**Normalization in PoseFit: 1NF and 2NF**

# Introduction

In the PoseFit project, which involves various modules such as exercise tracking, mental wellness, nutrition, and more, it’s crucial to design a robust and scalable database. Normalization helps in reducing data redundancy, maintaining consistency, and improving the efficiency of queries. This document demonstrates normalization through two examples: Water Intake Tracker and Sleep Tracker. We will normalize one table to 1NF and the other to 2NF, including detailed SQL commands used in each transformation step.

# Part 1: Water Intake Tracker – Normalization to 1NF

## Step 1: Unnormalized Table Structure

We start with a table named Water\_Intake\_Tracker. It contains the following fields:  
- User\_ID  
- User\_Name  
- Date  
- Water\_Intake\_Times (multiple times in a single field)

## Unnormalized Table: Water\_Intake\_Tracker



**Problem:** The Water\_Intake\_Times column contains multiple values, which violates the First Normal Form (1NF) that requires all values to be atomic.

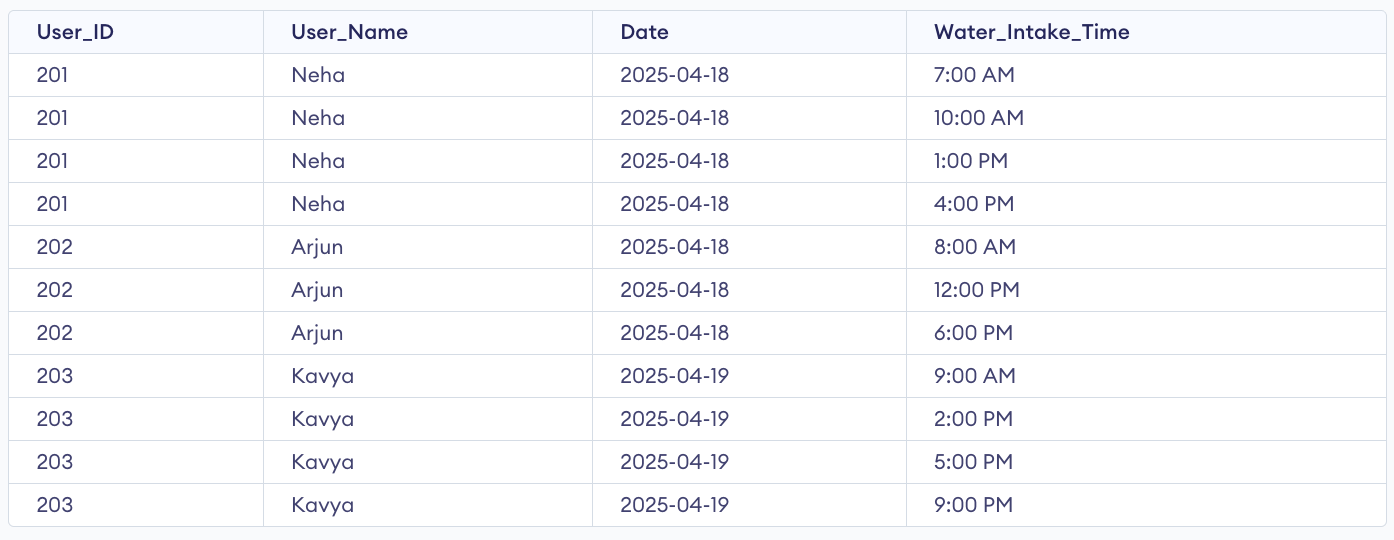
## Step 2: Converting to 1NF

We split the multivalued column into multiple rows, so each row holds one atomic value of water intake time.

SQL Steps:

-- Create new normalized table  
CREATE TABLE Water\_Intake\_1NF (  
 User\_ID INT,  
 User\_Name VARCHAR(50),  
 Date DATE,  
 Water\_Intake\_Time VARCHAR(20)  
);  
-- Insert data after splitting  
INSERT INTO Water\_Intake\_1NF VALUES (201, 'Neha', '2025-04-18', '7:00 AM');  
INSERT INTO Water\_Intake\_1NF VALUES (201, 'Neha', '2025-04-18', '10:00 AM');  
INSERT INTO Water\_Intake\_1NF VALUES (201, 'Neha', '2025-04-18', '1:00 PM');  
INSERT INTO Water\_Intake\_1NF VALUES (201, 'Neha', '2025-04-18', '4:00 PM');  
INSERT INTO Water\_Intake\_1NF VALUES (202, 'Arjun', '2025-04-18', '8:00 AM');  
INSERT INTO Water\_Intake\_1NF VALUES (202, 'Arjun', '2025-04-18', '12:00 PM');  
INSERT INTO Water\_Intake\_1NF VALUES (202, 'Arjun', '2025-04-18', '6:00 PM');

Resulting Table: Water\_Intake\_1NF



# Part 2: Sleep Tracker – Normalization to 2NF

## Step 1: Initial Table in 1NF

The initial table is already in 1NF but contains partial dependencies. User\_Name depends only on User\_ID, not the full primary key.

## Unnormalized Table: Sleep\_Tracker

## 

## Step 2: Convert to 2NF

We create two new tables:  
1. User (User\_ID, User\_Name)  
2. Sleep\_Entry (Entry\_ID, User\_ID, Sleep\_Date, Sleep\_Start, Sleep\_End, Sleep\_Quality)

SQL Steps:

-- Create User table  
CREATE TABLE User (  
 User\_ID INT PRIMARY KEY,  
 User\_Name VARCHAR(50)  
);  
  
-- Create Sleep\_Entry table  
CREATE TABLE Sleep\_Entry (  
 Entry\_ID INT PRIMARY KEY,  
 User\_ID INT,  
 Sleep\_Date DATE,  
 Sleep\_Start VARCHAR(20),  
 Sleep\_End VARCHAR(20),  
 Sleep\_Quality VARCHAR(20),  
 FOREIGN KEY (User\_ID) REFERENCES User(User\_ID)  
);  
  
-- Insert data into User  
INSERT INTO User VALUES (201, 'Neha');  
INSERT INTO User VALUES (202, 'Arjun');  
  
-- Insert data into Sleep\_Entry  
INSERT INTO Sleep\_Entry VALUES (301, 201, '2025-04-17', '10:30 PM', '6:30 AM', 'Good');  
INSERT INTO Sleep\_Entry VALUES (302, 202, '2025-04-17', '11:00 PM', '7:00 AM', 'Excellent');

## Final Normalized Tables

## User

## 

### Sleep\_Entry

### 

### Sleep\_Quality

### 

# Conclusion

Through these examples, we've shown step-by-step how to normalize unstructured tables into 1NF and 2NF. This process involves identifying multivalued fields and partial dependencies, and resolving them using SQL to create properly normalized, scalable relational schemas.