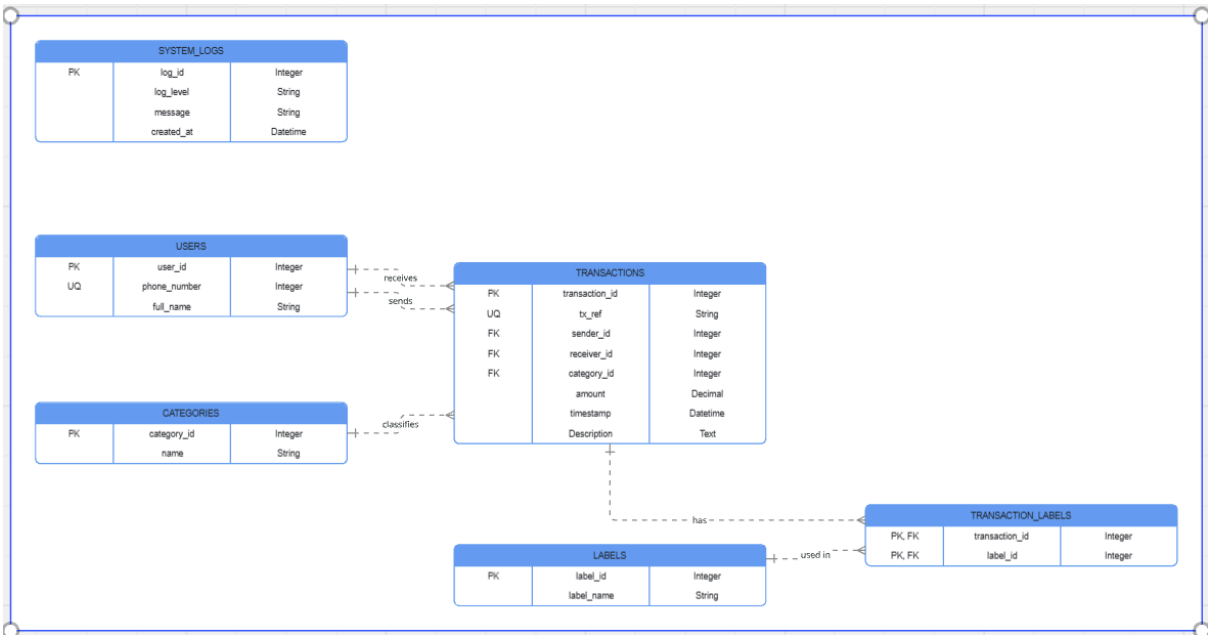
Week 2 Assignment – Task 2
Group:

1. Entity Relationship Diagram (ERD)

https://miro.com/app/board/uXjVGL5Z6Nk=/

Figure 1: Complete ERD

(Entities: Users, Categories, Transactions, Labels, Transaction_Labels, System_Logs)



2. Design Rationale and Justification

The database schema was designed to faithfully represent the MoMo SMS transaction processing requirements while ensuring performance, data integrity, and future scalability.

Core entities were derived directly from the ERD:

- **Users** stores sender/receiver information with phone_number as UNIQUE BIGINT for fast lookup and indexing.
- **Categories** classifies transaction types (airtime, P2P, bill payment, etc.) to enable grouping and reporting.
- **Transactions** is the central fact table with composite relationships to Users (sender & receiver) and Categories. tx_ref is UNIQUE VARCHAR to prevent duplicates from SMS parsing. Amount uses DECIMAL(15,2) to handle large transfers accurately.
- **Labels** and **Transaction_Labels** implement a many-to-many relationship for flexible tagging (fraud, high-value, promo, etc.).
- **System_Logs** captures parsing/processing audit trail, linked optionally to transactions.

Key decisions:

- InnoDB engine for referential integrity and ACID compliance.
- FOREIGN KEY constraints with RESTRICT on core relationships to prevent orphan records, CASCADE on junction/logs for cleanup.
- CHECK (amount >= 0) prevents invalid negative values.
- Indexes on timestamp, sender_id, receiver_id, tx_ref, and composite (category_id + timestamp) optimize common queries (history by date, per user, per category).
- Phone numbers as BIGINT improve numeric indexing vs VARCHAR. Column comments and ENUM for log_level enhance maintainability.

This design balances normalization, query performance, and real-world MoMo constraints while allowing easy extension (e.g., adding fraud detection flags).

3. Data Dictionary

Table: users

- user_id: INT AUTO_INCREMENT PRIMARY KEY
- phone_number: BIGINT NOT NULL UNIQUE COMMENT 'Phone number stored as integer for indexing (e.g. 254712345678)'
- full_name: VARCHAR(120) NOT NULL COMMENT 'Customer full name'
- created_at: DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT 'Record creation timestamp'

Table: categories

- category_id: INT AUTO_INCREMENT PRIMARY KEY
- name: VARCHAR(80) NOT NULL UNIQUE COMMENT 'Transaction type (e.g. Airtime Purchase, P2P Transfer)'
- created_at: DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP

Table: transactions

- transaction_id: INT AUTO_INCREMENT PRIMARY KEY
- tx_ref: VARCHAR(64) NOT NULL UNIQUE COMMENT 'MoMo reference ID from SMS/provider'
- sender_id: INT NOT NULL FOREIGN KEY → users.user_id
- receiver_id: INT NOT NULL FOREIGN KEY → users.user_id
- category_id: INT NOT NULL FOREIGN KEY → categories.category_id
- amount: DECIMAL(15,2) NOT NULL CHECK (amount >= 0) COMMENT 'Transaction amount in KES'
- timestamp: DATETIME NOT NULL COMMENT 'Transaction occurrence time'
- description: TEXT NULL COMMENT 'Optional SMS excerpt or notes'
- created_at: DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP

Table: labels

- label_id: INT AUTO_INCREMENT PRIMARY KEY

- label_name: VARCHAR(80) NOT NULL UNIQUE COMMENT 'Tag e.g. High Value, Potential Fraud'

Table: transaction_labels (junction)

- transaction_id: INT NOT NULL FOREIGN KEY → transactions
- label_id: INT NOT NULL FOREIGN KEY → labels
- assigned_at: DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
- PRIMARY KEY (transaction_id, label_id)

Table: system_logs

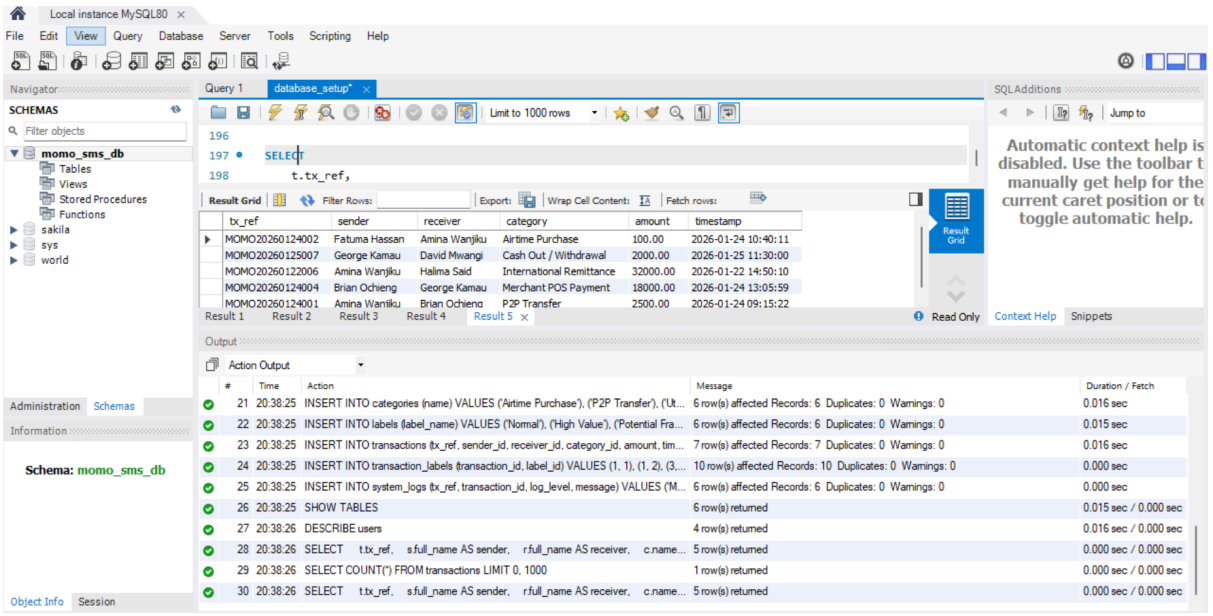
- log_id: INT AUTO_INCREMENT PRIMARY KEY
- tx_ref: VARCHAR(64) NULL
- transaction_id: INT NULL FOREIGN KEY → transactions
- log_level: ENUM('INFO','WARNING','ERROR','DEBUG') NOT NULL DEFAULT 'INFO'
- message: TEXT NOT NULL
- created_at: DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP

4. Sample Queries Demonstrating Functionality

Query 1: Recent high-value transactions (SELECT + JOIN)

SQL

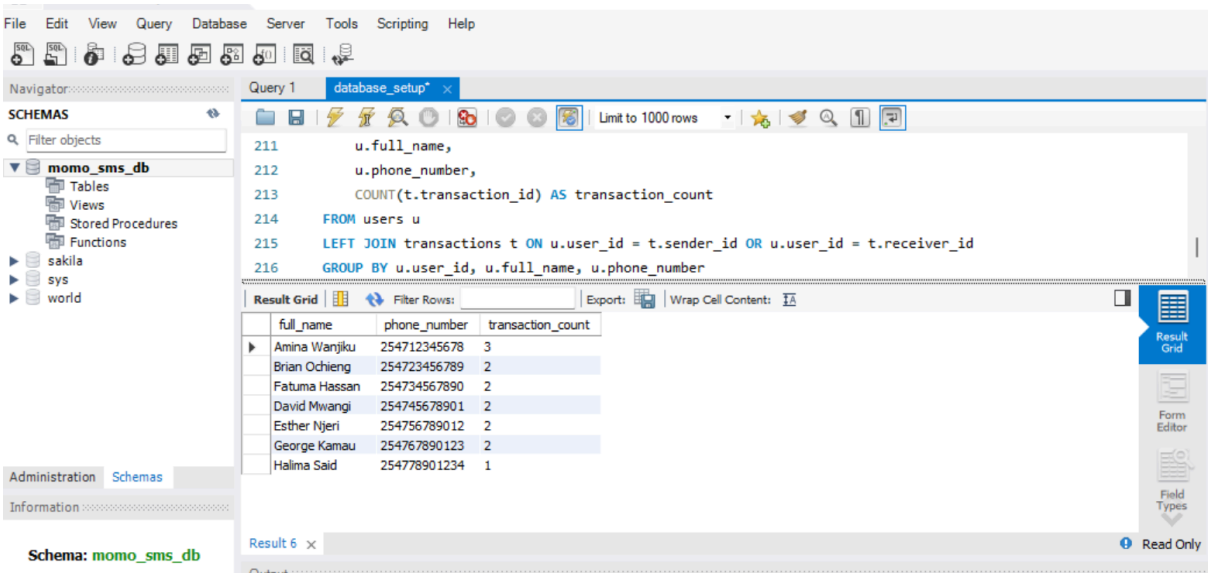
```
SELECT t.tx_ref, s.full_name AS sender, r.full_name AS receiver, c.name AS category, t.amount
FROM transactions t
JOIN users s ON t.sender_id = s.user_id
JOIN users r ON t.receiver_id = r.user_id
JOIN categories c ON t.category_id = c.category_id
WHERE t.amount > 5000
ORDER BY t.timestamp DESC
LIMIT 5;
```



Query 2: User transaction count

SQL

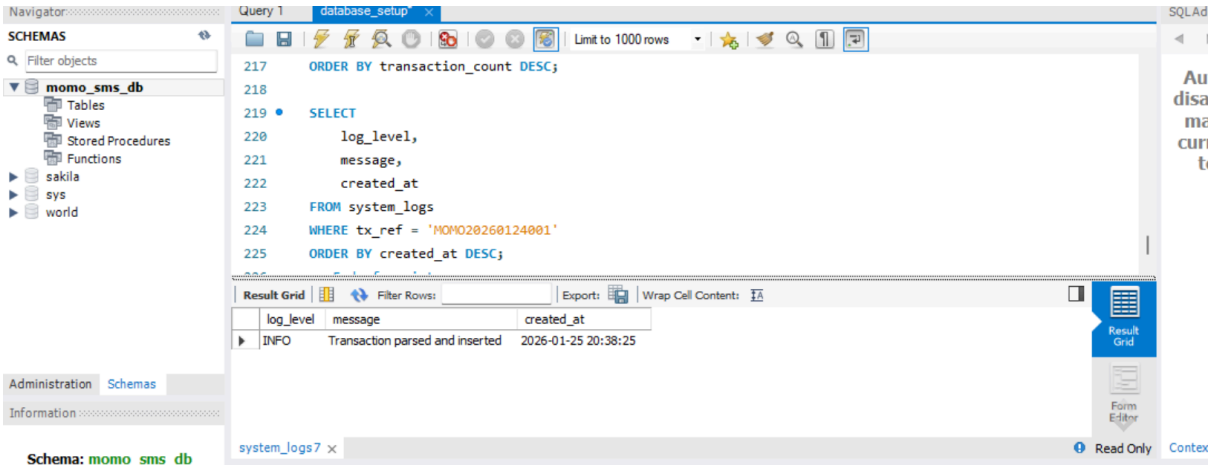
```
SELECT u.full_name, u.phone_number, COUNT(t.transaction_id) AS transaction_count
FROM users u
LEFT JOIN transactions t ON u.user_id = t.sender_id OR u.user_id = t.receiver_id
GROUP BY u.user_id
ORDER BY transaction_count DESC;
```



Query 3: Logs for a specific transaction (READ log)

SQL

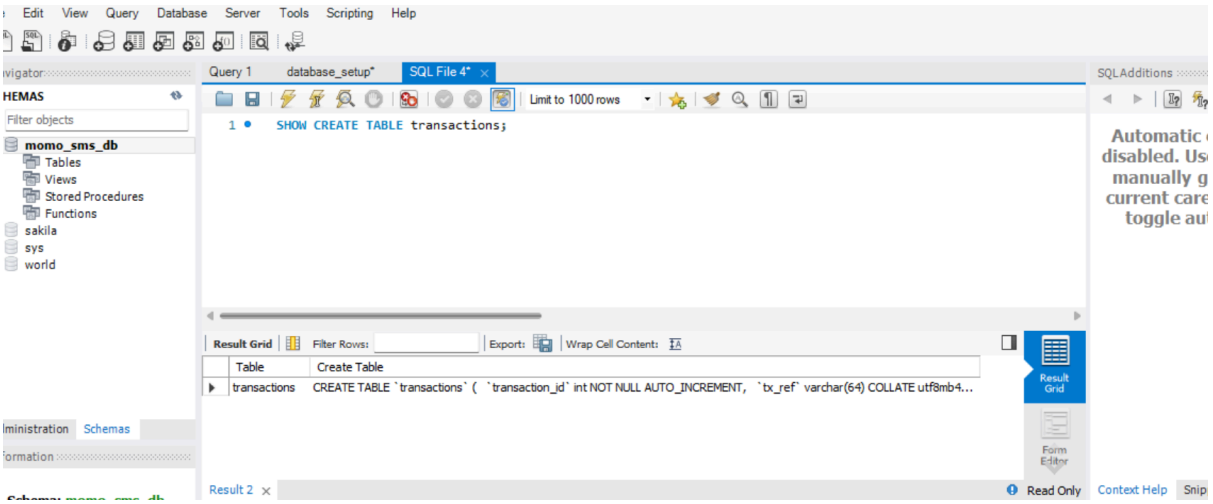
```
SELECT log_level, message, created_at
FROM system_logs
WHERE tx_ref = 'MOMO20260124001'
ORDER BY created_at DESC;
```



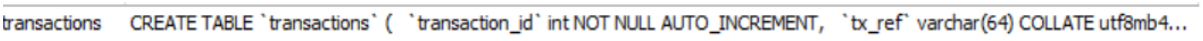
5. Unique Rules Added for Security & Accuracy (with screenshots)

Rule 1: CHECK constraint on amount (prevents negative/invalid amounts)

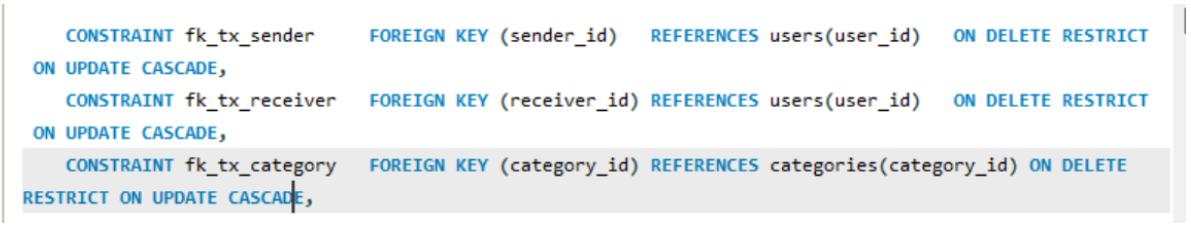
SQL
CHECK (amount >= 0)



Rule 2: UNIQUE constraint on tx_ref (prevents duplicate transactions)



Rule 3: FOREIGN KEY constraints with ON DELETE RESTRICT (protects data integrity)



Rule 4: phone_number UNIQUE in users (prevents duplicate user records)

```
CREATE TABLE users (
    user_id      INT AUTO_INCREMENT PRIMARY KEY,
    phone_number BIGINT       NOT NULL UNIQUE COMMENT 'Phone number (e.g. 2547xxxxxxxx format as
integer)',
```

These rules help maintain data accuracy (no negative money, no duplicates) and basic security (referential integrity).