**DBMS PROJECT**

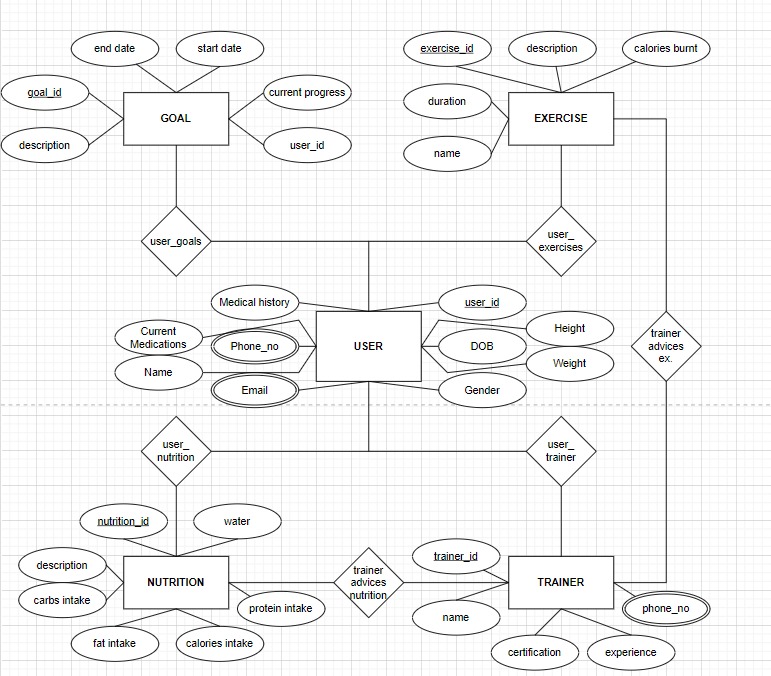
**GYM MANAGEMENT SYSTEM**

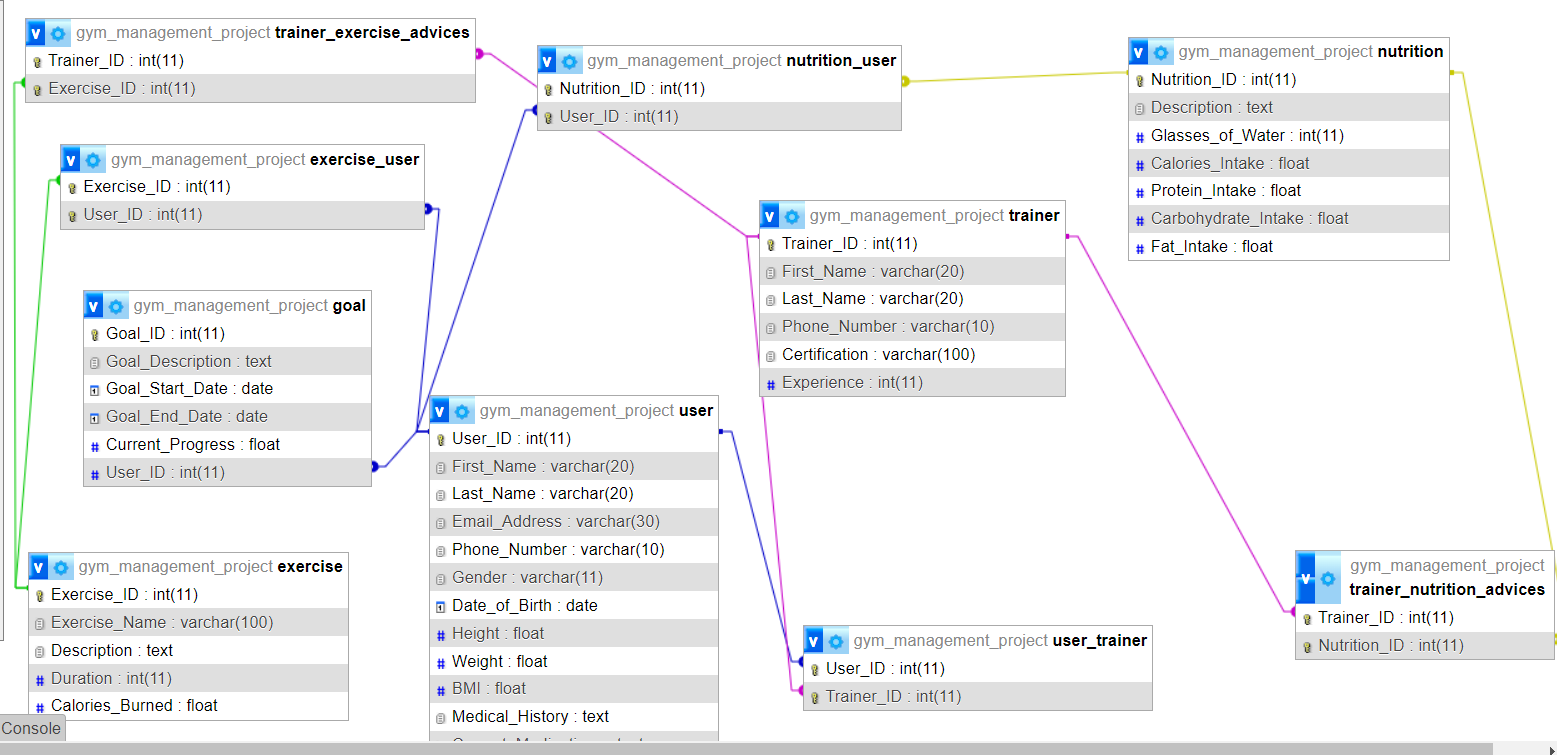
**PROBLEM STATEMENT:**

# Tarun Singh is the owner of a gym and fitness center. He wants to create a personal healthcare management system for his members that will help them track their fitness progress and manage their gym membership. The current system is manual and lacks an organized approach towards managing member data, which can lead to errors and inefficiencies.

# To address this issue, Tarun wants to create a new system that can store and manage member data in a secure and organized manner. The system should allow members to track their fitness progress, schedule appointments with trainers, and receive personalized workout plans. The system should also allow Tarun to manage member data, track member attendance, and generate reports to analyze the performance of the gym.

**ER MODEL**



****

**The following entites have been included in the ER model:**

**User:**The user has attributes such as user ID, name, email, phone number, and address,blood type, height, weight, allergies,e-mail,gender,BMI,medical history,current medications.

**Exercise**:The exercise entity has attributes such as exercise ID, name,description, duration, and calories burnt.

**Nutrition**:The nutrition entity has attributes such as nutrition ID, glasses of water, calories intake,protein intake,carbohydrate intake,fat intake.

**Trainer:**The trainer entity has attributes trainer id,name , phone no, certifications,experience.

**Goal:**The goal entity has attributes are description of goal, goal id, start date,end date,current progress,user id.

**The following Relationships have been made in the ER model:**

**User-Trainer:**is a many-to-one relationship between the user and trainer. Each user can have only one trainer, but each trainer may be associated with many users.

**Goal-User:**is a one-to-one relationship between the goal and user.

**User-Exercise:**This is many-to-many relationship between the user and exercise entities. Each user can have multiple recommended exercises, also each exercise is associated with many users.

**Trainer-Exercise:**This is a many-to-many relationship between the trainer and exercise entities. Each trainer can suggest multiple exercises, also each exercise is associated with many trainers.

**User-Nutrition:**This is a many-to-many relationship between the user and nutrition entities. Each user can have multiple nutrition entries, also eachnutrition entry is associated with many users.

**Trainer-Nutrition:**This is a many-to-many relationship between the trainer and nutrition entities. Each trainer can suggest multiple nutrition, also each nutrition can be suggested by many trainers.

**1)RELATIONAL SCHEMA:**

User(User\_ID, First\_Name, Last\_Name, Email\_Address, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight, BMI, Medical\_History, Current\_Medications)

Trainer(Trainer\_ID, First\_Name, Last\_Name, Phone\_Number, Certification, Experience)

Exercise(Exercise\_ID, Exercise\_Name, Description, Duration, Calories\_Burned) Nutrition(Nutrition\_ID, Description, Glasses\_of\_Water, Calories\_Intake, Protein\_Intake, Carbohydrate\_Intake, Fat\_Intake)

Goal(Goal\_Description, Goal\_Start\_Date, Goal\_End\_Date, Current\_Progress, User\_ID)

User\_Trainer(User\_ID, Trainer\_ID)

Exercise\_User(Exercise\_ID, User\_ID)

Nutrition\_User(Nutrition\_ID, User\_ID)

Trainer\_Exercise\_Advices(Trainer\_ID, Exercise\_ID)

Trainer\_Nutrition\_Advices(Trainer\_ID, Nutrition\_ID)

Userphone(User ID,Phone Number)

Useremail(User ID, Email\_Address)

**2)FUNCTIONAL DEPENDENCIES**

User\_ID -> First\_Name, Last\_Name, Email\_Address, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight, BMI, Medical\_History, Current\_Medications

Phone\_Number -> First\_Name, Last\_Name, Email\_Address, Gender, Date\_of\_Birth, Height, Weight, BMI, Medical\_History, Current\_Medications

Email\_Address -> First\_Name, Last\_Name, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight, BMI, Medical\_History, Current\_Medications

User\_ID -> Goal\_Description, Goal\_Start\_Date, Goal\_End\_Date, Current\_Progress

Exercise\_ID -> Exercise\_Name, Description, Duration, Calories\_Burned

Nutrition\_ID -> Description, Glasses\_of\_Water, Calories\_Intake, Protein\_Intake, Carbohydrate\_Intake, Fat\_Intake

User\_ID -> Trainer\_ID

Trainer\_ID -> First\_Name, Last\_Name, Phone\_Number, Certification, Experience

Trainer\_ID, Exercise\_ID -> Exercise\_Name, Description, Duration, Calories\_Burned

Trainer\_ID, Nutrition\_ID -> Description, Glasses\_of\_Water, Calories\_Intake, Protein\_Intake, Carbohydrate\_Intake, Fat\_Intake

User\_ID -> Phone\_Number

User\_ID -> Email\_Address

Height, Weight -> BMI

**3A)PARTIAL DEPENDENCIES**

**In User:**

User\_ID -> First\_Name, Last\_Name, Email\_Address, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight, BMI, Medical\_History, Current\_Medications

Phone\_Number -> First\_Name, Last\_Name, Email\_Address, Gender, Date\_of\_Birth, Height, Weight, BMI, Medical\_History, Current\_Medications

Email\_Address -> First\_Name, Last\_Name, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight, BMI, Medical\_History, Current\_Medications

Height, Weight -> BMI

**In Trainer:**

Trainer\_ID -> First\_Name, Last\_Name, Phone\_Number, Certification, Experience

**In Exercise:**

Exercise\_ID -> Exercise\_Name, Description, Duration, Calories\_Burned

**In Nutrition:**

Nutrition\_ID -> Description, Glasses\_of\_Water, Calories\_Intake, Protein\_Intake, Carbohydrate\_Intake, Fat\_Intake

**In Goal:**

User\_ID -> Goal\_Description, Goal\_Start\_Date, Goal\_End\_Date, Current\_Progress

**3B)TRANSITIVE DEPENDENCIES**

**In User:**

User\_ID -> Goal\_Description, Goal\_Start\_Date, Goal\_End\_Date, Current\_Progress (transitive through User\_Trainer relation)

**In Trainer:**

Trainer\_ID, Exercise\_ID -> Exercise\_Name, Description, Duration, Calories\_Burned (transitive through Trainer\_Exercise\_Advices relation)

Trainer\_ID, Nutrition\_ID -> Description, Glasses\_of\_Water, Calories\_Intake, Protein\_Intake, Carbohydrate\_Intake, Fat\_Intake (transitive through Trainer\_Nutrition\_Advices relation)

**In User\_Trainer**

User\_ID -> Trainer\_ID

**In Exercise\_User:**

Exercise\_ID -> User\_ID

**In Nutrition\_User:**

Nutrition\_ID -> User\_ID

**In Trainer\_Exercise\_Advices:**

Trainer\_ID -> Exercise\_ID

**In Trainer\_Nutrition\_Advices:**

Trainer\_ID -> Nutrition\_ID

**In Userphone**:

User ID -> Phone Number

**In Useremail:**

User ID -> Email\_Address

**4)CONVERSION TO 3RD NORMAL FORM**

**STEP 1:**

To convert the given relations to 1NF (First Normal Form), we need to make sure that each attribute within a relation contains atomic values. Here's how we can do that:

User(User\_ID, First\_Name, Last\_Name, Email\_Address, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight, BMI, Medical\_History, Current\_Medications)

1.Create a separate relation for Medical\_History and Current\_Medications since they are multi-valued attributes:

User\_Medical\_History(User\_ID, Medical\_History\_Item)

User\_Current\_Medications(User\_ID, Medication\_Item)

2.The BMI attribute is not atomic since it is calculated based on Height and Weight. We can remove the BMI attribute from the User relation and calculate it as needed:

User(User\_ID, First\_Name, Last\_Name, Email\_Address, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight)

Trainer(Trainer\_ID, First\_Name, Last\_Name, Phone\_Number, Certification, Experience)

Exercise(Exercise\_ID, Exercise\_Name, Description, Duration, Calories\_Burned)

Nutrition(Nutrition\_ID, Description, Glasses\_of\_Water, Calories\_Intake, Protein\_Intake, Carbohydrate\_Intake, Fat\_Intake)

Goal(Goal\_Description, Goal\_Start\_Date, Goal\_End\_Date, Current\_Progress, User\_ID)

User\_Trainer(User\_ID, Trainer\_ID)

Exercise\_User(Exercise\_ID, User\_ID)

Nutrition\_User(Nutrition\_ID, User\_ID)

Trainer\_Exercise\_Advices(Trainer\_ID, Exercise\_ID)

Trainer\_Nutrition\_Advices(Trainer\_ID, Nutrition\_ID)

Userphone(User\_ID, Phone\_Number)

Useremail(User\_ID, Email\_Address)

3.Note that the above relations are now in 1NF since each attribute contains atomic values.

**STEP 2:**

To convert the given relations to 2NF (Second Normal Form), we need to make sure that the relation is in 1NF and there are no partial dependencies. Here's how we can do that:

User(User\_ID, First\_Name, Last\_Name, Email\_Address, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight)

1. Create a separate relation for User medical information since it is a multi-valued attribute and does not depend on the primary key:

User\_Medical\_History(User\_ID, Medical\_History\_Item)

User\_Current\_Medications(User\_ID, Medication\_Item)

2. Create a separate relation for User phone numbers and email addresses since they are multi-valued attributes and do not depend on the primary key:

User\_Phone\_Number(User\_ID, Phone\_Number)

User\_Email\_Address(User\_ID, Email\_Address)

Goal(Goal\_Description, Goal\_Start\_Date, Goal\_End\_Date, Current\_Progress, User\_ID)

3. The Goal relation is already in 2NF since it only has one candidate key (User\_ID) and each non-key attribute is fully dependent on the key.

Trainer(Trainer\_ID, First\_Name, Last\_Name, Phone\_Number, Certification, Experience)

4. The Trainer relation is already in 2NF since it only has one candidate key (Trainer\_ID) and each non-key attribute is fully dependent on the key.

Exercise(Exercise\_ID, Exercise\_Name, Description, Duration, Calories\_Burned)

5. The Exercise relation is already in 2NF since it only has one candidate key (Exercise\_ID) and each non-key attribute is fully dependent on the key.

Nutrition(Nutrition\_ID, Description, Glasses\_of\_Water, Calories\_Intake, Protein\_Intake, Carbohydrate\_Intake, Fat\_Intake)

6. The Nutrition relation is already in 2NF since it only has one candidate key (Nutrition\_ID) and each non-key attribute is fully dependent on the key.

User\_Trainer(User\_ID, Trainer\_ID)

7. The User\_Trainer relation is already in 2NF since it only has two foreign keys (User\_ID and Trainer\_ID) that together form the primary key.

Exercise\_User(Exercise\_ID, User\_ID)

8. The Exercise\_User relation is already in 2NF since it only has two foreign keys (Exercise\_ID and User\_ID) that together form the primary key.

Nutrition\_User(Nutrition\_ID, User\_ID)

9. The Nutrition\_User relation is already in 2NF since it only has two foreign keys (Nutrition\_ID and User\_ID) that together form the primary key.

Trainer\_Exercise\_Advices(Trainer\_ID, Exercise\_ID)

10. The Trainer\_Exercise\_Advices relation is already in 2NF since it only has two foreign keys (Trainer\_ID and Exercise\_ID) that together form the primary key.

Trainer\_Nutrition\_Advices(Trainer\_ID, Nutrition\_ID)

11. The Trainer\_Nutrition\_Advices relation is already in 2NF since it only has two foreign keys (Trainer\_ID and Nutrition\_ID) that together form the primary key.

Therefore, all relations are now in 2NF.

**STEP 3:**

To convert the given relations to 3NF (Third Normal Form), we need to make sure that the relation is in 2NF and there are no transitive dependencies. Here's how we can do that:

User(User\_ID, First\_Name, Last\_Name, Email\_Address, Phone\_Number, Gender, Date\_of\_Birth, Height, Weight)

1. Create a separate relation for User medical information since it is a multi-valued attribute and does not depend on the primary key:

User\_Medical\_History(User\_ID, Medical\_History\_Item)

User\_Current\_Medications(User\_ID, Medication\_Item)

2. Create a separate relation for User phone numbers and email addresses since they are multi-valued attributes and do not depend on the primary key:

User\_Phone\_Number(User\_ID, Phone\_Number)

User\_Email\_Address(User\_ID, Email\_Address)

3. The User relation is now in 3NF since each attribute is directly dependent on the primary key.

Trainer(Trainer\_ID, First\_Name, Last\_Name, Phone\_Number, Certification, Experience)

4. The Trainer relation is already in 3NF since each non-key attribute is fully dependent on the key.

Exercise(Exercise\_ID, Exercise\_Name, Description, Duration, Calories\_Burned)

5. The Exercise relation is already in 3NF since each non-key attribute is fully dependent on the key.

Nutrition(Nutrition\_ID, Description, Glasses\_of\_Water, Calories\_Intake, Protein\_Intake, Carbohydrate\_Intake, Fat\_Intake)

6.The Nutrition relation is already in 3NF since each non-key attribute is fully dependent on the key.

Goal(Goal\_Description, Goal\_Start\_Date, Goal\_End\_Date, Current\_Progress, User\_ID)

7. The Goal relation is already in 3NF since each non-key attribute is fully dependent on the key.

User\_Trainer(User\_ID, Trainer\_ID)

8. The User\_Trainer relation is already in 3NF since each attribute is directly dependent on the primary key.

Exercise\_User(Exercise\_ID, User\_ID)

9. The Exercise\_User relation is already in 3NF since each attribute is directly dependent on the primary key.

Nutrition\_User(Nutrition\_ID, User\_ID)

10. The Nutrition\_User relation is already in 3NF since each attribute is directly dependent on the primary key.

Trainer\_Exercise\_Advices(Trainer\_ID, Exercise\_ID)

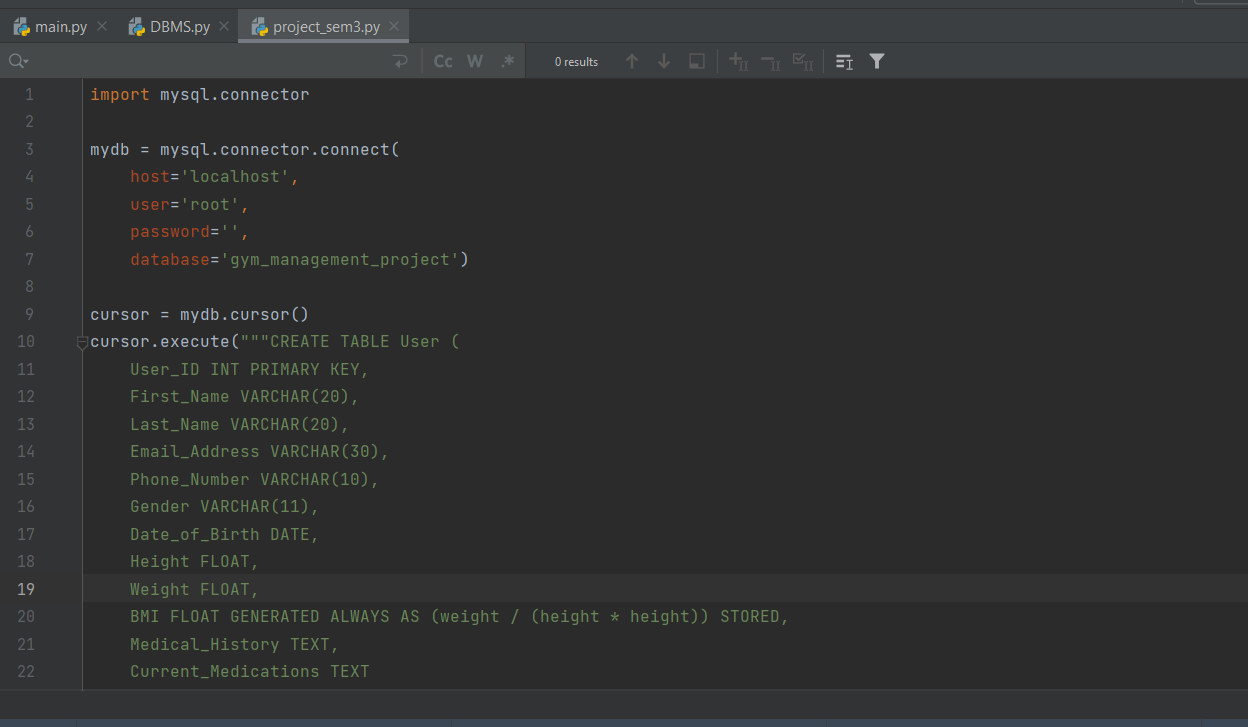
11. The Trainer\_Exercise\_Advices relation is already in 3NF since each attribute is directly dependent on the primary key.

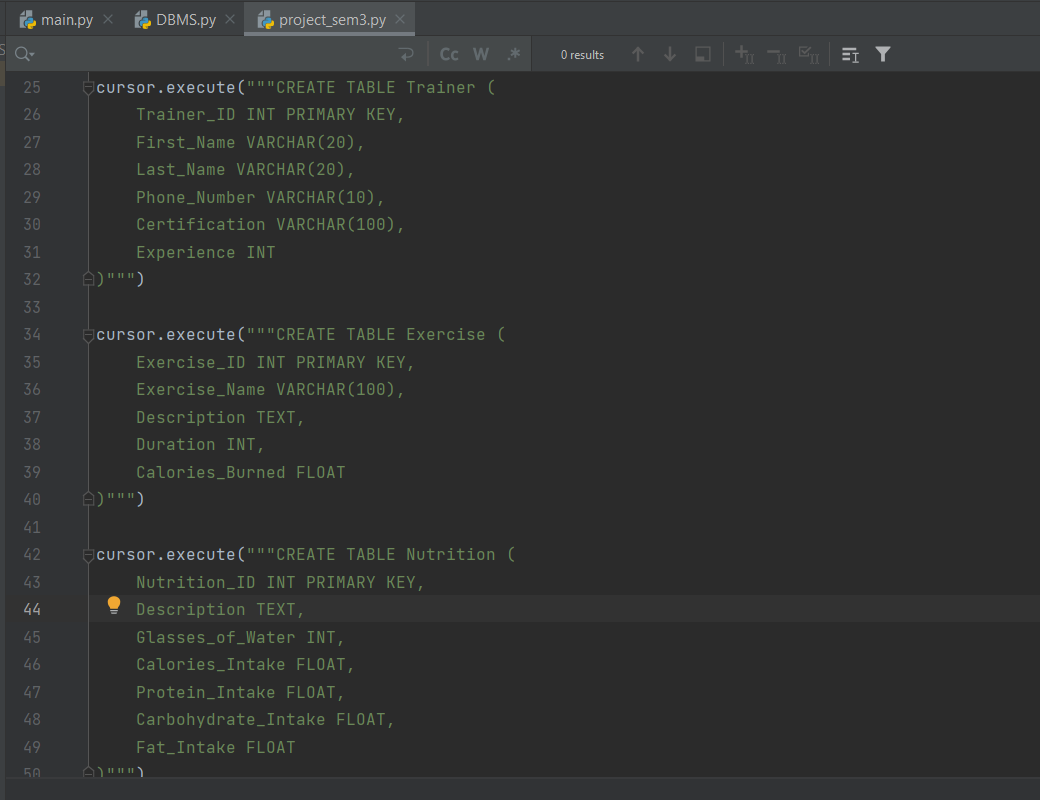
Trainer\_Nutrition\_Advices(Trainer\_ID, Nutrition\_ID)

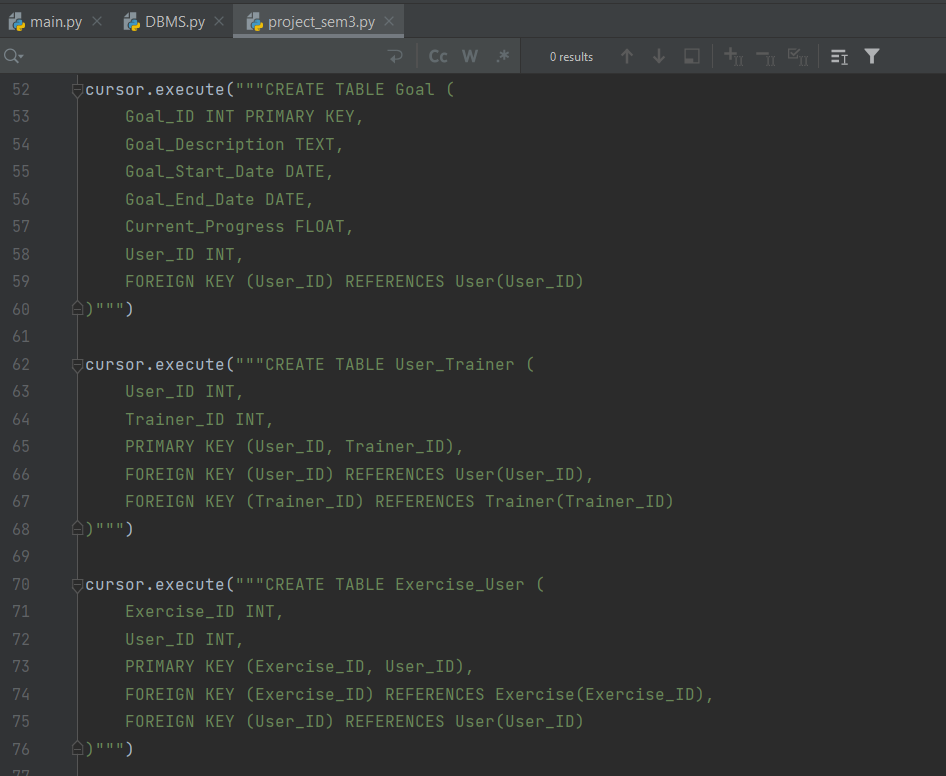
12. The Trainer\_Nutrition\_Advices relation is already in 3NF since each attribute is directly dependent on the primary key.

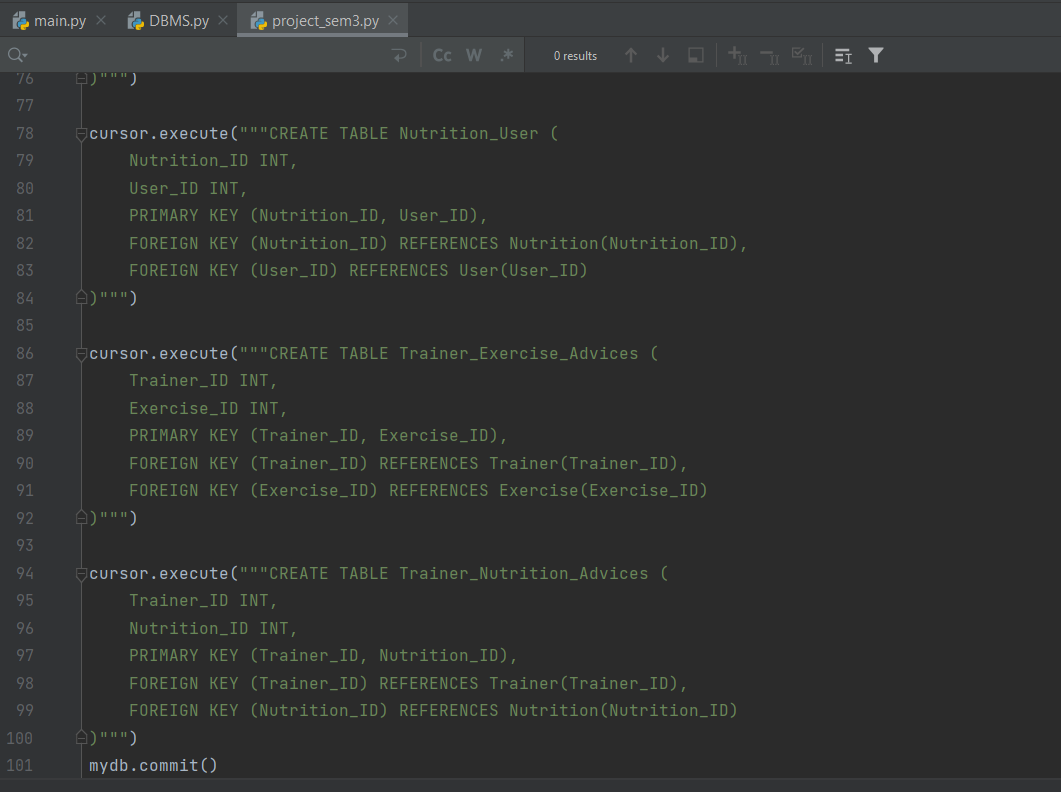
Therefore, all relations are now in 3NF.

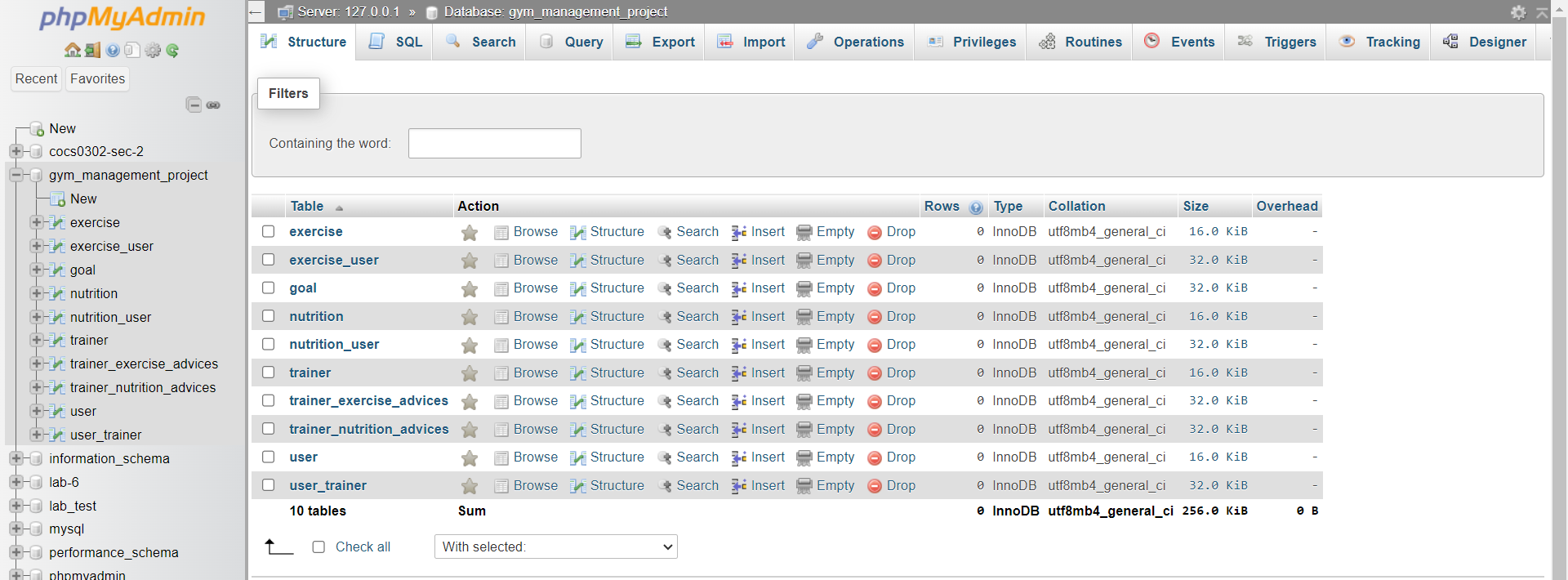
**TABLE CREATION (IN PYTHON)**



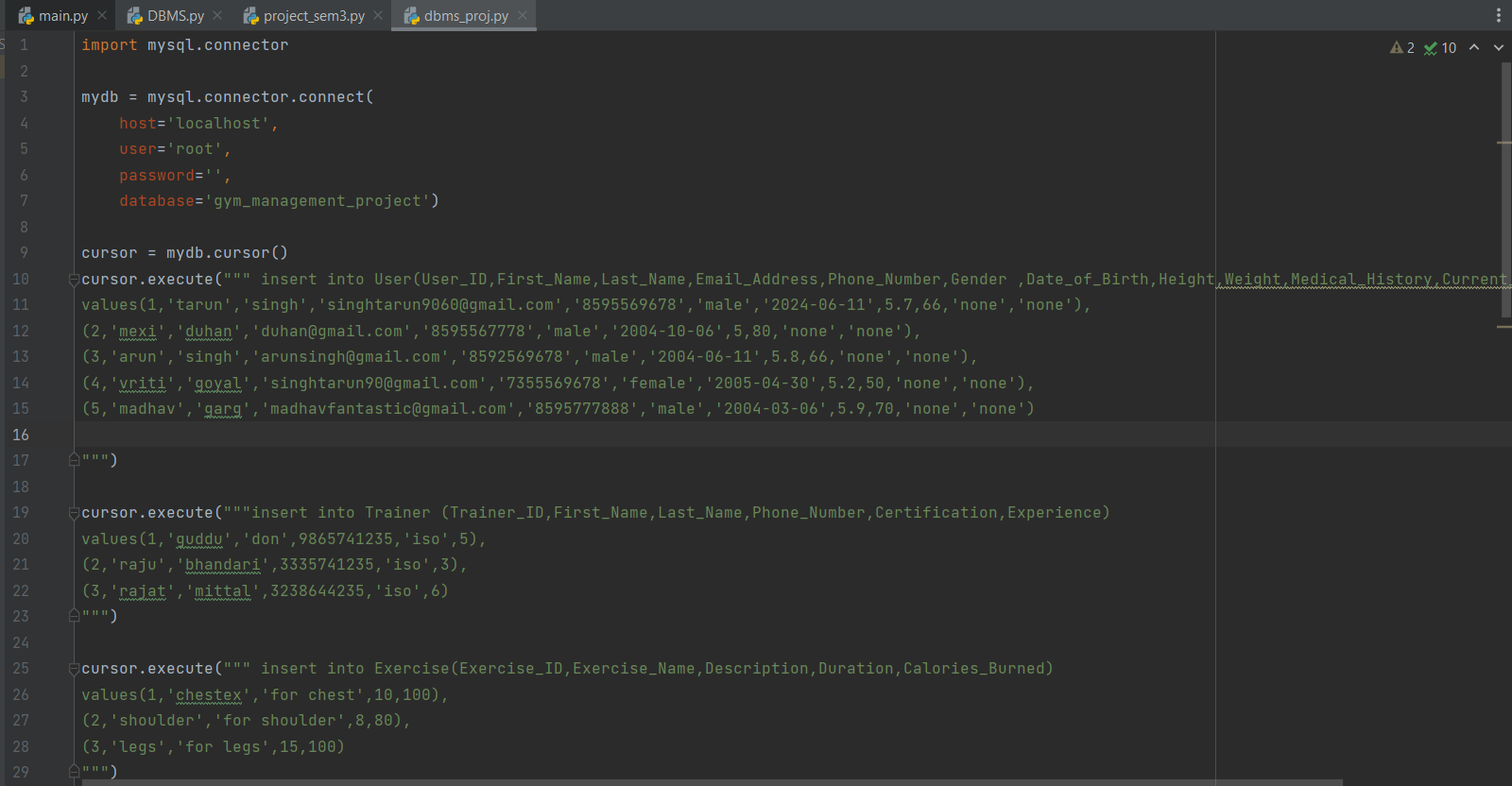


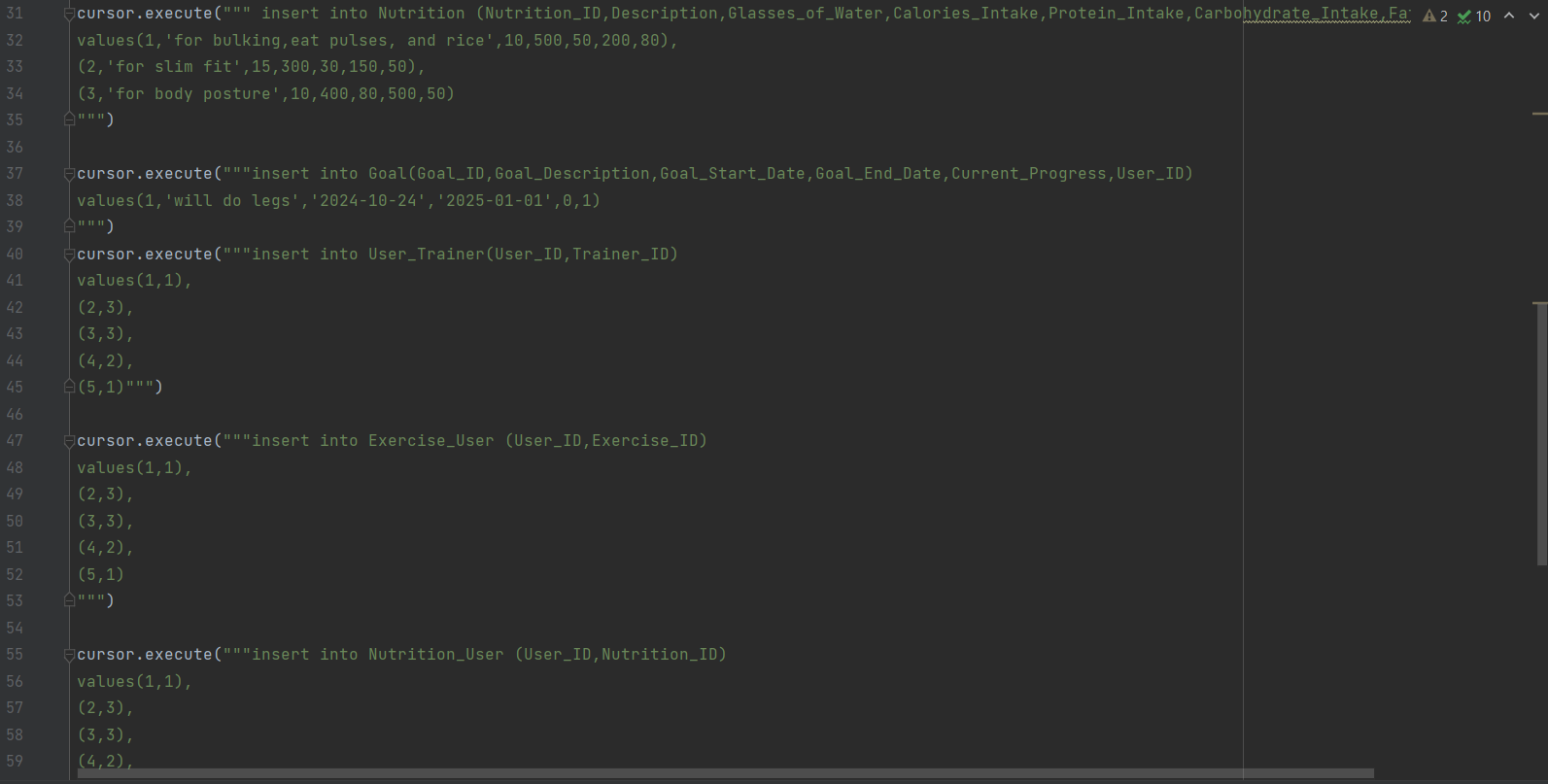


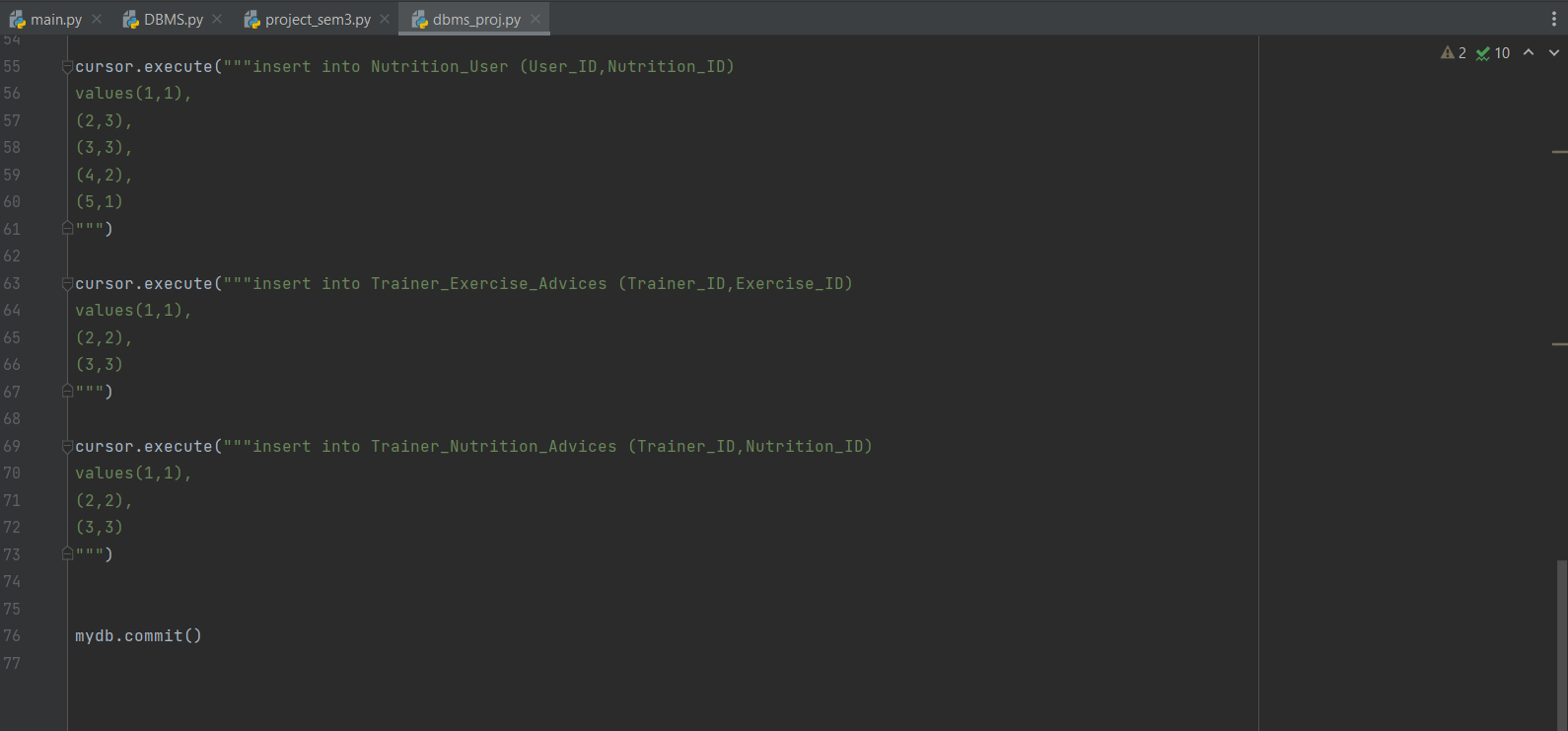




**DATA INSERTION**



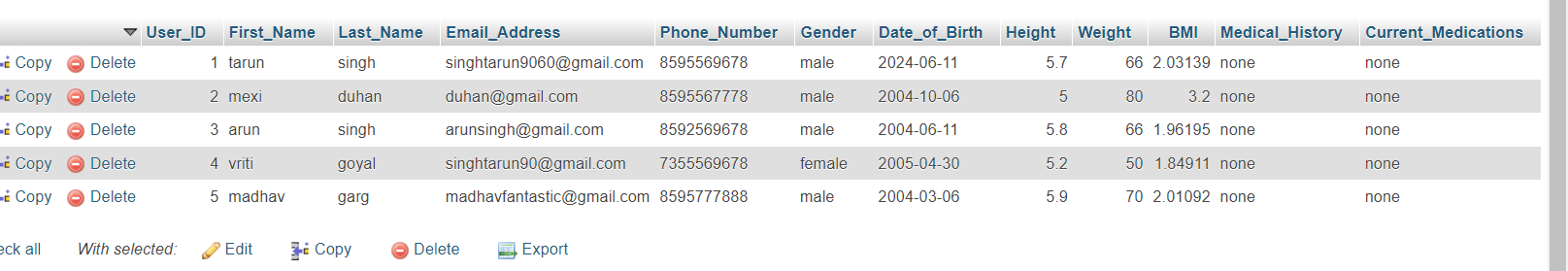




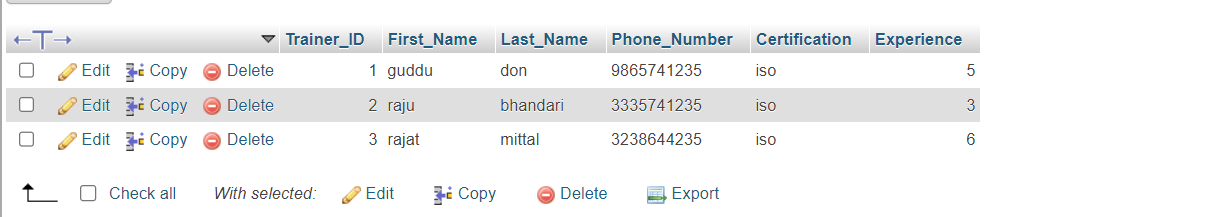
**QUERIES AND OUTPUT**

import mysql.connector  
mydb = mysql.connector.connect(  
 host='localhost',  
 user='root',  
 password='',  
 database='gym\_management\_project')  
cursor = mydb.cursor()

**#USER TABLE**cursor.execute(""" insert into User(User\_ID,First\_Name,Last\_Name,Email\_Address,Phone\_Number,Gender ,Date\_of\_Birth,Height,Weight,Medical\_History,Current\_Medications)  
values(1,'tarun','singh','singhtarun9060@gmail.com','8595569678','male','2024-06-11',5.7,66,'none','none'),  
(2,'mexi','duhan','duhan@gmail.com','8595567778','male','2004-10-06',5,80,'none','none'),  
(3,'arun','singh','arunsingh@gmail.com','8592569678','male','2004-06-11',5.8,66,'none','none'),  
(4,'vriti','goyal','singhtarun90@gmail.com','7355569678','female','2005-04-30',5.2,50,'none','none'),  
(5,'madhav','garg','madhavfantastic@gmail.com','8595777888','male','2004-03-06',5.9,70,'none','none')  
""")



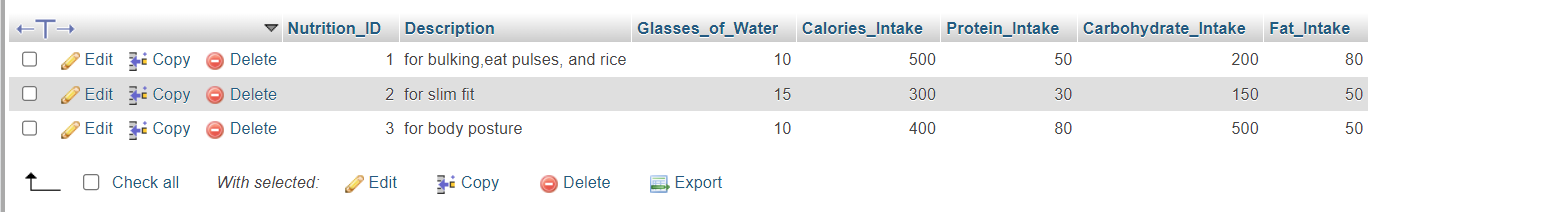
**#TRAINER TABLE**  
cursor.execute("""insert into Trainer (Trainer\_ID,First\_Name,Last\_Name,Phone\_Number,Certification,Experience)  
values(1,'guddu','don',9865741235,'iso',5),  
(2,'raju','bhandari',3335741235,'iso',3),  
(3,'rajat','mittal',3238644235,'iso',6)  
""")



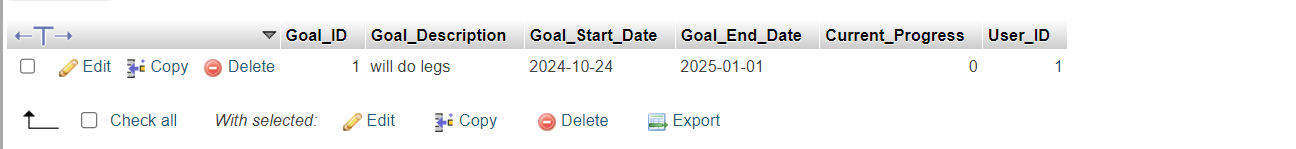
**#EXERCISE TABLE**cursor.execute(""" insert into Exercise(Exercise\_ID,Exercise\_Name,Description,Duration,Calories\_Burned)  
values(1,'chestex','for chest',10,100),  
(2,'shoulder','for shoulder',8,80),  
(3,'legs','for legs',15,100)  
""")



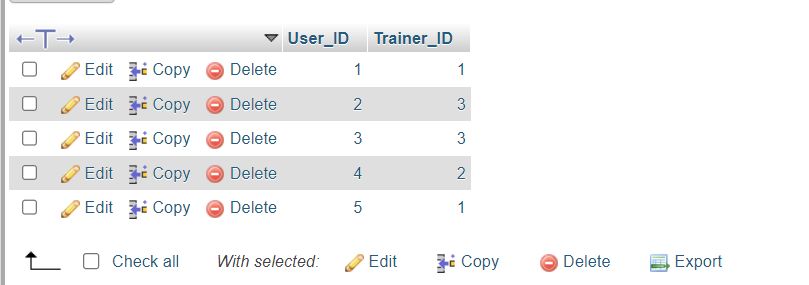
**#NUTRITION TABLE**cursor.execute(""" insert into Nutrition (Nutrition\_ID,Description,Glasses\_of\_Water,Calories\_Intake,Protein\_Intake,Carbohydrate\_Intake,Fat\_Intake)  
values(1,'for bulking,eat pulses, and rice',10,500,50,200,80),  
(2,'for slim fit',15,300,30,150,50),  
(3,'for body posture',10,400,80,500,50)  
""")



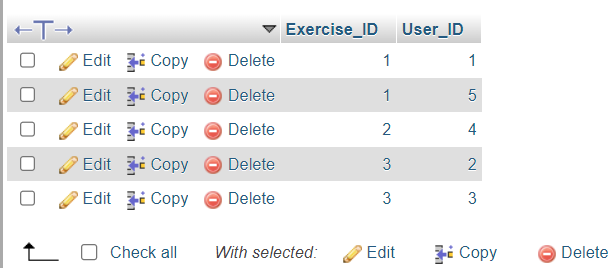
**#GOAL TABLE**  
cursor.execute("""insert into Goal(Goal\_ID,Goal\_Description,Goal\_Start\_Date,Goal\_End\_Date,Current\_Progress,User\_ID)  
values(1,'will do legs','2024-10-24','2025-01-01',0,1)  
""")



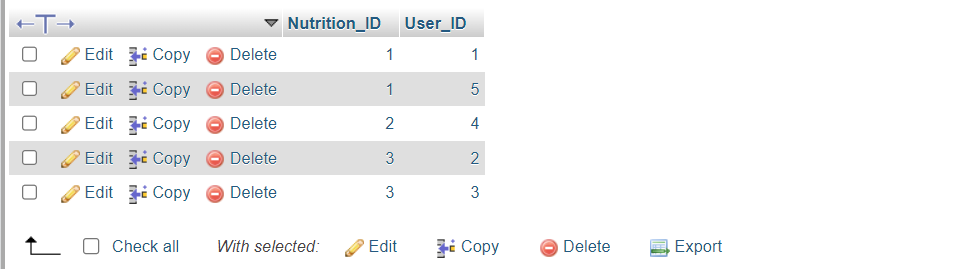
**#USER\_TRAINER TABLE**  
cursor.execute("""insert into User\_Trainer(User\_ID,Trainer\_ID)  
values(1,1),  
(2,3),  
(3,3),  
(4,2),  
(5,1)""")



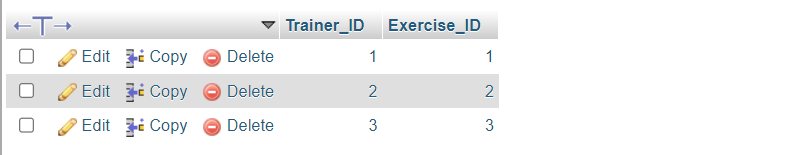
**#EXERCISE\_USER TABLE**  
cursor.execute("""insert into Exercise\_User (User\_ID,Exercise\_ID)  
values(1,1),  
(2,3),  
(3,3),  
(4,2),  
(5,1)  
""")



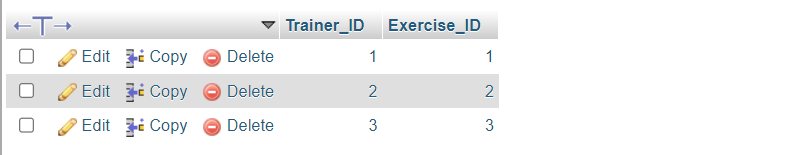
**#NUTRITION\_USER TABLE**cursor.execute("""insert into Nutrition\_User (User\_ID,Nutrition\_ID)  
values(1,1),  
(2,3),  
(3,3),  
(4,2),  
(5,1)  
""")



**#TRAINER\_EXERCISE\_ADVICES**  
cursor.execute("""insert into Trainer\_Exercise\_Advices (Trainer\_ID,Exercise\_ID)  
values(1,1),  
(2,2),  
(3,3)  
""")



**#TRAINER\_NUTRITION\_ADVICES**  
cursor.execute("""insert into Trainer\_Nutrition\_Advices (Trainer\_ID,Nutrition\_ID)  
values(1,1),  
(2,2),  
(3,3)  
""")



mydb.commit()