MoMo SMS Data Processing System – Database Design Document

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

- The User entity captures information about customers, such as their names, phone numbers, identification numbers, and locations.
- The Transaction entity is central, storing the transaction ID, amount, currency, timestamp, balance, and raw SMS message.
- The Transaction_category entity classifies transactions into categories such as payment, withdrawal, or transfer.
- The Transaction_participants entity resolves the many-to-many relationship between Users and Transactions by assigning each participant a role, such as sender, receiver, or merchant.
- The System_logs entity records system events, including errors and actions, and can optionally link to users or transactions for traceability.

The ERD clearly shows primary keys, foreign keys, and relationship cardinalities such as one-to-many and many-to-many.

Design Rationale and Justification

The design reflects how mobile money transactions occur in reality while ensuring data integrity and scalability. Transactions are linked to categories to avoid redundancy, and users are kept in a separate entity to maintain unique customer information. The junction table for participants allows the system to capture multiple roles in a single transaction, which is essential for accuracy and flexibility. System logs add accountability by tracking both general errors and transaction-specific issues.

This structure minimizes redundancy, normalizes data, and ensures flexibility. The database is scalable and can support more categories, users, or log types without requiring major redesign.

Data Dictionary

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

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

Transaction entity: Contains transaction_id as the primary key. Other attributes include transaction_amount, currency, date_time, new_balance, and raw_message. Each transaction references a category through category_id as a foreign key.

Transaction_participants entity: Contains transaction_id and user_id as a composite primary key. Both are foreign keys referencing Transaction and User respectively. It also includes the role of the user in the transaction.

System_logs entity: Contains log_id as the primary key. Other attributes are log_type, action, date_time, and system_message. Transaction_id and user_id can be included as optional foreign keys to link logs to specific records.

Sample Queries

Insert a new transaction:

```
INSERT INTO Transaction (transaction_id, category_id, transaction_amount, currency, date_time, new_balance, raw_message)
VALUES ('TXN001', 1, 500.00, 'KES', '2025-09-14 10:00:00', 1200.00, 'Paid 500 to Merchant XYZ');
```

Retrieve all transactions by a specific user:

```
SELECT t.transaction_id, t.transaction_amount, t.currency, tp.user_role
FROM Transaction t
JOIN Transaction_participants tp ON t.transaction_id = tp.transaction_id
JOIN User u ON tp.user_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 transactions:

```
SELECT l.log_id, l.system_message, t.transaction_id
FROM System_logs l
LEFT JOIN Transaction t ON l.transaction_id = t.transaction_id
WHERE l.log_type = 'ERROR';
```