SPRING2025

PERSONAL HEALTH DATA TRACKER

[Subject]

Kirill Kuznetsov

Table of Contents

The Case And Application Requirements

1.1 The Case/Business

With the increasing popularity and availability of wearable health technologies such as smartwatches, fitness trackers, and fitness bracelets, people now have unprecedented access to detailed personal health data. Despite the ease of use of the devices, it is still difficult for the user to standardize their data, set goals, and analyze the data to track health trends.

Personal Health Data Tracker aims to solve this problem by providing users with a centralized, easy-to-use database system. This system aggregates health data from wearable devices, allowing users to easily track their health metrics, analyze their progress, identify trends, and correlate certain activities with health outcomes. Ultimately, this project gives users a tool to understand their health data.

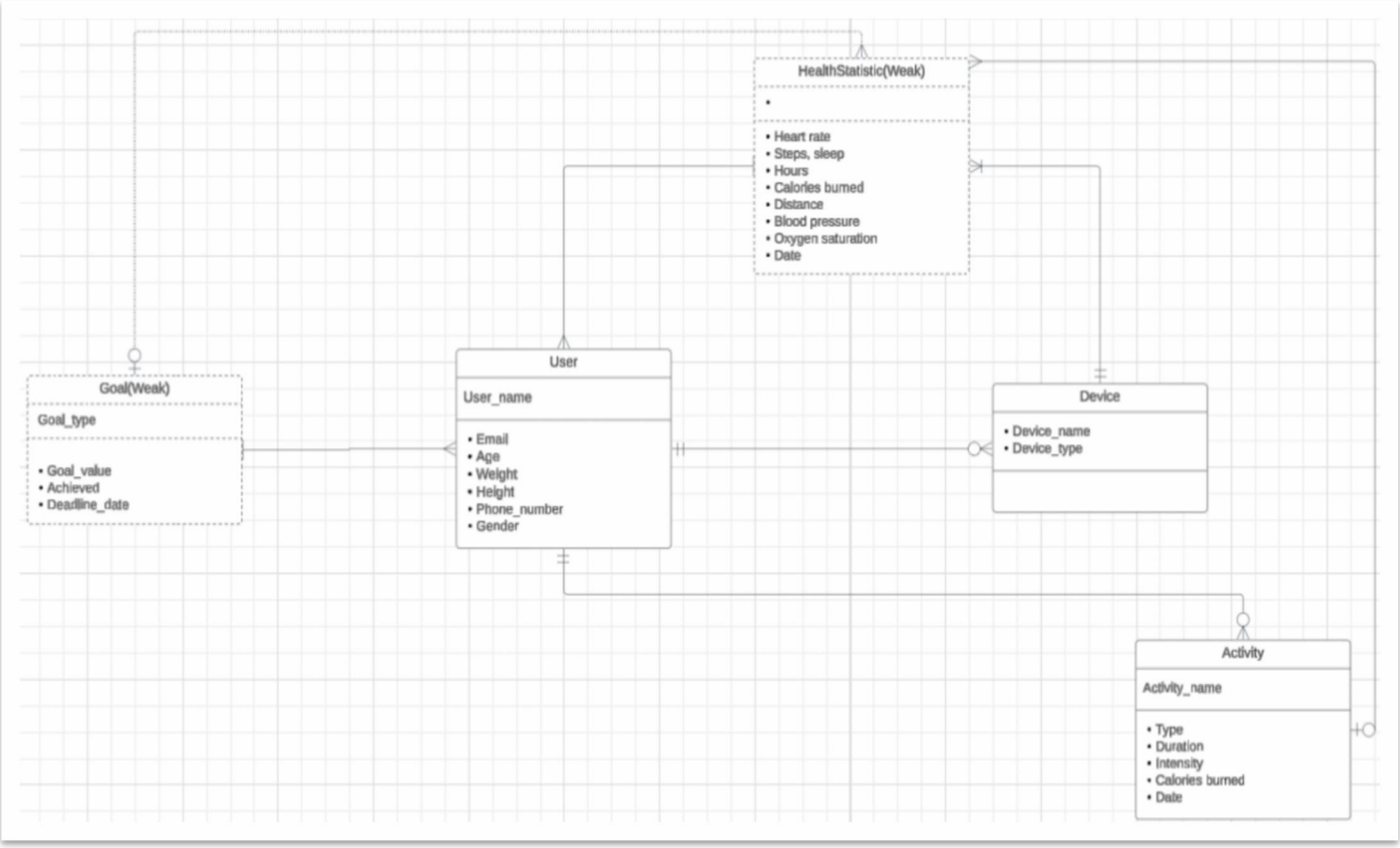
1.2 Application Requirements

he Personal Health Data Tracker must fulfill the following application requirements:

1. **Data Collection and Integration:**
    - Collect data from various wearable devices, including smartwatches and fitness trackers.
    - Capture essential health metrics such as heart rate, steps, sleep patterns, calories burned, distance covered, blood pressure, and oxygen saturation.
2. **User Information Management:**
    - Maintain basic user profile information, including name, age, email (unique identifier), height, weight, phone number (optional), and gender.
3. **Activity Tracking:**
    - Record and store detailed activity information such as type, duration (minutes), intensity, calories burned, and distance (km).
    - Associate each activity record with a specific user and date.
4. **Health Goals Management:**
    - Allow users to set personalized health goals (e.g., achieving a certain number of daily steps, burning specific calories, or improving sleep hours).
    - Track and manage progress toward these goals, clearly indicating achievement status and deadlines (if any).
5. **Device Management:**
    - Store details of user-linked wearable devices, including device type and device name/model.
6. **Health Data Analytics:**
    - Analyze collected data to provide meaningful insights, including health trends over time, activity correlation with health outcomes, and progress toward defined health goals.
7. **User-Friendly Querying and Reporting:**
    - Implement effective query capabilities for easy extraction and analysis of health data.
    - Enable retrieval of data that supports insights such as activity-performance correlation, trend analysis, and goal monitoring

Step 1 – Create Data Model From Application Requirements

1.1 Data Model



1.2 Supportive Documentation

The conceptual ER diagram outlines the core entities and relationships of the Personal Health Data Tracker system using Crow's Foot notation. It presents a logical view of the data without implementation details such as keys or data types.

Entities and Attributes:

- **User**: Email, Name, Age, Weight, Height, Phone number (optional), Gender
- **Device** *(weak)*: Device name, Device type — depends on User
- **Activity**: Activity name, Type, Duration, Intensity, Calories burned, Date
- **Goal** *(weak)*: Goal type, Goal value, Achieved, Deadline date — user-specific
- **HealthStatistic** *(weak)*: Heart rate, Steps, Sleep hours, Calories burned, Distance, Blood pressure, Oxygen saturation, Date — linked to User, and optionally to Activity and Device

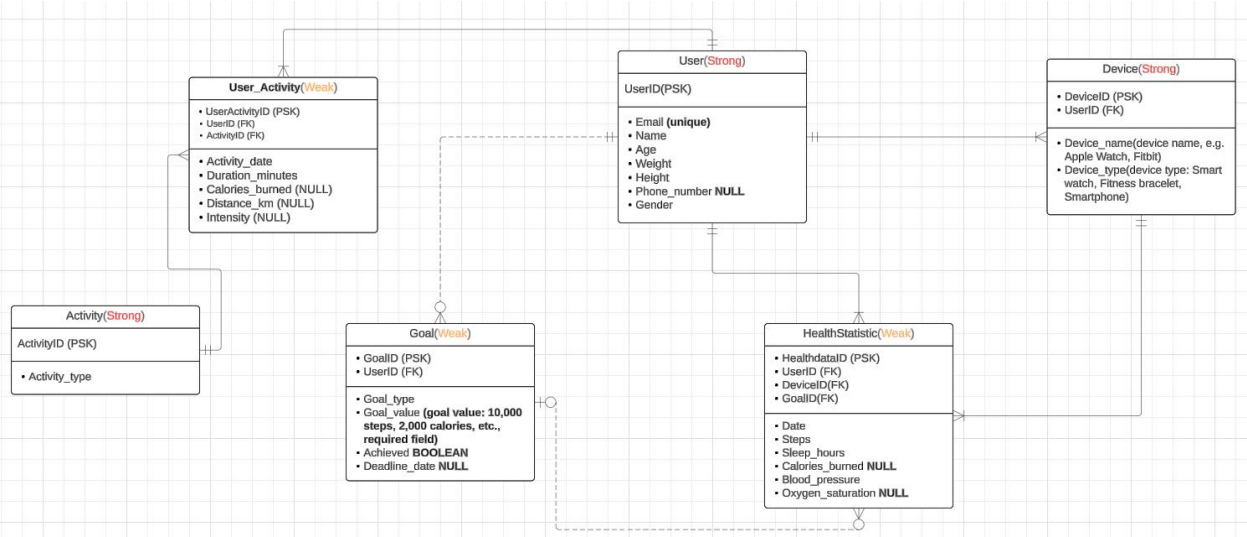
Relationships:

- One **User** can have many **Devices**, **Goals**, **Activities**, and **HealthStatistic** entries
- **Device**, **Goal**, and **HealthStatistic** are modeled as weak entities that rely on **User**
- **HealthStatistic** may also reference a specific **Device** and **Activity**

This model establishes a clear foundation for translating application requirements into a relational database design in the next phase.

Step 2 – Transform Data Model Into Database Design

2.1 Database Design



The relational database design for the *Personal Health Data Tracker* application is based on the conceptual ER model developed in Phase 1.1. The model has been transformed into a relational schema with appropriate **surrogate primary keys (PSKs)**, foreign keys (FKs), and descriptive attributes. This structure supports data consistency, integrity, and efficient querying.

Each table uses a surrogate key where needed for simplicity and performance. The schema is also structured to eliminate redundancy and support all system requirements defined earlier.

User

Represents individuals registered in the system.

- UserID (PSK) – surrogate primary key
- Email (UNIQUE) – candidate key
- Name
- Age
- Weight
- Height
- Phone_number (nullable)
- Gender

Device

Stores wearable devices associated with users.

- DeviceID (PSK)
- UserID (FK → User.UserID)`
- Device_name
- Device_type

Activity

Defines types of activities.

- ActivityID (PSK)
- Activity_name
- Type

User_Activity

Captures specific activity sessions performed by users.

- UserActivityID (PSK)
- UserID (FK)`
- ActivityID (FK)`
- Activity_date
- Duration_minutes
- Calories_burned (nullable)
- Distance_km (nullable)
- Intensity (nullable)

Goal

Tracks health-related goals set by users.

- GoalID (PSK)
- UserID (FK)`
- Goal_type
- Goal_value
- Achieved (Boolean)

- Deadline_date (nullable)

HealthStatistic

Holds daily health measurements and contextual data.

- Healthdata(PSK)
- UserID(FK)
- DeviceID(FK)
- GoalID(FK)
- Date
- Steps
- Sleep_hours
- Calories_burned(NULL)
- Blood_pressure
- Oxygen_soluration (NULL)

2.2 Normalization

The database schema for the *Personal Health Data Tracker* was carefully normalized through successive stages to improve data integrity, minimize redundancy, and support efficient and flexible data access. The normalization process has been completed up to the **Fourth Normal Form (4NF)**.

Below is a detailed explanation of the normalization steps applied, with examples drawn from the HealthStatistic table

HEALTHSTATISTIC(UserID, DeviceID, ActivityID, GoalID, Date, Steps, Sleep_hours, Calories_burned, Distance, Blood_pressure, Oxygen_saturation)

Primary Key: HealthDataID (PSK)

For 1NF

Requirements: All attributes must be atomic, and each record must be uniquely identifiable

Problem:

- Possible non-atomic values, such as:
    - Blood_pressure = "120/80, 130/90"
    - Sleep_hours = "6.5, 8.0"

Solution:

- Ensure all values are atomic (single value per field)
- One row = one measurement instance

For 2NF

Requirements:

- Must satisfy 1NF
- All non-key attributes must be **fully functionally dependent** on the **entire** primary key (no partial dependency)

Current Primary Key:

- HealthDataID (surrogate key — PSK)

Explanation:

- Since we use a **single-column surrogate PK**, no chance of partial dependencies

- Every attribute in the row depends **only** on HealthDataID

For 3NF

Requirements:

- Must satisfy 2NF
- There must be no transitive dependencies (i.e., no non-key attribute depending on another non-key attribute)

Example Transitive Dependency

> GoalID → UserID
> (If a goal already implies a user, storing both may introduce redundancy)

Solution:

- Keep GoalID and UserID for access and JOIN purposes
- Recognize transitive logic, but avoid derived or duplicate values
- Keep data **normalized but practical**

Result: Table satisfies 3NF — transitive dependencies are avoided or managed

For 4NF

Requirements:

- Must satisfy 3NF (and BCNF)
- No multivalued dependencies: no attribute should independently repeat for the same key

Potential problem

UserID →→ DeviceID

UserID →→ ActivityID

If both are stored in one row:

- User has multiple devices and multiple activities → violates 4NF

Solution:

- Device and Activity are moved to separate tables

- HealthStatistic refers to only one DeviceID and one ActivityID (optional FKs)
- If both are involved → create multiple HealthStatistic rows

Result: Multivalued dependencies are eliminated → 4NF achieved

To summarize all written

1NF — Atomic attributes, unique records (HealthDataID)

2NF — No partial dependencies (single-column PK)

3NF — No transitive dependencies (GoalID/UserID handled properly)

4NF — Multivalued dependencies resolved via table separation

While this section focuses on the normalization of the HealthStatistic table as a representative example, all other tables in the database — including User, Device, Activity, Goal, and User_Activity — have been reviewed and also comply with normalization rules up to **Fourth Normal Form (4NF)**. This ensures a consistent and efficient relational structure across the entire database.

2.3 Data Dictionary

USER Tabel

Attribute Name	Data Type	Size	Null / Not Null	Default	Description
UserID	INT	–	Not Null	AUTO_INCREMENT	Surrogate primary key for each user
Email	VARCHAR	100	Not Null	-	Unique email address (candidate key)
Name	VARCHAR	100	Not Null	-	Full name of the user
Age	INT	-	Not Null	-	Age of the user
Weight	DECIMAL	5.2	Not Null	-	User's weight in kilograms
Height	DECIMAL	5.2	Not Null	-	User's height in centimeters
Phone_number	VARCHAR	20	Null	-	Optional phone number
Gender	CHAR	1	Not Null	-	Gender (M,F,O)

DEVICE Tabel

Attribute Name	Data Type	Size	Null / Not Null	Default	Description

DeviceID	INT	-	Not Null	AUTO_INCREMENT	Surrogate primary key for each device
UserID	INT	-	Not Null	-	Foreign key referencing User(UserID)
Device_name	VARCHAR	100	Not Null	-	Name or model of the device
Device_type	VARCHAR	50	Not Null	-	Type of device(smart watches, fitness tracker)

ACTIVITY Table

Attribute Name	Data Type	Size	Null / Not Null	Default	Description
ActivityID	INT	-	Not Null	AUTO_INCREMENT	Surrogate primary key for each activity
Activity_name	VARCHAR	100	Not Null	-	Name of activity
Type	VARCHAR	50	Not Null	-	Category or type of the activity

USER_ACTIVITY Table

Attribute Name	Data Type	Size	Null / Not Null	Default	Description

UserActivityID	INT	–	Not Null	AUTO_INCREMENT	Surrogate primary key for each user activity session
UserID	INT	-	Not Null	-	Foreign key referencing User(UserID)
ActivityID	INT	-	Not Null	-	Foreign key referencing Activity(ActivityID)
Activity_date	DATE	-	Not Null	-	Date when the activity occurred
Duration_minutes	INT	-	Not Null	-	Duration of the activity in minutes
Calories_burned	DECIMAL	6.2	Null	-	Calories burned during the activity (optional)
Distance_km	DECIMAL	6.2	Null	-	Distance covered during the activity in kilometers (optional)
Intensity	VARCHAR	20	Null	-	Intensity level of the activity (e.g., Low, Medium, High)

GOAL Table

Attribute Name	Data Type	Size	Null / Not Null	Default	Description

GoalID	INT	–	Not Null	AUTO_INCREMENT	Surrogate primary key for each goal
UserID	INT	-	Not Null	-	Foreign key referencing User(UserID)
Goal_type	VARCHAR	50	Not Null	-	Type of goal (e.g., Steps, Sleep, Calories)
Goal_value	DECIMAL	6.2	Not Null	-	Target value for the goal
Achieved	BOOLEAN	-	Not Null	FALSE	Whether the goal has been achieved
Deadline_date	DATE	-	Null	-	Optional deadline for achieving the goal

HEALTH_STATISTIC Table

Attribute Name	Data Type	Size	Null / Not Null	Default	Description

HealthDataID	INT	–	Not Null	AUTO_INCREMENT	Surrogate primary key for each goal
UserID	INT	-	Not Null	-	Foreign key referencing User(UserID)
DeviceID	INT	-	Not Null	-	Foreign key referencing Device(DeviceID)
GoalID	INT	-	Null	-	Optional foreign key referencing Goal(GoalID)
Date	DATA	-	Not Null	-	Date of measurement
Steps	INT	-	Not Null	0	Number of steps taken on this date
Sleep_hours	DECIMAL	4.2	Not Null	0.00	Number of hours slept
Calories_burned	DECIMAL	6.2	Null	-	Calories burned (optional)
Distance	DECIMAL	6.2	Not Null	0.00	Distance covered in kilometers
Blood_pressure	VARCHAR	15	Not Null	-	Blood pressure reading
Oxygen_saturation	INT	-	Null	-	Oxygen saturation percentage (optional)

2.4 Minimum Cardinality Enforcement

Parent: User

Child: Device

Parent Required	Action On Parent	Action On Child
INSERT	None	Get a UserID (FK must be provided)
MODIFY	Prohibit – UserID is referenced	Prohibit – must match valid UserID
DELETE	Cascade – delete all related Device records	None

Parent: User

Child: Goal

Parent Required	Action On Parent	Action On Child
INSERT	None	Must provide valid UserID (FK)
MODIFY	Prohibit — UserID is referenced in Goal	Prohibit — FK must match existing UserID
DELETE	Cascade — delete all related Goal records	None

Parent: User

Child: User_Activity

Parent Required	Action On Parent	Action On Child
INSERT	None	Must provide valid UserID (FK NOT NULL)
MODIFY	Prohibit — UserID is referenced in activities	Prohibit — FK must point to existing User
DELETE	Cascade — delete all related User_Activity	None

Parent: Activity

Child: User_Activity

Parent Required	Action On Parent	Action On Child
INSERT	None	Must reference existing ActivityID
MODIFY	Prohibit — ActivityID is used in child table	Prohibit — FK must point to valid activity
DELETE	Prohibit — can't delete an activity if referenced	None

Parent: User

Child: HealthStatistic

Parent Required	Action On Parent	Action On Child
INSERT	None	Must provide valid UserID (FK NOT NULL)
MODIFY	Prohibit — UserID is referenced in child table	Prohibit — must match existing UserID
DELETE	Cascade — delete all related HealthStatistic rows	None

Parent: Device

Child: HealthStatistic

Parent Required	Action On Parent	Action On Child
INSERT	None	If used, must provide valid DeviceID (FK)
MODIFY	Prohibit — DeviceID is referenced	Prohibit — FK must match existing device
DELETE	Set NULL or Restrict — depending on chosen DB rule	None

Parent: Goal

Child: HealthStatistic

Parent Required	Action On Parent	Action On Child
INSERT	None	If used, must provide valid GoalID (FK is optional)
MODIFY	Prohibit — GoalID is referenced	Prohibit — must match existing GoalID
DELETE	Set NULL or Restrict — depending on DB rule	None

STEP 3 – DATABASE IMPLEMENTATION

3.1 Database Creation

The following SQL statements define the relational schema for the Personal Health Data Tracker database. The schema was implemented using MySQL, as it supports relational integrity features such as foreign keys, cascading deletes, and default values.

Each table has been created with appropriate:

- **Primary and foreign keys**
- **NOT NULL constraints**
- **AUTO_INCREMENT primary keys**
- **ON DELETE rules** to enforce minimum cardinality

This setup ensures data consistency, referential integrity, and easy scalability in the future.

```
CREATE DATABASE HealthTrackerDB;

USE HealthTrackerDB ;

CREATE TABLE User (
   UserID INT AUTO_INCREMENT PRIMARY KEY,
   Email VARCHAR(100) NOT NULL UNIQUE,
   Name VARCHAR(100) NOT NULL,
   Age INT NOT NULL,
   Weight DECIMAL(5,2) NOT NULL,
   Height DECIMAL(5,2) NOT NULL,
   Phone_number VARCHAR(20),
   Gender CHAR(1) NOT NULL
);
CREATE TABLE Device(
DeviceID INT AUTO_INCREMENT PRIMARY KEY,
USERID INT NOT NULL,
Device_name VARCHAR(100) NOT NULL,
Device_type VARCHAR(50) NOT NULL,
```

```
FOREIGN KEY (UserID) REFERENCES User(UserID)
    ON DELETE CASCADE
);
CREATE TABLE Activity(
ActivityID INT AUTO_INCREMENT PRIMARY KEY,
Activity_name VARCHAR(100) NOT NULL,
Type VARCHAR(50) NOT NULL
);
CREATE TABLE User_Activity(
UserActivityID INT AUTO_INCREMENT PRIMARY KEY,
UserID INT NOT NULL,
ActivityID INT NOT NULL,
Activity_date DATE NOT NULL,
Duration_minutes INT NOT NULL,
Calories_burned DECIMAL(6,2),
Distance_km DECIMAL(6,2),
Intensity VARCHAR(20),

FOREIGN KEY (UserID) REFERENCES User(UserID)
 ON DELETE CASCADE,

FOREIGN KEY (ActivityID) REFERENCES Activity(ActivityID)
   ON DELETE RESTRICT
);
CREATE TABLE Goal (
GoalID INT AUTO_INCREMENT PRIMARY KEY,
UserID INT NOT NULL,
Goal_Type VARCHAR(50) NOT NULL,
```

```
Goal_Value DECIMAL(6,2) NOT NULL,
Achieved BOOLEAN NOT NULL DEFAULT FALSE,
Deadline_date DATE,

FOREIGN KEY (UserID) REFERENCES User(UserID)
 ON DELETE CASCADE
);
CREATE TABLE HealthStatistic(
HealthDataID INT AUTO_INCREMENT PRIMARY KEY,
UserID INT NOT NULL,
DeviceID INT,
GoalID INT,
Date DATE NOT NULL,
Sleep_hours DECIMAL(4,2) NOT NULL DEFAULT 0.00,
Calories_burned DECIMAL(6,2),
Distance DECIMAL(6,2) NOT NULL DEFAULT 0.00,
Blood_pressure VARCHAR(15) NOT NULL,
Oxygen_saturation INT,
Steps INT,
FOREIGN KEY (UserID) REFERENCES User(UserID)
    ON DELETE CASCADE,
FOREIGN KEY (DeviceID) REFERENCES Device(DeviceID)
    ON DELETE SET NULL,
FOREIGN KEY (GoalID) REFERENCES Goal(GoalID)
    ON DELETE SET NULL
);
```

3.2 Insertion of Sample/Test Data

The following SQL `INSERT INTO` statements populate the database with sample data for testing.

Each table receives one or more rows to verify the schema integrity and relationships, such as foreign key constraints and cascading delete behavior.

Insert records into User

INSERT INTO User(Email, Name, Age, Weight, Height, Phone_number, Gender)

VALUES

('alice@example.com', 'Alice Johnson', 28, 65.5, 170.0, '1234567890', 'F'),

('bob@example.com', 'Bob Smith', 34, 80.0, 180.0, NULL, 'M'),

('kirill@exampletheBEStstudent.com', 'Kirill Kuznetov',19,73.0, 89011091950 ,'M');

UserID	Email	Name	Age	Weight	Height	Phone_number	Gender
1	alice@example.com	Alice Johnson	28	65.50	170.00	1234567890	F
2	bob@example.com	Bob Smith	34	80.00	180.00	NULL	M
3	kirill@exampletheBEStstudent.com	Kirill Kuznetov	19	73.00	180.00	89011091950	M
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Insert records into Device

INSERT INTO Device (UserID, Device_name, Device_type)

VALUES

(1, 'Fitbit Charge 5', 'Fitness Bracelet'),

(2, 'Apple Watch Series 7', 'Smartwatch'),

(3, 'Garmin Forerunner 245', 'Smartwatch');

DeviceID	USERID	Device_name	Device_type
1010	1	Fitbit Charge 5	Fitness Bracelet
1011	2	Apple Watch Series 7	Smartwatch
1012	3	Garmin Forerunner 245	Smartwatch
NULL	NULL	NULL	NULL

Insert records into Activity

```
ALTER TABLE Activity AUTO_INCREMENT = 100;

INSERT INTO Activity (Activity_name, Type)

VALUES

('Running', 'Cardio'),

('Walking', 'Cardio'),

('Swimming', 'Cardio'),

('Weight Lifting', 'Strength'),

('Sleeping', 'Rest');
```

ActivityID	Activity_name	Type
100	Running	Cardio
101	Walking	Cardio
102	Swimming	Cardio
103	Weight Lifting	Strength
104	Sleeping	Rest
NULL	NULL	NULL

Insert records into User_Activity

```
INSERT INTO User_Activity (

   UserID, ActivityID, Activity_date, Duration_minutes,

   Calories_burned, Distance_km, Intensity

)
```

VALUES

(1, 100, '2024-05-10', 45, 300.5, 5.2, 'High'),

(2, 101, '2024-05-11', 30, 180.0, 2.5, 'Medium'),

(3, 102, '2024-05-12', 60, 400.0, 1.0, 'Low');

UserActivityID	UserID	ActivityID	Activity_date	Duration_minutes	Calories_burned	Distance_km	Intensity
2010	1	100	2024-05-10	45	300.50	5.20	High
2011	2	101	2024-05-11	30	180.00	2.50	Medium
2012	3	102	2024-05-12	60	400.00	1.00	Low
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Insert records into Goal

ALTER TABLE Goal MODIFY Goal_value DECIMAL(8,2);

INSERT INTO Goal (UserID, Goal_type, Goal_value, Achieved, Deadline_date)

VALUES

(1, 'Steps', 10000, FALSE, '2024-06-01'),

(2, 'Sleep', 8, TRUE, '2024-05-10'),

(3, 'Calories', 500, FALSE, NULL);

GoalID	UserID	Goal_Type	Goal_value	Achieved	Deadline_date
3010	1	Steps	10000.00	0	2024-06-01
3011	2	Sleep	8.00	1	2024-05-10
3012	3	Calories	500.00	0	NULL
NULL	NULL	NULL	NULL	NULL	NULL

Insert records into HealthStatistic

```
INSERT INTO HealthStatistic (
    UserID, DeviceID, GoalID, Date,
    Sleep_hours, Calories_burned, Distance,
    Blood_pressure, Oxygen_saturation, Steps
)
VALUES
(1, 1010, 3010, '2024-05-15', 7.5, 350.25, 4.2, '120/80', 97, 10500),
(2, 1011, 3011, '2024-05-15', 6.0, 250.0, 3.1, '125/85', 95, 8800),
(3, 1012, 3012, '2024-05-15', 8.2, 500.0, 2.5, '130/90', 98, 12000);
```

HealthDataID	UserID	DeviceID	GoalID	Date	Sleep_hours	Calories_burned	Distance	Blood_pressure	Oxygen_saturation	Steps
4013	1	1010	3010	2024-05-15	7.50	350.25	4.20	120/80	97	10500
4014	2	1011	3011	2024-05-15	6.00	250.00	3.10	125/85	95	8800
4015	3	1012	3012	2024-05-15	8.20	500.00	2.50	130/90	98	12000
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

STEP 4 – QUERY PROCESSING

4.1 Query Implementation

Query 1 – Show all users and their basic data

User Request:

Show a list of all users including name, email, age, weight, and height.

```
SELECT Name, Email, Age, Weight, Height
FROM User;
```

Name	Email	Age	Weight	Height
Alice Johnson	alice@example.com	28	65.50	170.00
Bob Smith	bob@example.com	34	80.00	180.00
Kirill Kuznetov	kirill@exampletheBESTstudent.com	19	73.00	180.00

Query 2 – Show all 'Cardio' activities

User Request:

List all activities of type 'Cardio'.

```
SELECT Activity_name
 FROM Activity
WHERE Type = 'Cardio';
```

Activity_name
Running
Walking
Swimming

Query 3 – List all devices and which user they belong to

User Request:

Show all devices with device name, type, and user ID.

```
SELECT Device_name, Device_type, UserID
FROM Device;
```

	Device_name	Device_type	UserID
▶	Fitbit Charge 5	Fitness Bracelet	1
	Apple Watch Series 7	Smartwatch	2
	Garmin Forerunner 245	Smartwatch	3

Query 4 – Show all unachieved goals

User Request:

Show goals that have not been achieved.

```
SELECT *
 FROM Goal
WHERE Achieved = FALSE;
```

	GoalID	UserID	Goal_Type	Goal_value	Achieved	Deadline_date
▶	3010	1	Steps	10000.00	0	2024-06-01
	3012	3	Calories	500.00	0	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL

Query 5 – Number of activities per user

User Request:

Show how many activities each user has completed.

(For clear ex I decided to add extra activity for user)

```
INSERT INTO User_Activity (
    UserID, ActivityID, Activity_date, Duration_minutes,
    Calories_burned, Distance_km, Intensity
)
VALUES
(2,102, '2024-05-11', 30, 180.0, 2.5, 'Medium')
;
```

```
SELECT UserID, COUNT(*) AS ActivityCount
FROM User_Activity
GROUP BY UserID;
```

	UserID	ActivityCount
▶	1	1
	2	2
	3	1

Query 6 – All health statistics for one user

User Request:

Show all health statistics for user with ID = 1.

(Just for ex)

```
INSERT INTO HealthStatistic (
    UserID, DeviceID, GoalID, Date,
    Sleep_hours, Calories_burned, Distance,
    Blood_pressure, Oxygen_saturation, Steps
)
VALUES
(1, 1010, 3010, '2024-05-16', 7.5, 350.25, 4.2, '110/88', 97, 7600),
(1, 1010, 3010, '2024-05-17', 10.4, 350.25, 4.2, '110/88', 97, 15000),
(1, 1010, 3010, '2024-05-18', 2.5, 350.25, 4.2, '110/88', 97, 2000),
;

SELECT *
FROM HealthStatistic
```

WHERE UserID = 1;

HealthDataID	UserID	DeviceID	GoalID	Date	Sleep_hours	Calories_burned	Distance	Blood_pressure	Oxygen_saturation	Steps
4013	1	1010	3010	2024-05-15	7.50	350.25	4.20	120/80	97	10500
4016	1	1010	3010	2024-05-16	7.50	350.25	4.20	110/88	97	7600
4017	1	1010	3010	2024-05-17	10.40	350.25	4.20	110/88	97	15000
4018	1	1010	3010	2024-05-18	2.50	350.25	4.20	110/88	97	2000
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query 7 – Average sleep per user

User Request:

Show average number of sleep hours per user.

SELECT UserID, AVG(Sleep_hours) AS AvgSleep

FROM HealthStatistic

GROUP BY UserID;

UserID	AvgSleep
1	6.975000
2	6.000000
3	8.200000

Query 8 – Users who burned more than 400 calories

User Request:

List health records where calories burned exceed 400.

SELECT *

FROM HealthStatistic

WHERE Calories_burned > 400;

HealthDataID	UserID	DeviceID	GoalID	Date	Sleep_hours	Calories_burned	Distance	Blood_pressure	Oxygen_saturation	Steps
4015	3	1012	3012	2024-05-15	8.20	500.00	2.50	130/90	98	12000
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query 9 – Join User and Goal to see name + goal info

User Request:

List each user's name with their goal type and status.

```
SELECT u.Name, g.Goal_type, g.Achieved
FROM User u
JOIN Goal g ON u.UserID = g.UserID;
```

Name	Goal_type	Achieved
Alice Johnson	Steps	0
Bob Smith	Sleep	1
Kirill Kuznetov	Calories	0

Query 10 – Sort activities by duration

User Request:

Show all user activities sorted by duration (highest first).

```
SELECT *
FROM User_Activity
ORDER BY Duration_minutes DESC;
```

UserActivityID	UserID	ActivityID	Activity_date	Duration_minutes	Calories_burned	Distance_km	Intensity
2012	3	102	2024-05-12	60	400.00	1.00	Low
2010	1	100	2024-05-10	45	300.50	5.20	High
2011	2	101	2024-05-11	30	180.00	2.50	Medium
2013	2	102	2024-05-11	30	180.00	2.50	Medium
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL