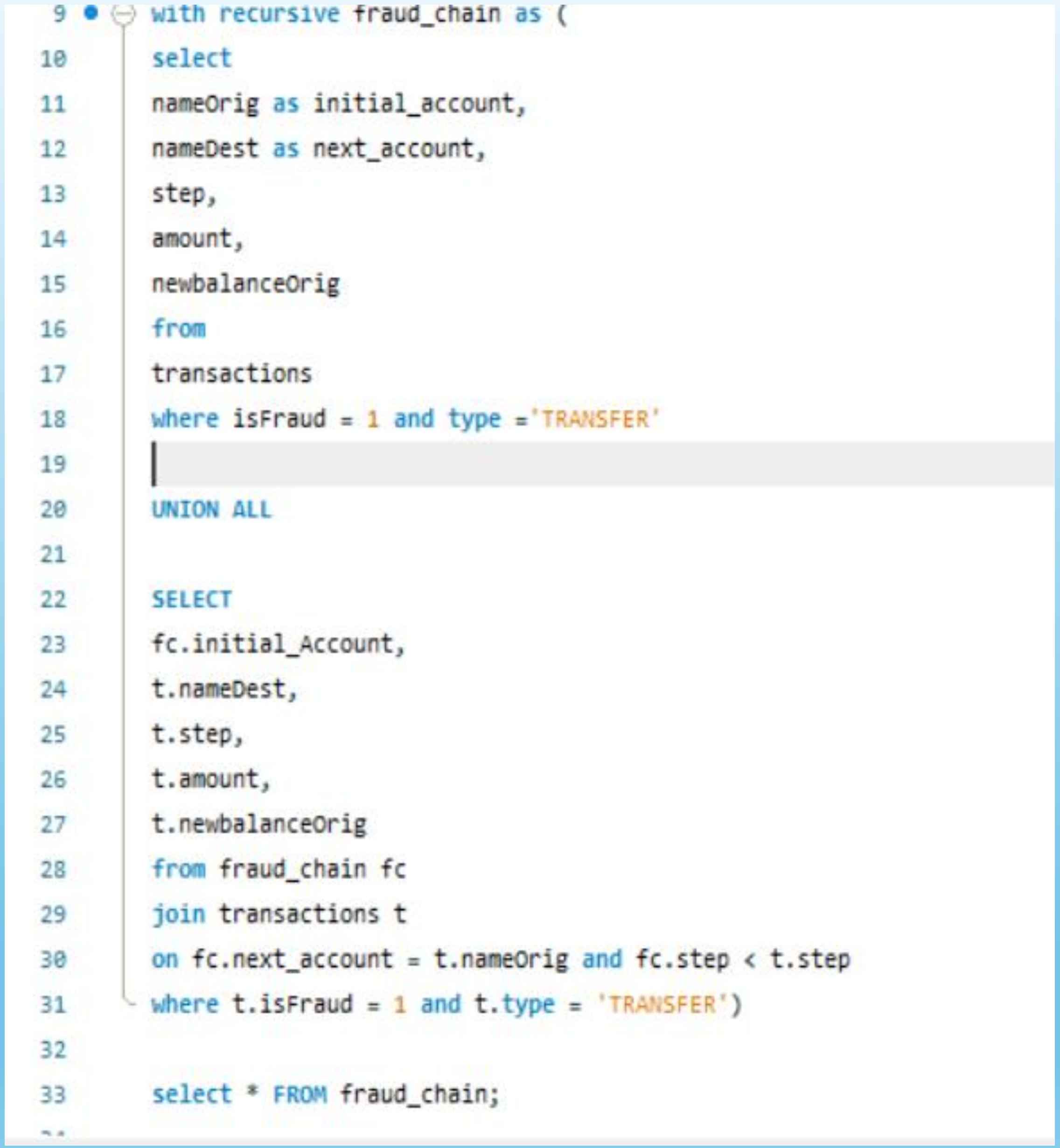
SQL DATA Analyst Project

By Sofia Crasto

1. Detecting Recursive Fraudulent Transactions

Question:
Use a recursive CTE to identify potential money laundering chains where money is transferred from one account to another across multiple steps, with all transactions flagged as fraudulent.

Solution:
This query uses a recursive CTE to track the flow of money through multiple accounts over successive steps. The recursive part of the CTE allows us to follow the chain of transactions and identify patterns that could indicate money laundering activities. It filters out chains where all transactions are marked as fraudulent.

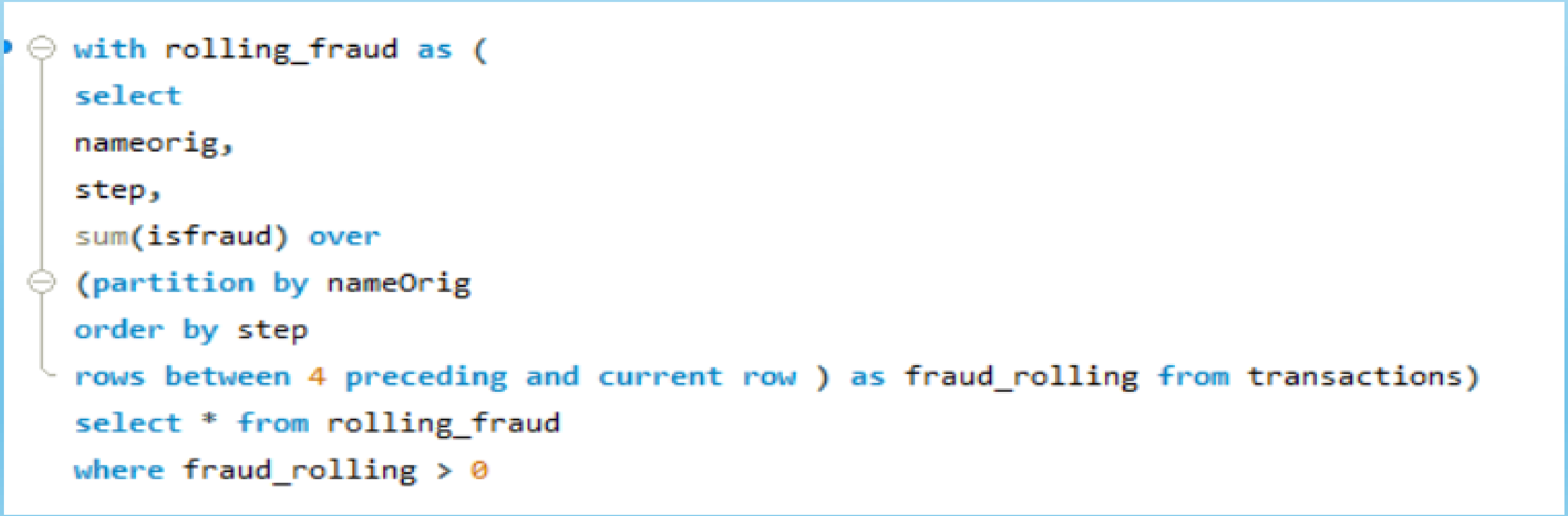


```sql
with recursive fraud_chain as (
select
nameOrig as initial_account,
nameDest as next_account,
step,
amount,
newbalanceOrig
from
transactions
where isFraud = 1 and type ='TRANSFER'


UNION ALL


SELECT
fc.initial_Account,
t.nameDest,
t.step,
t.amount,
t.newbalanceOrig
from fraud_chain fc
join transactions t
on fc.next_account = t.nameOrig and fc.step < t.step
where t.isFraud = 1 and t.type = 'TRANSFER')

select * FROM fraud_chain;
```

2. Analyzing Fraudulent Activity over Time

Question:
Use a CTE to calculate the rolling sum of fraudulent transactions for each account over the last 5 steps.

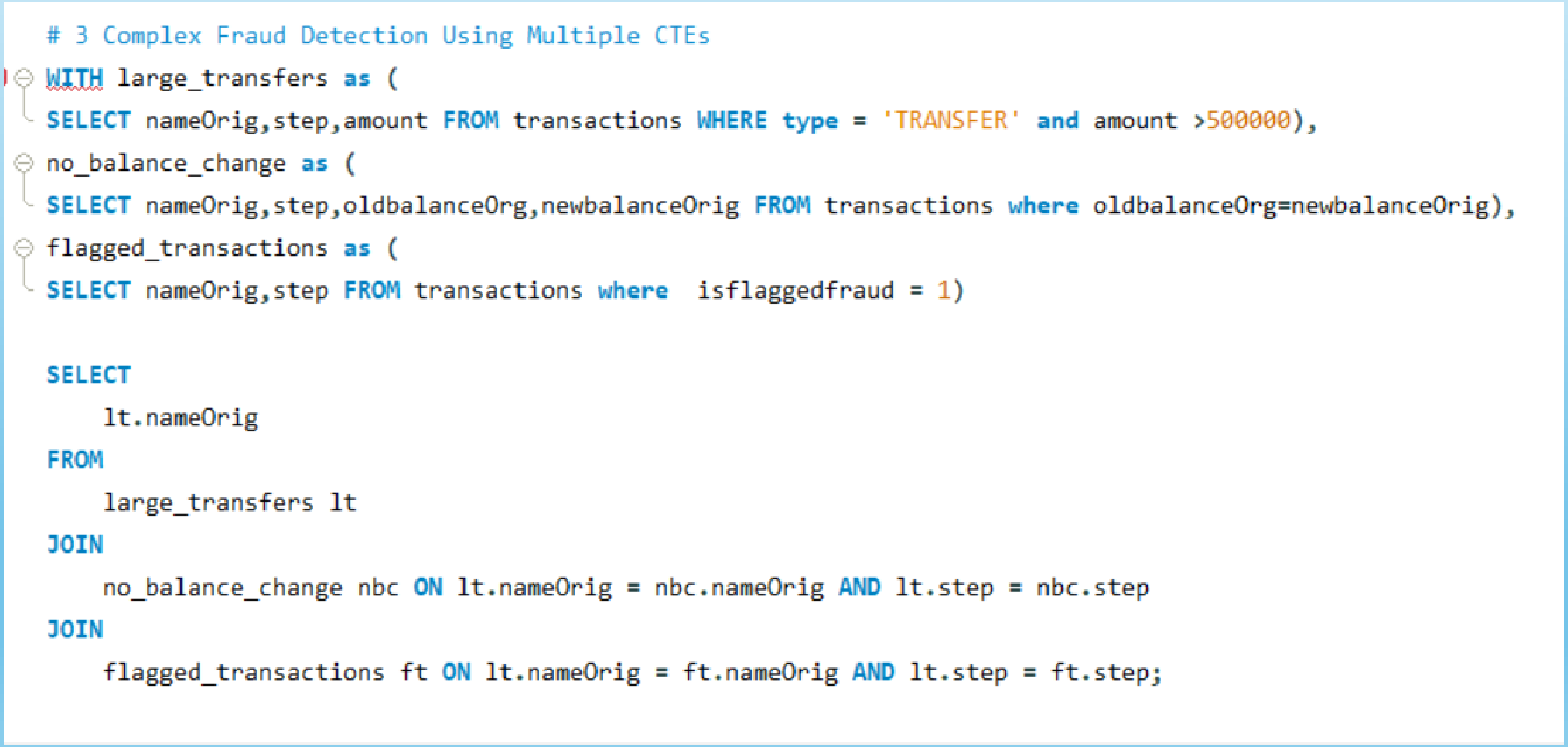
Solution :
This query uses a CTE to calculate the cumulative sum of fraudulent transactions for each account over the last five steps. It helps in understanding the temporal distribution of fraudulent activities, which is crucial for identifying patterns over time.



```
with rolling_fraud as (
select
nameorig,
step,
sum(isfraud) over
(partition by nameOrig
order by step
rows between 4 preceding and current row ) as fraud_rolling from transactions)
select * from rolling_fraud
where fraud_rolling > 0
```

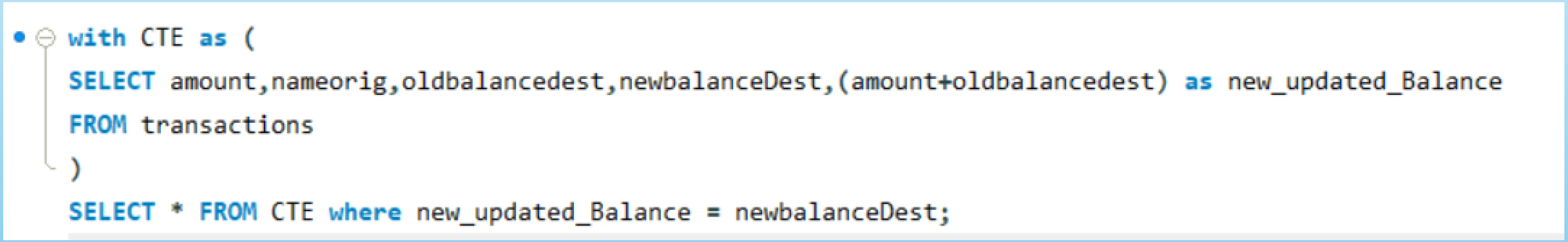
3. Complex Fraud Detection Using Multiple CTEs

Question:
Use multiple CTEs to identify accounts with suspicious activity, including large transfers, consecutive transactions without balance change, and flagged transactions.



```
# 3 Complex Fraud Detection Using Multiple CTEs
WITH large_transfers as (
SELECT nameOrig,step,amount FROM transactions WHERE type = 'TRANSFER' and amount >500000),
no_balance_change as (
SELECT nameOrig,step,oldbalanceOrg,newbalanceOrig FROM transactions where oldbalanceOrg=newbalanceOrig),
flagged_transactions as (
SELECT nameOrig,step FROM transactions where  isflaggedfraud = 1)

SELECT
    lt.nameOrig
FROM
    large_transfers lt
JOIN
    no_balance_change nbc ON lt.nameOrig = nbc.nameOrig AND lt.step = nbc.step
JOIN
    flagged_transactions ft ON lt.nameOrig = ft.nameOrig AND lt.step = ft.step;
```

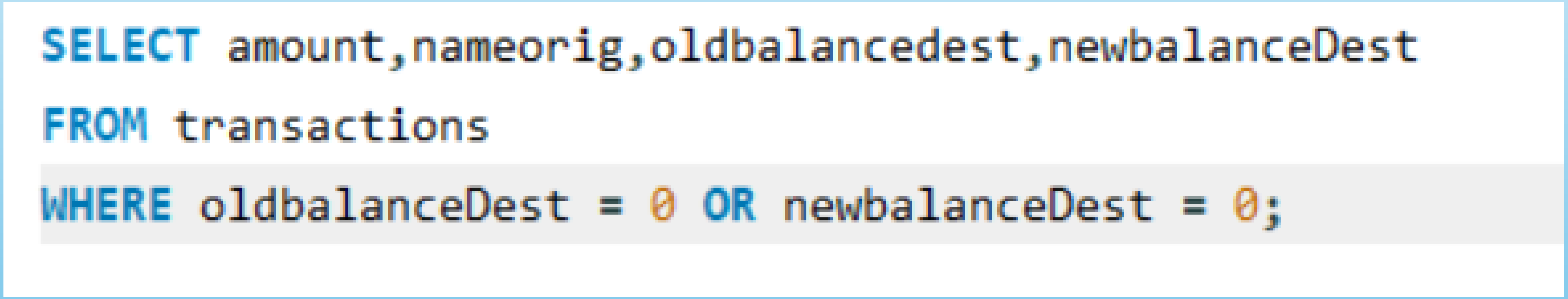
Write me a query that checks if the computed new_updated_Balance is the same as the actual newbalanceDest in the table. If they are equal, it returns those rows.



```
with CTE as (
SELECT amount,nameorig,oldbalancedest,newbalanceDest,(amount+oldbalancedest) as new_updated_Balance
FROM transactions
)
SELECT * FROM CTE where new_updated_Balance = newbalanceDest;
```

Detect Transactions with Zero Balance Before or After

● **Question**: Find transactions where the destination account had a zero balance before or after the transaction.
● **SQL Prompt**: Write a query to list transactions where oldbalanceDest or newbalanceDest is zero.



```sql
SELECT amount,nameorig,oldbalancedest,newbalanceDest
FROM transactions
WHERE oldbalanceDest = 0 OR newbalanceDest = 0;
```