MoMo SMS Data Processing System – Database Design Document

Entity Relationship Diagram

The Entity Relationship Diagram for this system includes five main entities: User, Transaction, Transaction_category, Transaction_Category_fee, and System_logs.

The User entity captures information about customers, such as their user ID, user type, first name, last name, and phone number. Phone numbers should remain unique to avoid duplication.

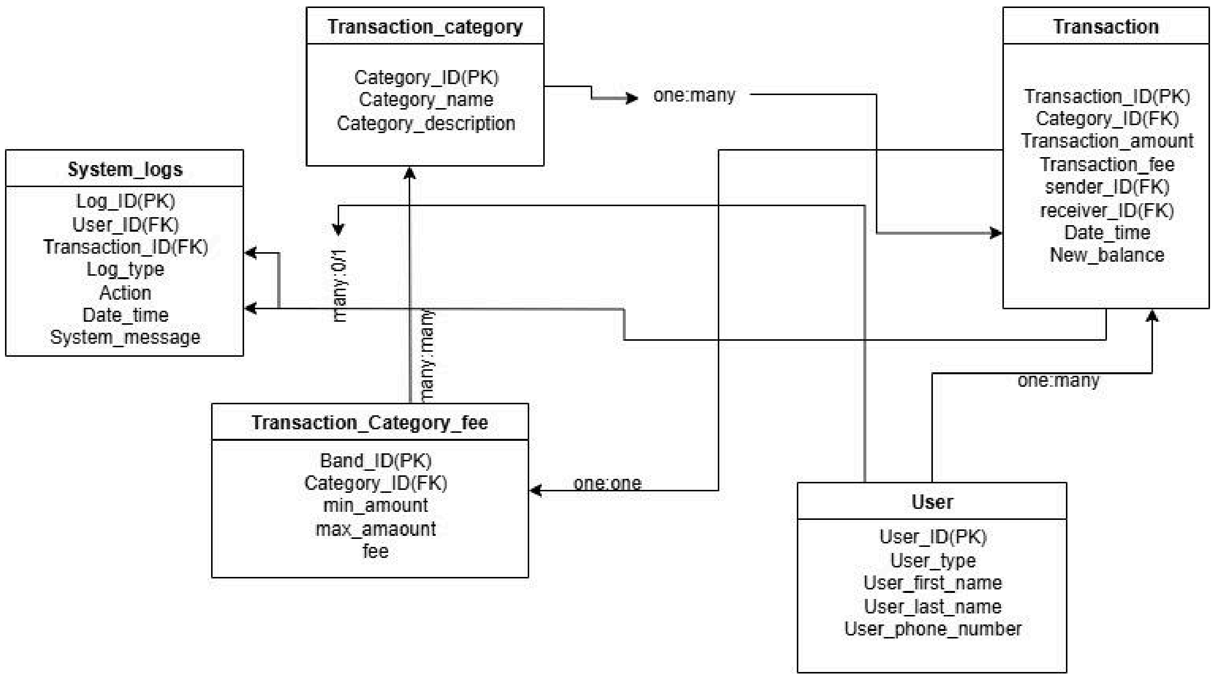
The Transaction entity is central, storing the transaction ID, amount, currency, timestamp, balance, and transaction fee. Each transaction is linked to a category through category_id as a foreign key. Sender_id and receiver_id are both foreign keys referencing User and User_type , which makes it possible to record the two parties involved in the transaction.

The Transaction_category entity classifies transactions into categories such as payment, withdrawal, or transfer. It contains category_id as the primary key and includes category_name and category_description.

The Transaction_Category_fee entity supports tiered fees. It contains band_id as the primary key, links to Transaction_category through category_id as a foreign key, and includes minimum amount, maximum amount, and the corresponding fee. This ensures that transactions can be charged according to different ranges.

The System_logs entity records system events, including errors and actions, and links them to users through user_id as a foreign key. Each log contains log_id as the primary key, as well as log type, action, timestamp, and system message.

The ERD clearly shows primary keys, foreign keys, and relationship cardinalities such as one-to-many between categories and transactions, one-to-many between users and logs, and many-to-many between categories and fee bands.



Design Rationale and Justification

The design reflects how mobile money transactions occur in reality while ensuring data integrity and scalability of sytems. The Transaction entity is at the center of the model because it connects to users, categories, and system logs, which ensures complete traceability.

Users are stored in their own entity to avoid duplication of personal details across multiple transactions. The use of sender_id and receiver_id allows the system to explicitly capture both participants in a single transaction. This creates a one-to-many relationship between User and Transaction, since one user can send or receive many transactions.

Transactions are linked to categories in a one-to-many relationship. Each transaction belongs to a single category, such as a withdrawal, deposit or transfer, while a category can be applied to many transactions. This avoids redundancy and ensures consistent classification.

The Transaction_Category_fee entity introduces a many-to-many relationship between transaction categories and fee bands. A single category can have multiple fee bands depending on transaction amounts, and different categories may share similar fee bands.

System_logs provide accountability by recording system-level events. Logs are linked to users in a one-to-many relationship or a many-to-none, as one user can generate multiple system logs over time and many users can fail to generate any system logs at a time. Logs may also reference transactions just as users, which strengthens traceability in the system.

By capturing one-to-many,many-to-many and many-to-none relationships, the database is well suited to reflect real-world complexities like multiple user roles, variable fee structures, and system monitoring.

Data Dictionary

User entity: Contains user_id as the primary key. Other fields include user_type, first name, last name, and phone number. Phone numbers must remain unique.

Transaction_category entity: Contains category_id as the primary key, with category_name and category_description as attributes.

Transaction_Category_fee entity: Contains band_id as the primary key. Other fields include minimum amount, maximum amount, fee, and category_id as a foreign key referencing Transaction_category.

Transaction entity: Contains transaction_id as the primary key. Other attributes include transaction_amount, transaction_fee, currency, date_time, and new_balance. Each transaction references a category through category_id as a foreign key, and links to sender_id and receiver_id as foreign keys referencing User.

System_logs entity: Contains log_id as the primary key. Other attributes are log_type, action, date_time, and system_message. User_id is a foreign key referencing User, and transaction_id may also be included as a foreign key referencing Transaction.

Sample Queries

Insert a new transaction:

```
INSERT INTO Transaction (transaction_id, category_id, transaction_amount,
transaction_fee, currency, date_time, new_balance, sender_id, receiver_id)

VALUES ('TXN001', 1, 500.00, 10.00, 'KES', '2025-09-14 10:00:00', 1200.00, 1, 2);
```

Retrieve all transactions by a specific user:

```
SELECT t.transaction_id, t.transaction_amount, t.currency, u.first_name, u.last_name

FROM Transaction t

JOIN User u ON t.sender_id = u.user_id OR t.receiver_id = u.user_id

WHERE u.phone_number = '0712345678';
```

Summarize transaction amounts per category:

```
SELECT c.category_name, SUM(t.transaction_amount) AS total_amount

FROM Transaction t
```

```
JOIN Transaction_category c ON t.category_id = c.category_id

GROUP BY c.category_name;
```

View error logs for users:

```
SELECT l.log_id, l.system_message, u.first_name, u.last_name

FROM System_logs l

JOIN User u ON l.user_id = u.user_id

WHERE l.log_type = 'ERROR';
```