

Team Packet Sniffers

MoMo SMS ETL & Dashboard

Database Design Document(DDD)

Prepared By: Sedem Kofi Amuzu

Joshua Agonzibwa

Larry Sentore

Detailed Database Design

This section describes the actual design of our MoMo analysis database.

Entity Relationship Diagram - Crow's Foot Notation

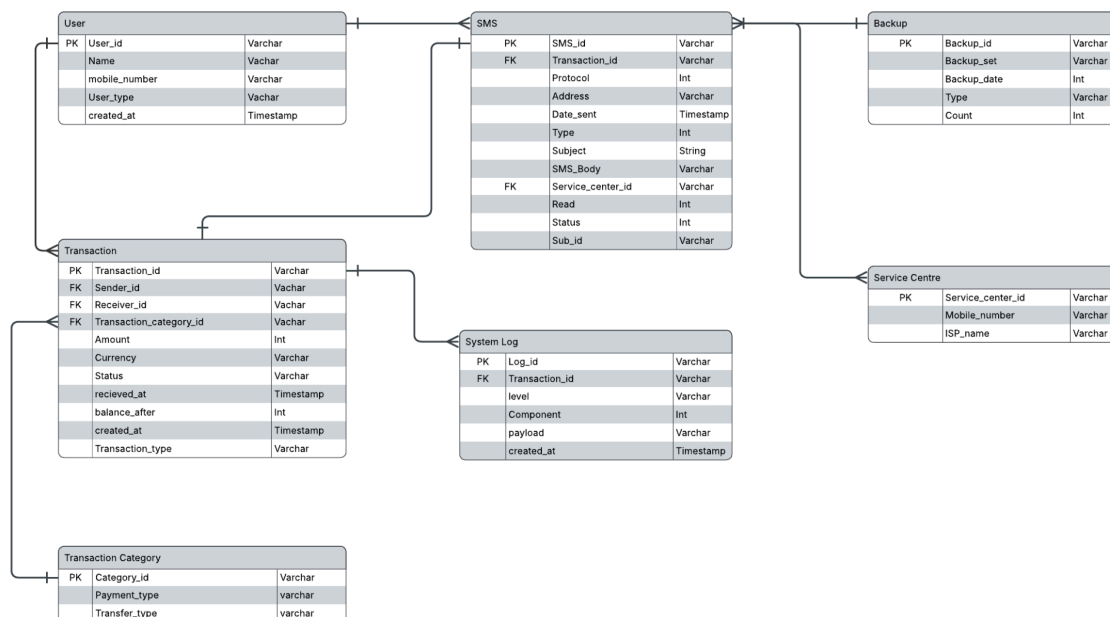


Figure 1: Database Design - Entity Relationship Diagram

Description & Purpose

We designed a database for our web app that will store mobile money information and separate it into categories that allow for easy analysis and display on our web app. The entities we identified are the user, the transactions, the transaction categories, the sms, the system log, the sms backup, and the service centre. The attributes for the users include a type to identify the different types of users (regular, agents, etc) to differentiate between transactions better. The transactions entity uses the private key of the user entity in order to identify the sender and the receiver, and this was designed this way to avoid confusion and to properly organise the data. The transaction category will have more details on what type of transaction occurred, which will help

with identifying the fees. We included the system log to track transaction processes and find errors or delays quickly and efficiently. After analysing the XML file, we noticed the first line was of some sort of backup, so we added it as an entity that interacts with the sms and backs it up. Finally, the service centre entity provides its private key to the sms entity for easier organisation.

Amounts transferred are in integer format, dates and times are formatted in DATETIME, and the other attributes are formatted in varchar. This allows for flexibility in naming and such. Overall, our design allows for easy viewing and accessing of data.

Data Dictionary

Database Dictionary For Element: User

Name	Data Type	Description
User_id(Primary Key)	Varchar	Unique Identifier to identify Users
First_name	Varchar	First name of the user
Last_name	Varchar	Last name of the user
Mobile_number	Varchar	Mobile number of the user
User_type	Varchar	Type of MoMo User(Customer, Agent etc)
Created_at	Timestamp	Time the user was created/added to the database

Database Dictionary For Element: Transaction

Name	Data Type	Description
Transaction_id(Primary Key)	Varchar	ID to identify transactions
Sender_id(Foreign Key)	Varchar	ID to identify the Sender
Receiver_id(Foreign Key)	Varchar	ID to identify the Receiver
Transaction_category_id(Foreign Key)	Varchar	ID to identify Transaction category
Amount	Int	The amount of money that was sent out
Currency	Varchar	Currency of the country
Status	Varchar	Whether or not the transaction was a success or not
Received_at	Timestamp	The time the transaction went through, and/money was received
Balance_after	Int	Amount of money left in the sender's account after the transaction
Created_at	Timestamp	The time at the transaction was created
Transaction_type	Varchar	Shows whether a user has sent or received a transaction

Database Dictionary For Element: Transaction Category

Name	Data Type	Description
Category_id(Primary Key)	Varchar	Unique identifier to identify transaction category
Payment_type	Varchar	First name of the user
Transfer_type	Varchar	Last name of the user

Database Dictionary For Element: SMS

Name	Data Type	Description
SMS_id(Primary Key)	Varchar	Unique identifier for each SMS message (auto-generated)
Transaction_id(Foreign Key)	Varchar	Reference to the related transaction
Protocol	Int	Protocol identifier for the SMS
Address	Varchar	The sender or recipient phone number
Date_sent	Timestamp	Date and Time when the SMS was sent
Type	Int	Type of SMS
Subject	String	Subject of the SMS body
SMS_Body	Varchar	This is where the body of the SMS is stored
Service_center_id(Foreign Key)	Varchar	The ID of the service centre from the Service Centre Table
Read	Int	Read status(1 if read, 0 if unread)
Status	Int	Delivery status of the SMS
Sub_id	Varchar	Subscription or SIM slot identifier

Database Dictionary For Element: System Log

Name	Data Type	Description
Log_id(Primary Key)	Varchar	Unique Identifier for the log entry
Transaction_id(Foreign Key)	Varchar	Foreign Key to the relation transaction
Level	Varchar	Severity level of the log entry (classification for filtering/alerts)
Component	Int	Numeric identifier for the subsystem/component that produced the log (for grouping and routing)
Payload	Varchar	Human-readable message or structured payload describing the event, error or metadata
Created_at	Timestamp	Timestamp when the log entry was created

Database Dictionary For Element: Backup

Name	Data Type	Description
Backup_id(Primary Key)	Varchar	Unique identifier for the backup record
Backup_set	Varchar	Identifier for the backup set
Backup_date	Int	Timestamp when the backup was taken
Type	Varchar	Backup type/category (Full, Incremental, Differential)
Count	Int	Number of SMS records

		included in the backup
--	--	------------------------

Database Dictionary For Element: Service Center

Name	Data Type	Description
Service_center_id(Primary Key)	Varchar	Unique identifier for the service centre
Mobile_number	Varchar	Mobile Number of the service Center
ISP_name	Varchar	The name of the Internet Service Provider(ISP)

Sample Queries

CRUD Operations

Below are examples of basic CRUD (Create, Read, Update, Delete) operations performed on the database. Each operation is illustrated with a screenshot.

Create

The following screenshot shows a successful insert operation into the User table



Figure 2: MYSQL query to insert data into the USER table

	Q	User_id varchar(50)	First_name varchar(255)	Last_name varchar(255)	Mobile_number varchar(50)	User_type varchar(50)	Created_at timestamp
	>	US0001	Alice	Smith	0791000001	customer	2025-09-21 03:44:57
	>	US0002	Bob	Jones	0791000002	customer	2025-09-21 03:44:57
	>	US0003	Carol	White	0791000003	admin	2025-09-21 03:44:57
	>	US0004	David	Black	0791000004	customer	2025-09-21 03:44:57
	>	US0005	Eve	Green	0791000005	customer	2025-09-21 03:44:57
	>	US0006	Frank	Blue	0791000006	Customer	2025-09-21 03:56:06

Figure 3: Database showing the Insertion

Read

The following screenshot shows a successful select operation from the User table.

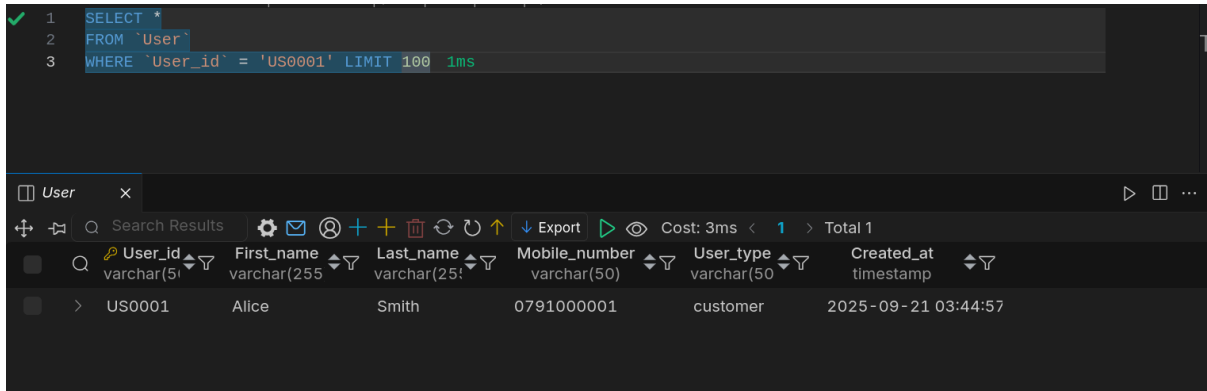


Figure 4: Successful select operation from the USER table

Update

The following screenshot shows a successful update operation on the Transaction table.

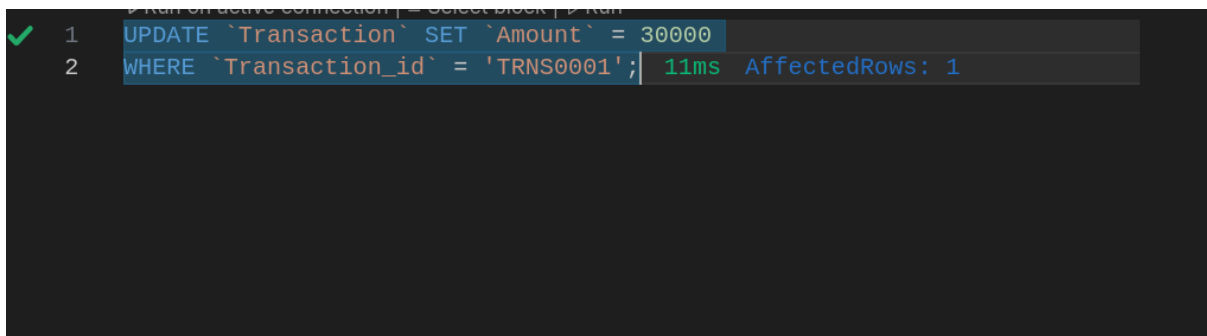


Figure 5: Successful Update operation from the TRANSACTION table

Delete

The following screenshot shows a successful delete operation from the SMS table.

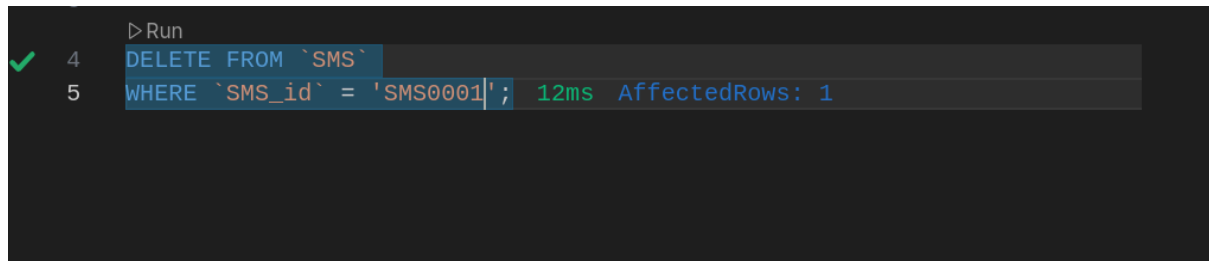
A screenshot of a database client interface. It shows a SQL query being executed: `DELETE FROM `SMS` WHERE `SMS_id` = 'SMS0001';`. The query is highlighted in blue. To the right of the query, it shows the execution time as `12ms` and the number of affected rows as `AffectedRows: 1`. A green checkmark is visible on the left side of the query, indicating a successful execution.

Figure 5: Successful Delete operation from the SMS table

Data Accuracy

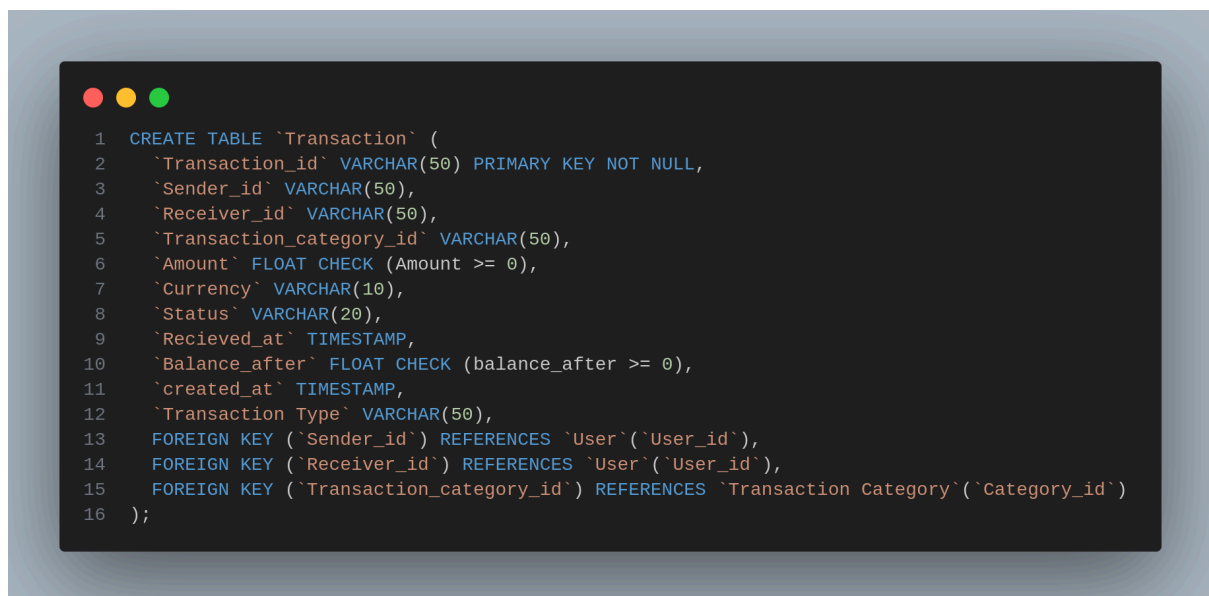
A screenshot of a database client interface showing a SQL query to create a table named 'Transaction'. The query is as follows: `CREATE TABLE `Transaction` (
 `Transaction_id` VARCHAR(50) PRIMARY KEY NOT NULL,
 `Sender_id` VARCHAR(50),
 `Receiver_id` VARCHAR(50),
 `Transaction_category_id` VARCHAR(50),
 `Amount` FLOAT CHECK (Amount >= 0),
 `Currency` VARCHAR(10),
 `Status` VARCHAR(20),
 `Recieved_at` TIMESTAMP,
 `Balance_after` FLOAT CHECK (balance_after >= 0),
 `created_at` TIMESTAMP,
 `Transaction Type` VARCHAR(50),
 FOREIGN KEY (`Sender_id`) REFERENCES `User`(`User_id`),
 FOREIGN KEY (`Receiver_id`) REFERENCES `User`(`User_id`),
 FOREIGN KEY (`Transaction_category_id`) REFERENCES `Transaction Category`(`Category_id`)
);`

Figure 6: Transaction Table query

The column in the table for the primary key has the “NOT NULL” command, which ensures that the primary key field is never empty, which is a key requirement when setting up primary keys

Further Data validation was added to ensure the accuracy of the data in the table. This can be seen in the “Amount” column of the table, where there is another validation

check to ensure that the amount of money involved in the transaction can never be negative, i.e ≥ 0

APPENDIX 1 - XML Schema

This is a part of the sample XML schema that our Database Design is centred around.

```
<?xml version='1.0' encoding='utf-8'?>
    <smses      count="1693"      backup_set="3af074dd-adfc-49ab-bb55-c3fd98e6146f"
    backup_date="1737023646162" type="full">
        <sms  protocol="0"  address="M-Money"  date="1715351458724"  type="1"  subject="null"
        body="You have received 2000 RWF from Jane Smith (*****013) on your mobile money account
        at 2024-05-10 16:30:51. Message from sender: . Your new balance:2000 RWF. Financial Transaction
        Id: 76662021700."  toa="null"  sc_toa="null"  service_center="+250788110381"  read="1"
        status="-1"  locked="0"  date_sent="1715351451000"  sub_id="6"  readable_date="10 May 2024
        4:30:58 PM"  contact_name="(Unknown)" />
```