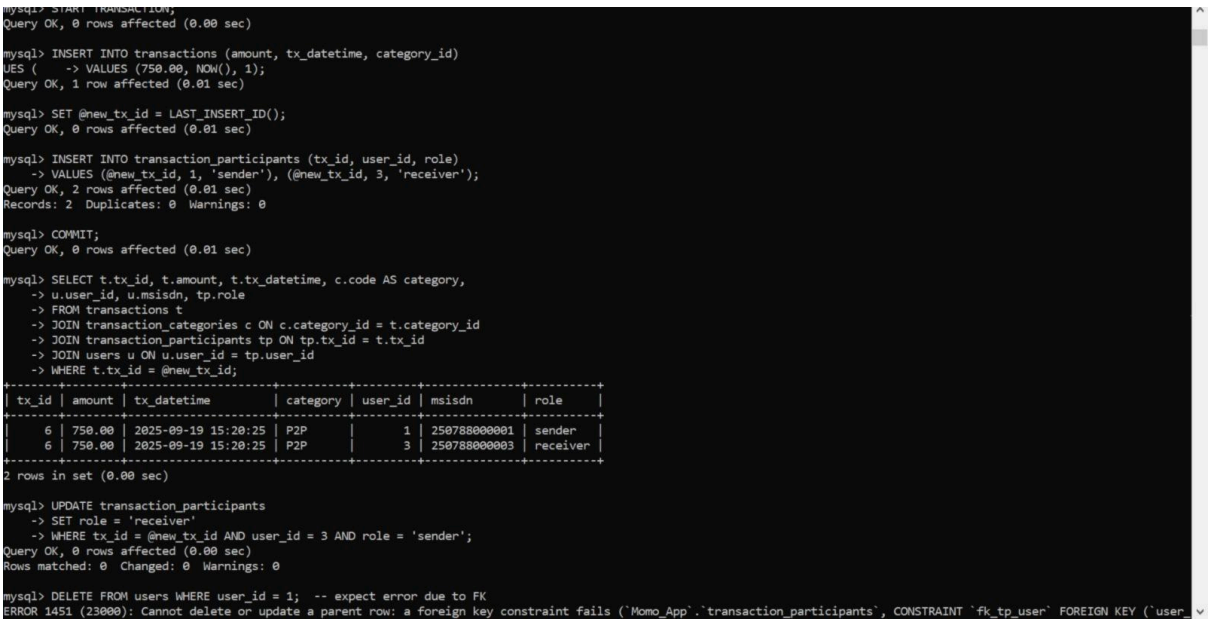
Group 2 ERD Document and Screenshots

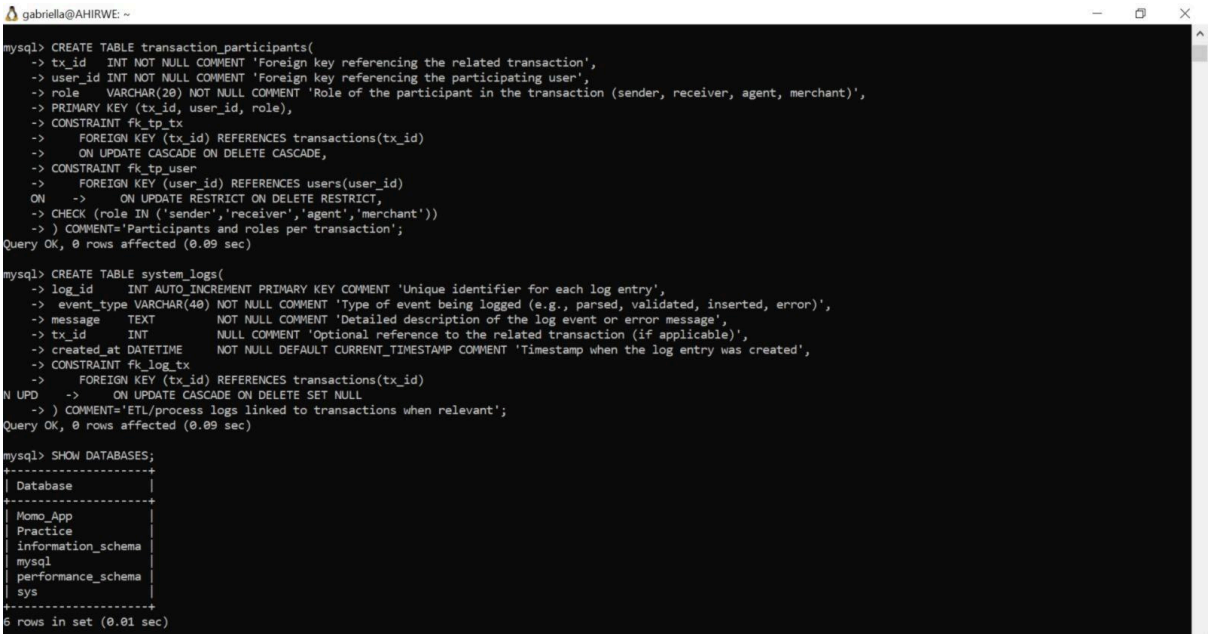


```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO transactions (amount, tx_datetime, category_id)
UES (    -> VALUES (750.00, NOW(), 1);
Query OK, 1 row affected (0.01 sec)

mysql> SET @new_tx_id = LAST_INSERT_ID();
Query OK, 0 rows affected (0.01 sec)

mysql> INSERT INTO transaction_participants (tx_id, user_id, role)
    -> VALUES (@new_tx_id, 1, 'sender'), (@new_tx_id, 3, 'receiver');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> COMMIT;
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT t.tx_id, t.amount, t.tx_datetime, c.code AS category,
    -> u.user_id, u.msisdn, tp.role
    -> FROM transactions t
    -> JOIN transaction_categories c ON c.category_id = t.category_id
    -> JOIN transaction_participants tp ON tp.tx_id = t.tx_id
    -> JOIN users u ON u.user_id = tp.user_id
    -> WHERE t.tx_id = @new_tx_id;
+-------+--------+---------------------+----------+---------+-------------+----------+
| tx_id | amount | tx_datetime         | category | user_id | msisdn      | role     |
+-------+--------+---------------------+----------+---------+-------------+----------+
|     6 | 750.00 | 2025-09-19 15:20:25 | P2P      |       1 | 250788000001 | sender   |
|     6 | 750.00 | 2025-09-19 15:20:25 | P2P      |       3 | 250788000003 | receiver |
+-------+--------+---------------------+----------+---------+-------------+----------+
2 rows in set (0.00 sec)

mysql> UPDATE transaction_participants
    -> SET role = 'receiver'
    -> WHERE tx_id = @new_tx_id AND user_id = 3 AND role = 'sender';
Query OK, 0 rows affected (0.00 sec)
Rows matched: 0  Changed: 0  Warnings: 0

mysql> DELETE FROM users WHERE user_id = 1;  -- expect error due to FK
ERROR 1451 (23000): Cannot delete or update a parent row: a foreign key constraint fails (`Momo_App`.`transaction_participants`, CONSTRAINT `fk_tp_user` FOREIGN KEY (`user_
```

Our ERD is built to support secure, scalable, and efficient handling of mobile money transactions. The Transactions table forms the core, storing transaction IDs, amounts, timestamps, and categories. To avoid redundancy, user details are kept in the Users table, linked via the Transaction_Participants table, which clearly distinguishes roles such as sender or receiver. This normalization prevents duplication, supports many-to-many relationships, and maintains flexibility as the system scales.

The Transaction_Categories table ensures consistent classification of payments, transfers, or withdrawals, enabling both operational clarity and analytical reporting. System_Logs are maintained independently to track errors, events, and system actions, providing an audit trail without burdening the transaction records.

Primary keys, foreign keys, and indexing enforce data integrity and speed up queries. Additional rules such as `NOT NULL`, `CHECK` constraints (e.g., non-negative amounts), and unique MSISDNs enhance accuracy and security. Overall, this design balances normalization with usability while supporting real-time queries and long-term reporting needs.


gabriella@AHIRWE: ~
mysql> CREATE TABLE transaction_participants(
-> tx_id INT NOT NULL COMMENT 'Foreign key referencing the related transaction',
-> user_id INT NOT NULL COMMENT 'Foreign key referencing the participating user',
-> role VARCHAR(20) NOT NULL COMMENT 'Role of the participant in the transaction (sender, receiver, agent, merchant)',
-> PRIMARY KEY (tx_id, user_id, role),
-> CONSTRAINT fk_tp_tx
-> FOREIGN KEY (tx_id) REFERENCES transactions(tx_id)
-> ON UPDATE CASCADE ON DELETE CASCADE,
-> CONSTRAINT fk_tp_user
-> FOREIGN KEY (user_id) REFERENCES users(user_id)
ON -> ON UPDATE RESTRICT ON DELETE RESTRICT,
-> CHECK (role IN ('sender','receiver','agent','merchant'))
->) COMMENT='Participants and roles per transaction';
Query OK, 0 rows affected (0.09 sec)
mysql> CREATE TABLE system_logs(
-> log_id INT AUTO_INCREMENT PRIMARY KEY COMMENT 'Unique identifier for each log entry',
-> event_type VARCHAR(40) NOT NULL COMMENT 'Type of event being logged (e.g., parsed, validated, inserted, error)',
-> message TEXT NOT NULL COMMENT 'Detailed description of the log event or error message',
-> tx_id INT NULL COMMENT 'Optional reference to the related transaction (if applicable)',
-> created_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT 'Timestamp when the log entry was created',
-> CONSTRAINT fk_log_tx
-> FOREIGN KEY (tx_id) REFERENCES transactions(tx_id)
N UPD -> ON UPDATE CASCADE ON DELETE SET NULL
->) COMMENT='ETL/process logs linked to transactions when relevant';
Query OK, 0 rows affected (0.09 sec)
mysql> SHOW DATABASES;
+--------------------+
| Database |
+--------------------+
| Momo_App |
| Practice |
| information_schema |
| mysql |
| performance_schema |
| sys |
+--------------------+
6 rows in set (0.01 sec)