0.1 Property Graph Design

The property graph data model consists of two sub property graphs: a stable and a volatile property graph. On the one hand, the stable is composed of the most static part of the data that a bank typically gathers such as information about its clients, cards, ATMs (Automated Teller Machines). On the other hand, the volatile property graph models the transaction operations, which defines the most frequent and reiterative kind of interaction between entities of the data model.
The main difference and the main reason for this separation is the semantics with which we intentionally define each of the subgraphs: the stable will be understood like a fixed static bank database, whereas the volatile will be understood as the data model to define the transactions, as continuous interactions between the entities of the model, which won't be permanently saved, but instead, only for a certain window of time under the mission of detecting anomalous bank operations. Note that we will only model the transaction interaction in the volatile subgraph, only letting them occur here. This separation will allow us to have a really simple and light property graph schema single-centered on the transactions with the minimal needed information (mostly identifiers of the entities a transaction links) and another, the stable, acting as a traditional bank database schema, from which to obtain the information details of the entities.

0.1.1 Stable property graph

Initially, we based our design in the artificial aforementioned Wisabi Bank Dataset **??**. Subsequently, although it is obvious that an actual banking data model is way more complex than the proposed one, the data model was simplified to the essential entities, relations and properties needed for the considered fraud detection patterns.

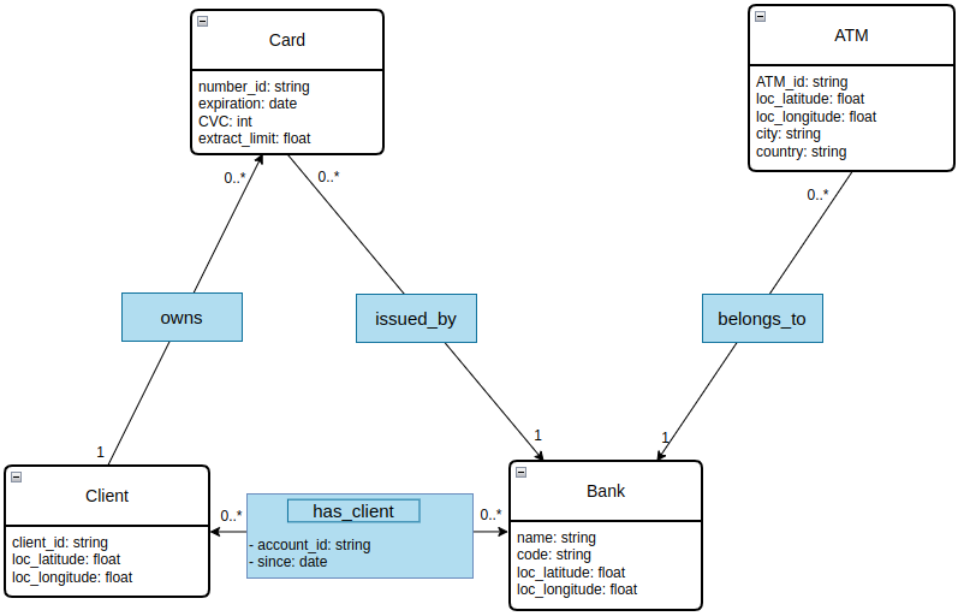

Figure 1: Initial stable property graph model

A first model proposal was the one shown in Figure 7. It contains four entities: Bank, ATM, Client and Cards, and the corresponding relations between them; in particular: a directed relationship from Client to Card: "owns" representing that a client can own multiple credit cards and that a card is owned by a unique client, then a bidirectional relation "has_client" between Client and Bank; representing bank accounts of the clients in the banks. Then the relation between Card and Bank to represent that a Card is "issued by" a Bank, and that a Bank can have multiple Cards issued. Finally, the relation "belongs_to" between the ATM and Bank entities, representing the ATMs that a Bank owns.

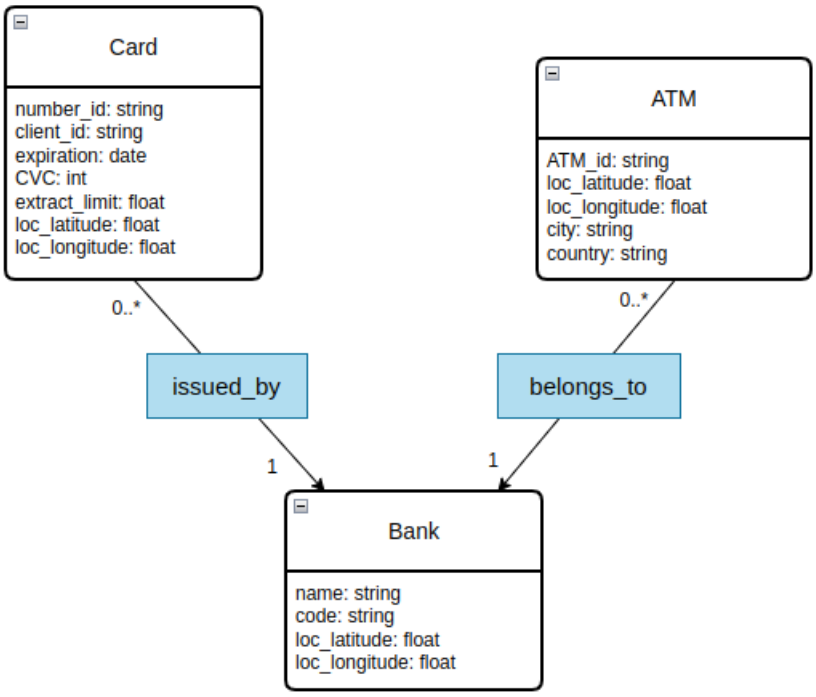

Figure 2: Definitive stable property graph model

However, the final version of the model (see Figure 2) was simplified to reduce it to the minimal needed entities. In particular, Client entity was decided to be removed and included inside the Card entity. This reduction allows to simplify the stable property graph to only three entities. For this, all the Client attributes were included in the Card entity. In the initial Figure 7 schema the Client entity was defined with three attributes: the identifier of the client and the GPS coordinates representing the usual residence of the client. This change is done while preserving the restriction of a Card belong to a unique client the same way it was previously done with the relation between Card and Client "owns" in the initial schema, which now is therefore removed.
Another derived consequence of this simplification is the removal of the other relation that the Client entity had with other entities: the "has_client" relation between Client and Bank, which was originally made with the intention of representing the bank accounts between clients and banks. Maintaining a bank account would imply having to consistently update the bank account state after each transaction of a client, and, since for the so far considered fraud detection patterns we are not considering patterns related with the accounts, the removal of the bank account relation is negligible and at the same time helpful for the

simplification of the model. However, for the sake of completeness the attribute *extract_limit* is introduced in the Card entity, representing a money amount limit a person can extract, which will be related with the amount of money a person owns. This will allow the detection of anomalies related with frequent or very high expenses.

The final entities and their selected attributes are described in what follows:

Entities

Bank

- name: Bank name.
- code: Bank identifier code.
- loc_latitude: Bank headquarters GPS-location latitude.
- loc_longitude: Bank headquarters GPS-location longitude.

Bank	
name	Bank name
code	Bank identifier code
loc_latitude	Bank headquarters GPS-location latitude
loc_longitude	Bank headquarters GPS-location longitude

Figure 3: Bank entity attributes

ATM

- ATM_id: Unique identifier of the ATM.
- loc_latitude: GPS-location latitude where the ATM is located.
- loc_longitude: GPS-location longitude where the ATM is located.
- city: City in which the ATM is located.
- country: Country in which the ATM is located.

ATM	
ATM_id	Unique identifier of the ATM
loc_latitude	GPS-location latitude where the ATM is located
loc_longitude	GPS-location longitude where the ATM is located
city	City in which the ATM is located
country	Country in which the ATM is located

Figure 4: ATM entity attributes

For the moment, this entity is understood as the classic Automated Teller Machine (ATM), however note that this entity could potentially be generalized to a Point Of Sale (POS), allowing a more general kind of transactions apart from the current Card-ATM transactions, where also online transactions could be included apart from the physical ones.

Card

- number_id: Unique identifier of the card.
- client_id: Unique identifier of the client.
- expiration: Validity expiration date of the card.
- CVC: Card Verification Code.
- extract_limit: Limit amount of money extraction associated with the card.
- loc_latitude: Client address GPS-location latitude.
- loc_longitude: Client address GPS-location longitude.

Card	
number_id	Unique identifier of the card
client_id	Unique identifier of the client
expiration	Validity expiration date of the card
CVC	Card Verification Code
extract_limit	Limit amount of money extraction associated with the card
loc_latitude	Client address GPS-location latitude
loc_longitude	Client address GPS-location longitude

Figure 5: Card entity attributes

Note that for both the ATM and the Card entities we have the GPS coordinates information. In the first case referring to the geolocation of each specific ATM and in the last case referring to each specific client address geolocation. This will be useful to be able to detect transaction frauds related to geolocation distances.

Note that the client is completely anonymized in the system (no name, surname, age, or any other confidential details) by using only a client_id. For the present purpose it is enough to uniquely identify each client.

Relations There are only two relations in the final stable subgraph, that is, the ones that are left after the deletion of the Client entity from the first version of the stable subgraph; the relation "issued_by" between the Card and the Bank entities and the "belongs_to" between ATM and Bank entities. For the moment they do not have any attribute, however they could potentially be added in the case this was needed.

0.1.2 Volatile property graph

It contains the minimal needed information to be able to recognize the anomaly fraud patterns we want to identify. This subgraph describes the most volatile part of our model, meaning the transactions between the client's cards and the ATMs. The idea is to have a data model to define the transactions, as continuous temporal interactions between the Card and ATM entities, restricting these kind of relations to the volatile subgraph. The idea is to have a really simple and light property graph schema single-centered on the transactions. For that the Card and ATM entities will be simplified to the last bullet, containing only the identifier of the both entities that each transaction relation matches. These

identifiers will be enough to be able to recover, if needed, the whole information about the specific Card or ATM entity in the stable subgraph.

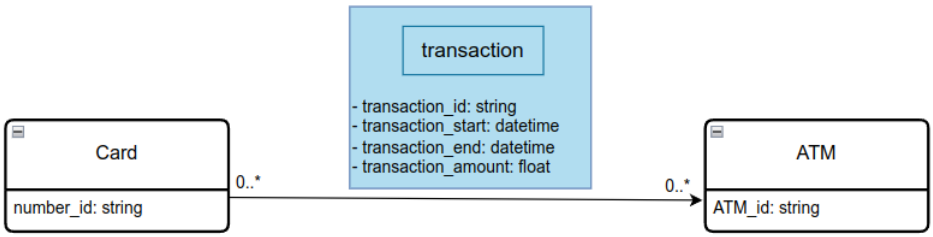

Figure 6: Volatile property graph model

Following this idea of volatile subgraph, the Card and ATM entities and transaction relations of it will not be permanently saved, but instead, only for a certain window of time under the mission of detecting anomalous bank operations.

The transaction relation will include some attributes that will be essential for the detection of the fraud patterns, in particular:

- transaction_id: Unique identifier for each transaction in the database.
- transaction_start: Datetime when the transaction started. Format: DD/MM/YYYY HH:MM (ex. 1/1/2022 4:50).
- transaction_end: Datetime when the transaction was completed. Format: DD/MM/YYYY HH:MM (ex. 1/1/2022 4:54).
- transaction_amount: Amount of money involved in the transaction.

Transaction	
transaction_id	Unique identifier for each transaction in the database
transaction_start	Datetime when the transaction started - format: DD/MM/YYYY HH:MM (ex. 1/1/2022 4:50)
transaction_end	Datetime when the transaction was completed - format: DD/MM/YYYY HH:MM (ex. 1/1/2022 4:54)
transaction_amount	Amount of money involved in the transaction

Figure 7: Transaction relation attributes