incredibly invasive. Requires checking a users typical IP address/region and device ID/fingerprint to see if there are geolocation mismatches (e.g. Arizona based customer is logged in in europe suddenly. Could be a trip. Would have to factor in things like login history to prevent false positives).

*Above data sourced from here

# Example synthetic fraud data

| Format | **transaction_id,timestamp,account_id,payer_id,payee_id,amount,currency,merchant_category,country,channel,device_id,ip_hash,balance_before,balance_after,label,notes** |
|---|---|
| EX1 | tx0001,2025-09-29T08:12:22Z,acct_1001,user_1001,merchant_9001,12.50,USD,grocery,US,app,dev_a1,haship1,100.00,87.50,0,normal weekly grocery |
| EX2 | tx0002,2025-10-01T22:05:03Z,acct_1001,user_1001,merchant_9002,9500.00,USD,investment,US,web,dev_a1,haship1,15000.00,5500.00,2,large one-off transfer to new payee (obvious fraud) |
| EX3 | tx0003,2025-10-02T02:15:09Z,acct_2002,user_2002,merchant_9003,4.99,USD,digital_goods,US,mobile,dev_b3,haship2,30.00,25.01,1,repeated micro-charges pattern |
| EX4 | tx0004,2025-10-02T02:16:01Z,acct_2002,user_2002,merchant_9004,4.99,USD,digital_goods,US,mobile,dev_b3,haship2,25.01,20.02,1,another micro-charge within 60s |
| EX5 | tx0005,2025-09-28T14:40:00Z,acct_3003,user_3003,merchant_9005,1200.00,USD,airfare,FR,web,dev_c5,haship3,5000.00,3800.00,0,normal vacation charge (non-fraud) |
| EX6 | tx0006,2025-10-02T11:30:00Z,acct_4004,user_4004,merchant_9006,15000.00,USD,wire,NG,web,dev_d7,haship4,20000.00,5000.00,2,high-amount wire to high-risk country (obvious fraud) |
| EX7 | tx0007,2025-10-02T11:31:10Z,acct_4004,user_4004,merchant_9007,14990.00,USD,wire,NG,web,dev_d8,haship5,5000.00,-9990.00,2,double large wire + negative balance (data inconsistency/fraud signal) |
| EX8 | tx0008,2025-10-03T09:00:00Z,acct_5005,user_5005,merchant_9008,75.00,USD,subscription,US,app,dev_e2,haship6,300.00,225.00,0,monthly software fee |

# Python script for basic fraud filtering and detection (detect_fraud.py)

Simple python script to flag potentially fraudulent rows based on some simple rules:

1. If the amount is >$5000
2. If it's in a high-risk country
3. If the balance after the transaction is negative

## There are also some ‘soft’ rules:

1. Many transactions repeated in a short time
2. Lots of transaction velocity in say a minute

## When the script it ran with rows of transaction data, it returns:

- **id**: transaction id carried over from input
- **pred_lable**: The fraud score assigned to the transaction: 0 is clean, 1 is suspicious, 2 is very likely fraud.
- **rule**: what rule triggered the above classification
- **amount**: transaction amount
- **account_id**: the account making/receiving transaction
- **timestamp**: when was the transaction
- **notes**: for example/project purposes to help make sure we are correctly identifying fraud (also explained above)

**It will look like this:**
id,pred_label,rule,amount,account_id,timestamp,notes

**Example:**
tx0001,0,clean,12.50,acct_1001,2025-09-29T08:12:22Z,normal weekly grocery

# Python script to synthesize financial data (generate_transactions.py)

In the file you can specify how many transactions you would like to generate by modifying this line:
**transactions = [generate_transaction(i, base_time) for i in range(1, 10001)]**
Change the range accordingly, the above example will generate 10,000 transactions. If you wanted 1,000 the upper bound would be 1001. Pretty self-explanatory.

The other line worth changing for brevity is this one:
**with open("transactions_10000.csv", "w", newline='', encoding="utf-8") as csvfile:**
It is just the csv file name so change ‘**transactions_10000**’ to an identifier easier to understand.

Works by assigning a fraud type from 3 broad categories:

- Not suspicious (~70%)
- Suspicious (~20%)
- Definitely fraud (10%)

These are overestimates–I imagine real fraud estimates are FAR lower than 10%, but that is just a guess–RESEARCH THIS.

Then the script assigns transaction type based on fraud type, assigns a label, and calculates balances (checks for things like negative balances) and writes the output to a csv file.