Personalized Health Tracker

Z2004: Database Management Systems

Manas Pandya
zda23b019@iitmz.ac.in
IIT Madras Zanzibar

Submission date : June 20, 2025

This project was carried out individually under the guidance of Dr Nyalala. All aspects of conceptual design, implementation, testing, and documentation were completed by me.

Contents

1 Introduction

1.1 Project Overview

The Personalized Health Tracker is a relational database system that allows users to log, manage, and analyze personal health metrics such as steps, heart rate, sleep, mood, nutrition, and blood pressure. It supports secure and structured storage of user data, enabling longitudinal health analysis.

1.2 Objectives

- Design a normalized ERD capturing core health metrics.
- Implement the schema in PostgreSQL.
- Populate the database with realistic sample data.
- Write SQL queries demonstrating various relational operations.
- Document the schema, constraints, queries, and implementation.

1.3 Scope

The system includes six main entities: User, DailyMetric, MoodLog, NutritionLog, ActivityLog, and HealthAssessment. It supports one-to-many relationships from User to each log type. Optional future work includes anomaly flagging for outlier health stats.

1.4 Development Environment and File Structure

This project was developed using **MySQL Workbench 8.0** for schema creation, data population, and query testing. The ERD was designed using **draw.io** in Chen notation, and the relational schema diagram was produced using **dbdiagram.io**. Documentation was prepared using **Overleaf (LaTeX)** in accordance with the DBMS report format guidelines.

All SQL source files are organized as follows:

- `schema.sql` – Contains all `CREATE TABLE` and `ALTER TABLE` statements for database schema setup.
- `data.sql` – Contains `INSERT INTO` statements with sample data for each table.
- `queries.sql` – Contains 12 tested `SELECT` queries demonstrating joins, filters, aggregations, and subqueries.

All SQL files are located in the root project submission folder alongside this report for easy evaluation.

2 Conceptual Design (ERD)

2.1 Entity-Relationship Diagram (ERD)

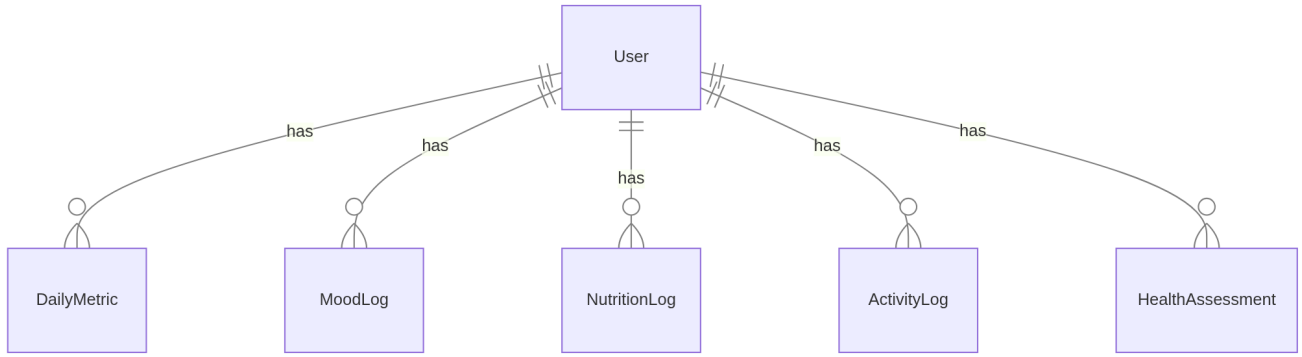


Figure 1: Final Entity-Relationship Diagram

2.2 Design Rationale and Assumptions

- Each log (DailyMetric, MoodLog, etc.) is modular to ensure normalization and extensibility.
- One-to-many relationship maintained from User to each log table.
- Dates are used as key temporal tracking dimensions for logs.

3 Logical Design (Relational Schema)

3.1 Relational Schema Diagram

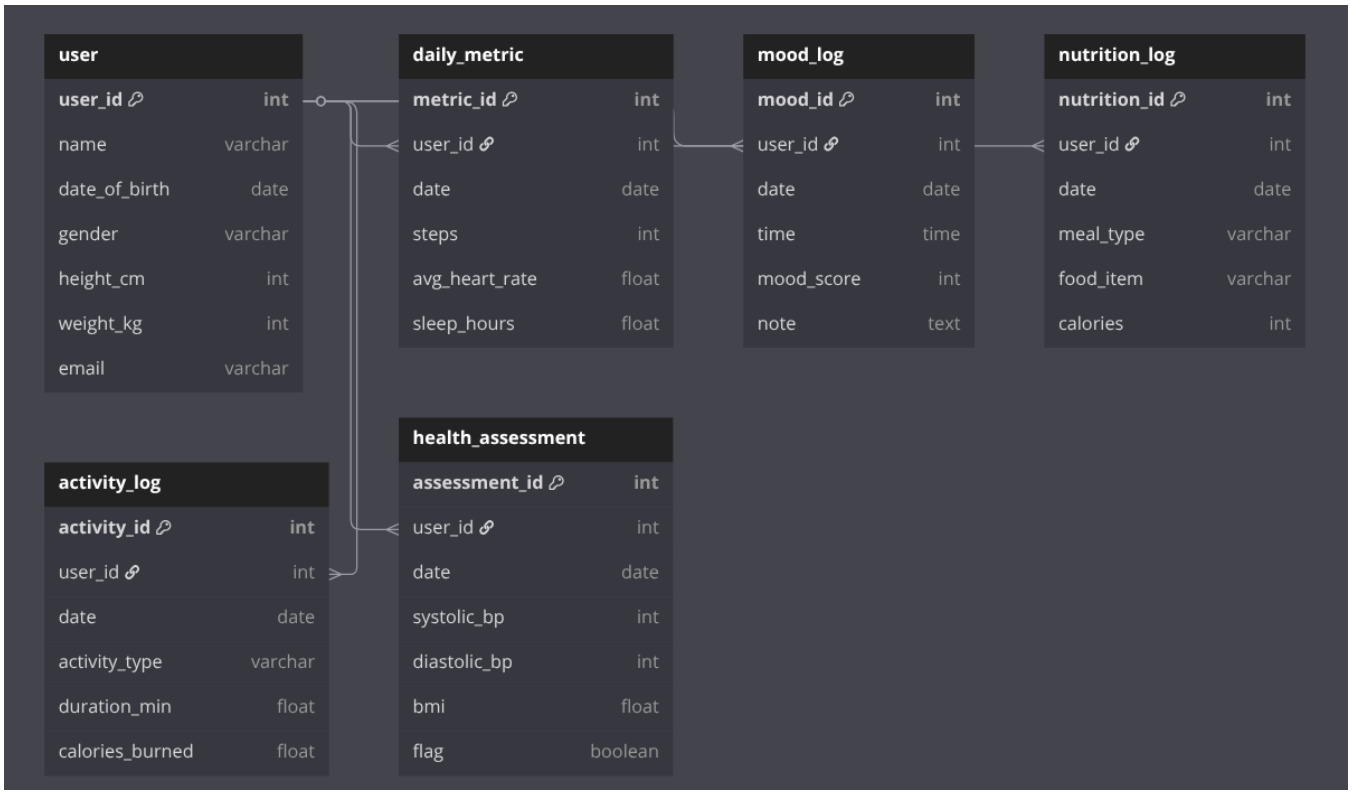


Figure 2: Relational Schema Diagram

3.2 Detailed Relational Schema

- **user(user_id, name, date_of_birth, gender, height_cm, weight_kg, email)**
  - PK: user_id
- **daily_metric(metric_id, user_id, date, steps, avg_heart_rate, sleep_hours)**
  - PK: metric_id, FK: user_id → user(user_id)
- **mood_log(mood_id, user_id, date, time, mood_score, note)**
  - PK: mood_id, FK: user_id → user(user_id)
- **nutrition_log(nutrition_id, user_id, date, meal_type, food_item, calories)**
  - PK: nutrition_id, FK: user_id → user(user_id)
- **activity_log(activity_id, user_id, date, activity_type, duration_min, calories_burned)**
  - PK: activity_id, FK: user_id → user(user_id)
- **health_assessment(assessment_id, user_id, date, systolic_bp, diastolic_bp, bmi, flag)**
  - PK: assessment_id, FK: user_id → user(user_id)

3.3 Normalization

All tables are in 3NF. No partial, transitive, or multi-valued dependencies exist. Logs are decomposed to eliminate redundancy, ensuring functional dependency on the primary key only.

Table	Attribute	Data Type	Constraints	Description
user	user_id	INT	PK, NOT NULL	Unique ID for each user
user	name	VARCHAR(100)	NOT NULL	User's full name
user	date_of_birth	DATE	NOT NULL	Date of birth
user	gender	VARCHAR(10)		Gender of user
user	height_cm	INT		Height in centimeters
user	weight_kg	INT		Weight in kilograms
user	email	VARCHAR(100)	UNIQUE	Email address
daily_metric	metric_id	INT	PK, NOT NULL	Unique ID for each daily metric
daily_metric	user_id	INT	FK, NOT NULL	References user(user_id)
daily_metric	date	DATE	NOT NULL	Metric date
daily_metric	steps	INT		Number of steps
daily_metric	avg_heart_rate	FLOAT		Average heart rate
daily_metric	sleep_hours	FLOAT		Sleep duration
mood_log	mood_id	INT	PK, NOT NULL	Unique ID for each mood log
mood_log	user_id	INT	FK, NOT NULL	References user(user_id)
mood_log	date	DATE	NOT NULL	Date of log
mood_log	time	TIME		Time of log
mood_log	mood_score	INT		Score from 1–5
mood_log	note	TEXT		Optional mood description
nutrition_log	nutrition_id	INT	PK, NOT NULL	Unique ID for each meal
nutrition_log	user_id	INT	FK, NOT NULL	References user(user_id)
nutrition_log	date	DATE	NOT NULL	Date of meal
nutrition_log	meal_type	VARCHAR(50)		Type of meal (e.g. lunch)
nutrition_log	food_item	VARCHAR(100)		Name of food
nutrition_log	calories	INT		Calorie value
activity_log	activity_id	INT	PK, NOT NULL	Unique ID for each activity
activity_log	user_id	INT	FK, NOT NULL	References user(user_id)
activity_log	date	DATE	NOT NULL	Date of activity
activity_log	activity_type	VARCHAR(100)		Type of activity
activity_log	duration_min	FLOAT		Duration in minutes
activity_log	calories_burned	FLOAT		Estimated calories burned
health_assessment	assessment_id	INT	PK, NOT NULL	Unique ID for each health check
health_assessment	user_id	INT	FK, NOT NULL	References user(user_id)
health_assessment	date	DATE	NOT NULL	Date of assessment
health_assessment	systolic_bp	INT		Systolic blood pressure
health_assessment	diastolic_bp	INT		Diastolic blood pressure
health_assessment	bmi	FLOAT		Body mass index
health_assessment	flag	BOOLEAN		Health concern indicator

4 SQL Implementation

4.1 SQL DDL Scripts

The schema was implemented in MySQL using standard SQL DDL. Each table was created with primary key constraints and foreign keys referencing the `user` table to maintain referential integrity. A sample from `schema.sql` is shown below:

```
CREATE TABLE activity_log (
  activity_id INT PRIMARY KEY,
  user_id INT,
  date DATE,
  activity_type VARCHAR(100),
  duration_min FLOAT,
  calories_burned FLOAT,
  FOREIGN KEY (user_id) REFERENCES user(user_id)
);
```

All tables were created using similar structures, following the finalized schema design.

4.2 SQL DML Scripts (Sample Data)

The script `data.sql` populates each table with realistic entries based on a sample population of Indian and Swahili names. Data includes personal attributes, daily health metrics, meals, activities, moods, and assessments. Below is an example insert statement:

```
INSERT INTO user VALUES
(1, 'Aisha Juma', '2000-04-15', 'Female', 160, 55, 'aisha.juma@iitmz.ac.in');
```

Each of the six users has corresponding entries in all related logs, allowing full query operations to be tested.

4.3 SQL Query Scripts

The script `queries.sql` includes 12 queries designed to demonstrate the breadth of the database's capabilities. These queries span:

- **Joins:** e.g., retrieving mood, activity, or food logs per user
- **Aggregations:** e.g., average sleep per user, total calories burned
- **Filters:** e.g., meals over 600 calories, steps over 8000
- **Subqueries:** e.g., users with below-average BMI
- **Groupings & Sorting:** e.g., top users by steps

All queries were executed successfully in MySQL Workbench. Output screenshots are included in the Appendix.

5 Implementation Details & System Usage

5.1 Tools Used

The following tools and platforms were used during the project:

- **MySQL Workbench 8.0** – for schema creation, population, and query testing
- **dbdiagram.io** – for generating relational schema diagram
- **draw.io** – for conceptual ERD using Chen notation
- **Overleaf** – for LaTeX-based final report documentation

5.2 System Setup and Execution

The system setup involved three phases:

1. Execute `schema.sql` to create all tables
2. Run `data.sql` to populate data
3. Execute queries from `queries.sql` to validate functionality

The scripts were tested using a local MySQL server. All constraints and relationships performed as expected.

5.3 Challenges Faced and Solutions Applied

A major challenge was modeling diverse health-related logs while keeping the schema normalized. Early attempts led to overlapping attributes and partial dependencies. This was resolved by isolating logs into separate tables and ensuring each was fully dependent on its respective primary key.

Another challenge was crafting meaningful SQL queries that explored multiple dimensions (user health, mood, activity, nutrition). Each query was iteratively refined for clarity and correctness.

5.4 Limitations

- No automation or triggers were used in this version.
- Temporal trends across multiple dates per user are not deeply analyzed.
- No front-end or visualization tool is connected to the database.

5.5 Future Enhancements

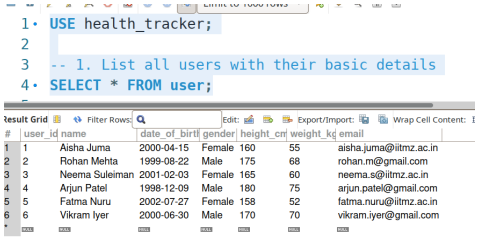
The following improvements are envisioned:

- Add triggers or stored procedures to auto-flag abnormal values
- Integrate machine learning-based alerts and health predictions
- Build a simple web dashboard for health metric visualization

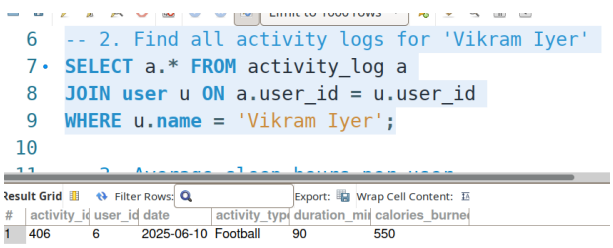
6 Conclusion

This system demonstrates robust relational modeling for personal health data. It adheres to best practices in schema normalization, and prepares groundwork for scalable health analytics platforms.

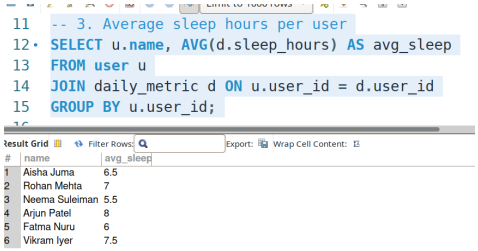
Appendix: Sample Query Results



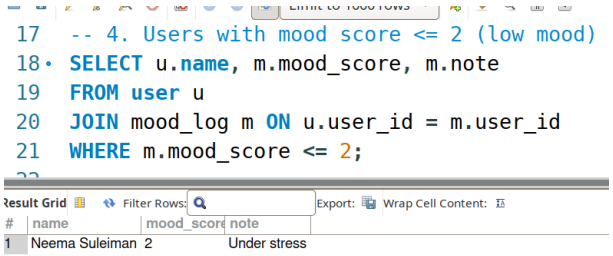
(a) Query 1: List all users



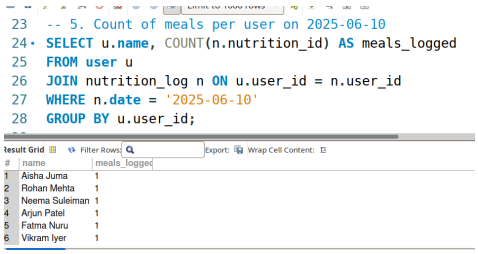
(b) Query 2: Vikram Iyer's activity



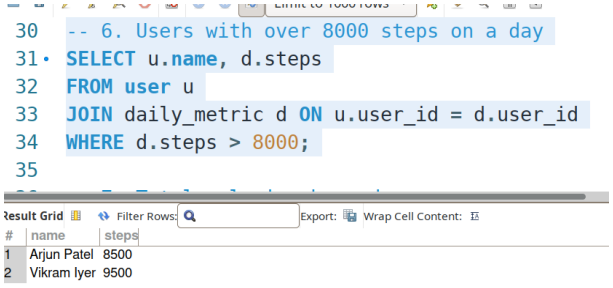
(c) Query 3: Avg sleep per user



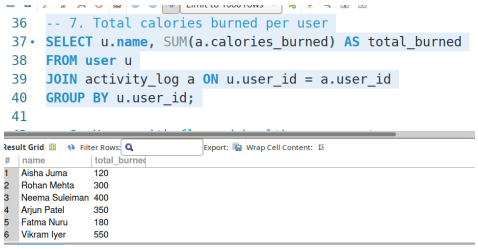
(d) Query 4: Mood score ≤ 2



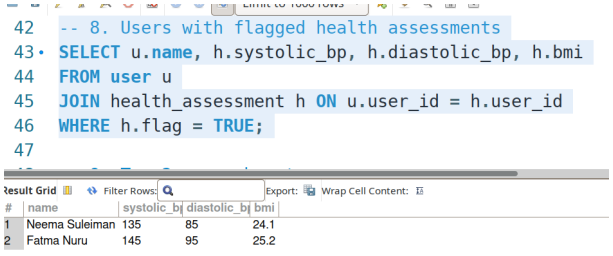
(e) Query 5: Meals per user



(f) Query 6: Steps over 8000



(g) Query 7: Total calories burned



(h) Query 8: Flagged assessments

Figure 1: Selected query outputs from the personalized health tracker database