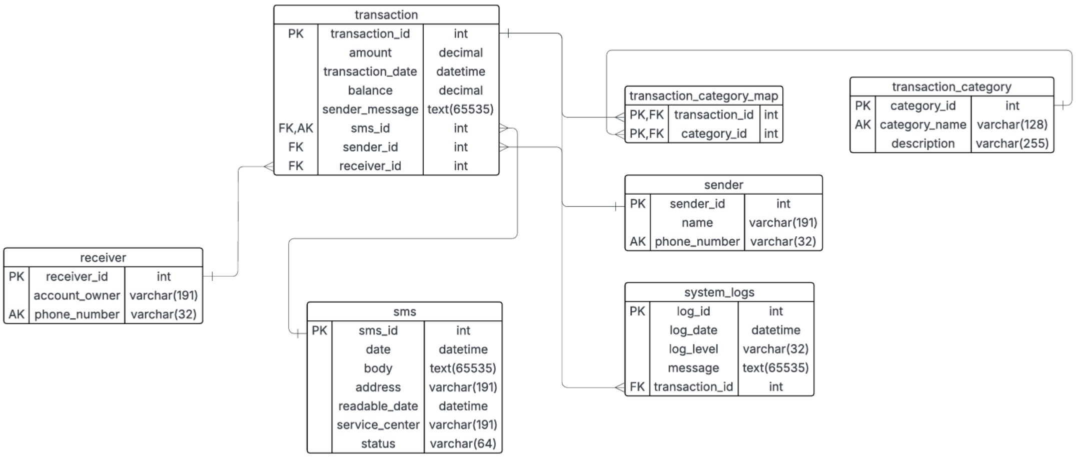
We developed the ALU_SMS database to track financial transactions from MoMo SMS notifications. Our database stores and analyzes transactions while maintaining data integrity. We designed our database by implementing an Entity Relationship Diagram to illustrate the relationship between entities.



transaction

PK	transaction_id	int
	amount	decimal
	transaction_date	datetime
	balance	decimal
	sender_message	text(65535)
FK,AK	sms_id	int
FK	sender_id	int
FK	receiver_id	int

transaction_category_map

PK,FK	transaction_id	int
PK,FK	category_id	int

transaction_category

PK	category_id	int
AK	category_name	varchar(128)
	description	varchar(255)

sender

PK	sender_id	int
	name	varchar(191)
AK	phone_number	varchar(32)

receiver

PK	receiver_id	int
	account_owner	varchar(191)
AK	phone_number	varchar(32)

sms

PK	sms_id	int
	date	datetime
	body	text(65535)
	address	varchar(191)
	readable_date	datetime
	service_center	varchar(191)
	status	varchar(64)

system_logs

PK	log_id	int
	log_date	datetime
	log_level	varchar(32)
	message	text(65535)
FK	transaction_id	int

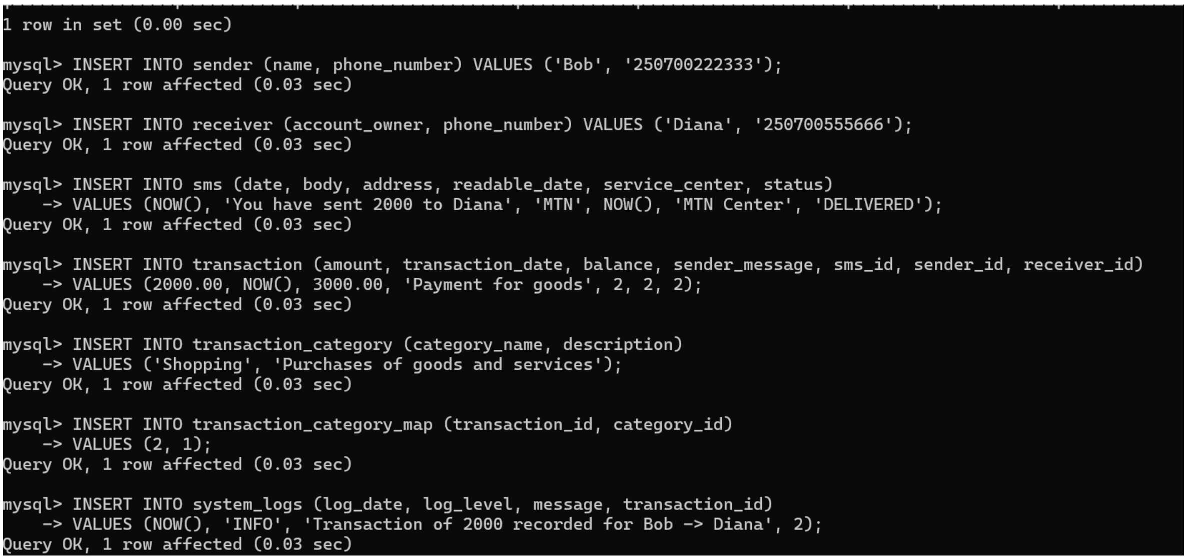
The sender and receiver tables capture the sender and receiver phone numbers, respectively. The SMS table contains raw message data and is directly linked to the Transaction table, so every transaction can be traced back to its original source. To support flexible categorization, we added the Transaction_Category table and the mapping table because there is a many-to-many relationship between transactions and categories. Using this structure, we applied foreign keys and unique constraints to ensure the uniformity of all relationships.

Data Dictionary

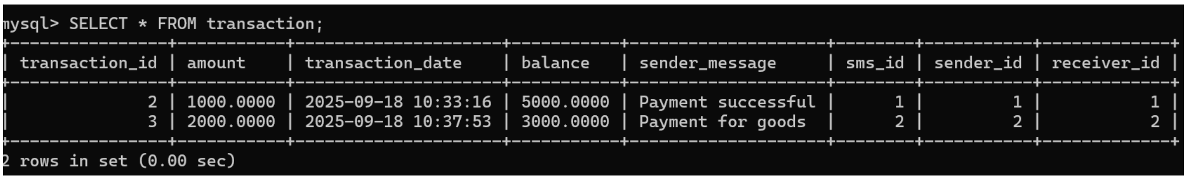
Table	Column	Description
Sender	Sender_id(PK), name, phone_number(AK)	Stores sender's details
Receiver	receiver_id(PK), account_owner, phone_number	Stores receiver's details
SMS	sms_id(PK), date, body, address, readable_date, service_center, status	Stores SMS messages
Transaction	transaction_id(PK), amount, transaction_date, balance, sender_message, sms_id(FK), sender_id(FK), receiver_id(FK)	Stores financial transactions
Transaction_Category	category_id(PK), category_name, description	Defines transaction categories like Utilities

Transaction_Category_Map	transaction_id(FK), category_id(FK)	Describes the many-to-many relationship between transactions and categories.
System_logs	log_id(PK), log_date, log_level, message, transaction_id(FK)	Handles system activity.

Sample Queries



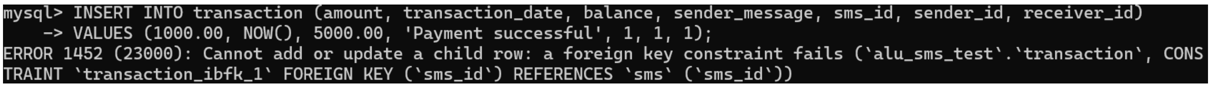
```
1 row in set (0.00 sec)

mysql> INSERT INTO sender (name, phone_number) VALUES ('Bob', '250700222333');
Query OK, 1 row affected (0.03 sec)

mysql> INSERT INTO receiver (account_owner, phone_number) VALUES ('Diana', '250700555666');
Query OK, 1 row affected (0.03 sec)

mysql> INSERT INTO sms (date, body, address, readable_date, service_center, status)
    -> VALUES (NOW(), 'You have sent 2000 to Diana', 'MTN', NOW(), 'MTN Center', 'DELIVERED');
Query OK, 1 row affected (0.03 sec)

mysql> INSERT INTO transaction (amount, transaction_date, balance, sender_message, sms_id, sender_id, receiver_id)
    -> VALUES (2000.00, NOW(), 3000.00, 'Payment for goods', 2, 2, 2);
Query OK, 1 row affected (0.03 sec)

mysql> INSERT INTO transaction_category (category_name, description)
    -> VALUES ('Shopping', 'Purchases of goods and services');
Query OK, 1 row affected (0.03 sec)

mysql> INSERT INTO transaction_category_map (transaction_id, category_id)
    -> VALUES (2, 1);
Query OK, 1 row affected (0.03 sec)

mysql> INSERT INTO system_logs (log_date, log_level, message, transaction_id)
    -> VALUES (NOW(), 'INFO', 'Transaction of 2000 recorded for Bob -> Diana', 2);
Query OK, 1 row affected (0.03 sec)
```



```
mysql> SELECT * FROM transaction;
+----------------+-----------+---------------------+-----------+--------------------+--------+-----------+-------------+
| transaction_id | amount    | transaction_date    | balance   | sender_message     | sms_id | sender_id | receiver_id |
+----------------+-----------+---------------------+-----------+--------------------+--------+-----------+-------------+
|              2 | 1000.0000 | 2025-09-18 10:33:16 | 5000.0000 | Payment successful |      1 |         1 |           1 |
|              3 | 2000.0000 | 2025-09-18 10:37:53 | 3000.0000 | Payment for goods  |      2 |         2 |           2 |
+----------------+-----------+---------------------+-----------+--------------------+--------+-----------+-------------+
2 rows in set (0.00 sec)
```

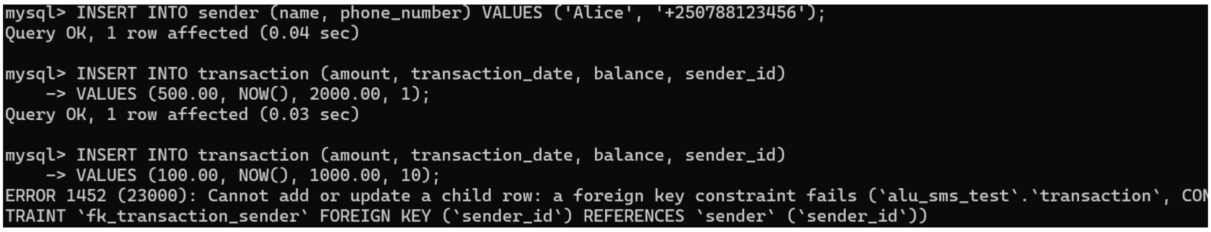
Unique Constraints

1. The foreign key(sms_id) constraint ensures that each transaction is linked to an existing SMS.



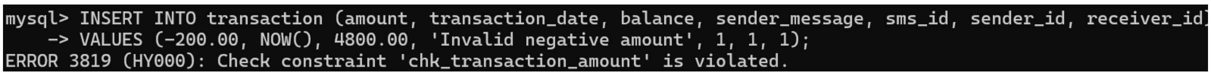
```
mysql> INSERT INTO transaction (amount, transaction_date, balance, sender_message, sms_id, sender_id, receiver_id)
    -> VALUES (1000.00, NOW(), 5000.00, 'Payment successful', 1, 1, 1);
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`alu_sms_test`.`transaction`, CONS
TRAINT `transaction_ibfk_1` FOREIGN KEY (`sms_id`) REFERENCES `sms` (`sms_id`))
```

2. This constraint ensures that the sender_id in the transaction table matches a valid sender_id in the sender table. It also ensures referential integrity.



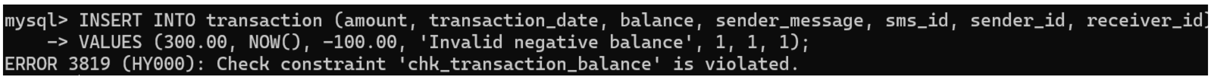
```
mysql> INSERT INTO sender (name, phone_number) VALUES ('Alice', '+250788123456');
Query OK, 1 row affected (0.04 sec)

mysql> INSERT INTO transaction (amount, transaction_date, balance, sender_id)
    -> VALUES (500.00, NOW(), 2000.00, 1);
Query OK, 1 row affected (0.03 sec)

mysql> INSERT INTO transaction (amount, transaction_date, balance, sender_id)
    -> VALUES (100.00, NOW(), 1000.00, 10);
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`alu_sms_test`.`transaction`, CON
TRAINT `fk_transaction_sender` FOREIGN KEY (`sender_id`) REFERENCES `sender` (`sender_id`))
```

3. The transaction amount must be positive.



```
mysql> INSERT INTO transaction (amount, transaction_date, balance, sender_message, sms_id, sender_id, receiver_id)
    -> VALUES (-200.00, NOW(), 4800.00, 'Invalid negative amount', 1, 1, 1);
ERROR 3819 (HY000): Check constraint 'chk_transaction_amount' is violated.
```

4. The Balance must be greater than or equal to zero.



```
mysql> INSERT INTO transaction (amount, transaction_date, balance, sender_message, sms_id, sender_id, receiver_id)
    -> VALUES (300.00, NOW(), -100.00, 'Invalid negative balance', 1, 1, 1);
ERROR 3819 (HY000): Check constraint 'chk_transaction_balance' is violated.
```