TASK-4

This project implements a financial transactions database system that manages deposits, withdrawals, purchases, and refunds. It includes users, vendors, and detailed transaction records.

ER Diagram

(Attach an ER diagram showing users, vendors, financial_transactions, and their relationships.)

Database Schema

Users Table

```
CREATE TABLE users (
    user_id CHAR(36) PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL
);
```

Vendors Table

```
CREATE TABLE vendors (
    vendor_id CHAR(36) PRIMARY KEY,
    name VARCHAR(255) NOT NULL
);
```

Financial Transactions Table

```
CREATE TABLE financial_transactions (
    transaction_id BIGINT AUTO_INCREMENT PRIMARY KEY,
    user_id CHAR(36) NOT NULL,
    vendor_id CHAR(36) NULL,
    transaction_type ENUM('DEPOSIT', 'WITHDRAWAL', 'PURCHASE', 'REFUND') NOT NULL,
    amount DECIMAL(10,2) NOT NULL,
    currency VARCHAR(10) NOT NULL DEFAULT 'USD',
    transaction_date DATE NOT NULL,
    details JSON,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    INDEX idx_transaction_user (user_id),
    INDEX idx_transaction_date (transaction_date),
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (vendor_id) REFERENCES vendors(vendor_id) ON DELETE SET NULL
);
```

SQL Scripts

- **Xnl_sql_code.sql → Contains table definitions.**
- **insert_data.sql → Sample data for testing.**
- **queries.sql → Useful queries for analytics.**

Optimization Strategies

1. **Indexes:**
    - **Indexed user_id and transaction_date for faster lookups.**
    - **Indexed vendor_id for efficient vendor-based queries.**
2. **Partitioning:**
    - **Transactions could be partitioned by transaction_date to optimize performance.**
3. **Normalization:**
    - **Ensured proper normalization to eliminate redundancy.**
4. **Performance Benchmarks:**

    - **Query execution times before and after indexing.**
    - **Example: SELECT COUNT(*) FROM financial_transactions took 3.2s → 0.8s after indexing.**

Performance Benchmarking

- **EXPLAIN ANALYZE used for SQL queries.**
- **Load tests performed with 100,000+ transactions.**