WORKOUT PLANNER AND TRACKER

Mini Project Report
Database Lab (DSE 2241)
Department of Data Science &
Computer Applications

B. Tech Data Science

4th Semester Batch: B3 Group:B3

Submitted By:

220968116	SAMYAK SINGH
220968118	AYUSHMAN RANJAN
220968120	PAREKH SHIVANSH HITESH
220968132	PRATHAM CHIRAG GHOSH
220968144	ABEER SETHIA

Mentored By

Vinayak M
Assistant Professor-Senior
DSCA, MIT

Archana H
Assistant Professor-Senior
DSCA, MIT

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Date: **24th February 2024**

CERTIFICATE

This is to certify that (220968116) SAMYAK SINGH, (220968118) AYUSHMAN RANJAN, (220968120) PAREKH SHIVANSH HITESH, (220968132) PRATHAM CHIRAG GHOSH, (220968144) ABEER SETHIA, have successfully executed a mini project titled “Workout Planner and Tracker” rightly bringing for the competencies and skill sets they have gained during the course- Database Lab (DSE 2241), thereby resulting in the culmination ofPPP this project.

Vinayak M	**Archana H**
Assistant Professor-Senior	**Assistant Professor-Senior**
DSCA, MIT	**DSCA, MIT**

ABSTRACT

In the dynamic landscape of health and wellness, the fusion of technology with fitness is pivotal. This project responds to the contemporary need for an intuitive Workout Planner and Tracker platform, empowering users to seamlessly plan and monitor their personalized fitness routines. Amidst the digital transformation era, where health consciousness is paramount, the demand for a comprehensive and accessible fitness tool has never been more pressing. The project's objective is to develop a robust system utilizing Oracle and SQL technologies, facilitating efficient workout planning and tracking.

The methodology involves meticulous design and implementation of a relational database using Oracle, adept at storing detailed exercise information, user profiles, and workout logs. SQL is employed for efficient data management and retrieval. The development process prioritizes creating an intuitive, user-friendly interface, ensuring users can effortlessly plan and track their fitness activities. The integration of Oracle and SQL establishes a solid foundation for seamless data interaction, benefiting both users and administrators.

The project yields a successful Workout Planner and Tracker platform, meeting criteria for user-friendliness, efficiency, and comprehensiveness. Users can plan workouts seamlessly, track progress, and access detailed exercise information. The significance lies in enhancing the user experience within the health and wellness domain, bridging the gap between technology and fitness. This project aligns with the escalating demand for advanced, accessible tools supporting individuals in their fitness journeys.

Emphasizing technology's role in elevating health and wellness experiences, this project signifies a leap forward in providing effective tools for managing and monitoring fitness activities. The successful integration of Oracle and SQL technologies, coupled with the development of a user-friendly interface, underscores progress in meeting the evolving needs of health-conscious individuals. Utilizing Oracle and SQL technologies for database management contributes to the efficiency and reliability of the Workout Planner and Tracker platform. This endeavor marks a milestone in reshaping fitness management paradigms.

Contents

Chapter 1
Introduction

In today's fast-paced world, health and fitness have become increasingly important aspects of our lives. With the rise of sedentary lifestyles and the prevalence of health issues related to physical inactivity, there is a growing need for tools that help individuals take charge of their fitness goals. Our project aims to address this need by developing a comprehensive Workout Planner and Tracker platform.

The Need for Change

Traditional methods of tracking workouts, such as pen and paper or basic spreadsheet applications, are often cumbersome and inefficient. They lack the functionality to provide personalized recommendations, track progress effectively, and adapt to changing fitness goals. Our project seeks to revolutionize this process by offering a modern, user-friendly solution that leverages the power of database technologies like Oracle and SQL.

Project Goals

Our primary goal is to create a platform that allows users to easily plan and monitor their workout routines. By providing a range of features such as exercise databases, customizable workout plans, progress tracking, and goal setting, we aim to empower users to take control of their fitness journey. Additionally, we want to ensure that the platform is intuitive and accessible to users of all fitness levels, from beginners to experienced athletes.

Advantages of the Work

Implementing this project will bring several advantages. Firstly, it will reduce the manual effort required to track workouts, making it easier for users to stay consistent with their fitness routines. Secondly, it will increase efficiency by providing tools for effective workout planning and monitoring. Lastly, by leveraging Oracle and SQL technologies, we can ensure robust data management and retrieval, enhancing the overall user experience.

In conclusion, our project seeks to combine technology with fitness to create a valuable tool for individuals looking to improve their health and wellness. By developing a user-friendly Workout Planner and Tracker platform, we aim to empower users to achieve their fitness goals and lead healthier lives..

Chapter 2
Synopsis

2.1 Proposed System

The proposed system aims to develop a comprehensive fitness planner equipped with a database to track user biodata and recommend exercises based on their data. The system addresses the challenge of personalizing fitness routines by utilizing individualized biodata to tailor exercise recommendations and track progress. Key components of the system include user profile management, biodata collection, exercise recommendation and progress tracking. By integrating these modules, the fitness planner offers a seamless user experience, facilitating efficient management of fitness goals and routines.

2.2 Objectives

The main objectives of the project are as follows:

- To develop a user-friendly interface for managing user profiles and inputting data.
- To create a database schema capable of storing diverse types of data.
- To track and display user progress over a period.
- To generate comprehensive visuals summarizing user fitness journey over time.
- To provide workout and diet options according to user level and goals.

Chapter 3
Functional Requirements

1. New User Registration:
   a. Input:
      i. Username, Password

   b. Processing:
      i. Check availability of username

   c. Output:
      i. User created successfully or Username already taken message

2. User Login:
   a. Input:
      i. username, Password

   b. Processing :
      i. Check the username and password against information stored in data storage

   c. Output:
      i. If user entered correct username & Password Login successful and open main application menu Else Display Login not successful, retry logging in

3. User Profile Management:
   a. Input:
      i. Personal Information (Age, Gender, Height, Weight, etc.)
      ii. Workout level (Beginner, Intermediate, Advanced)
      iii. Workout Goals

   b. Processing:
      i. Allows users to manage and update personal information and fitness goals.

   c. Output:
      i. Information updated successfully

4. Workout Filters:

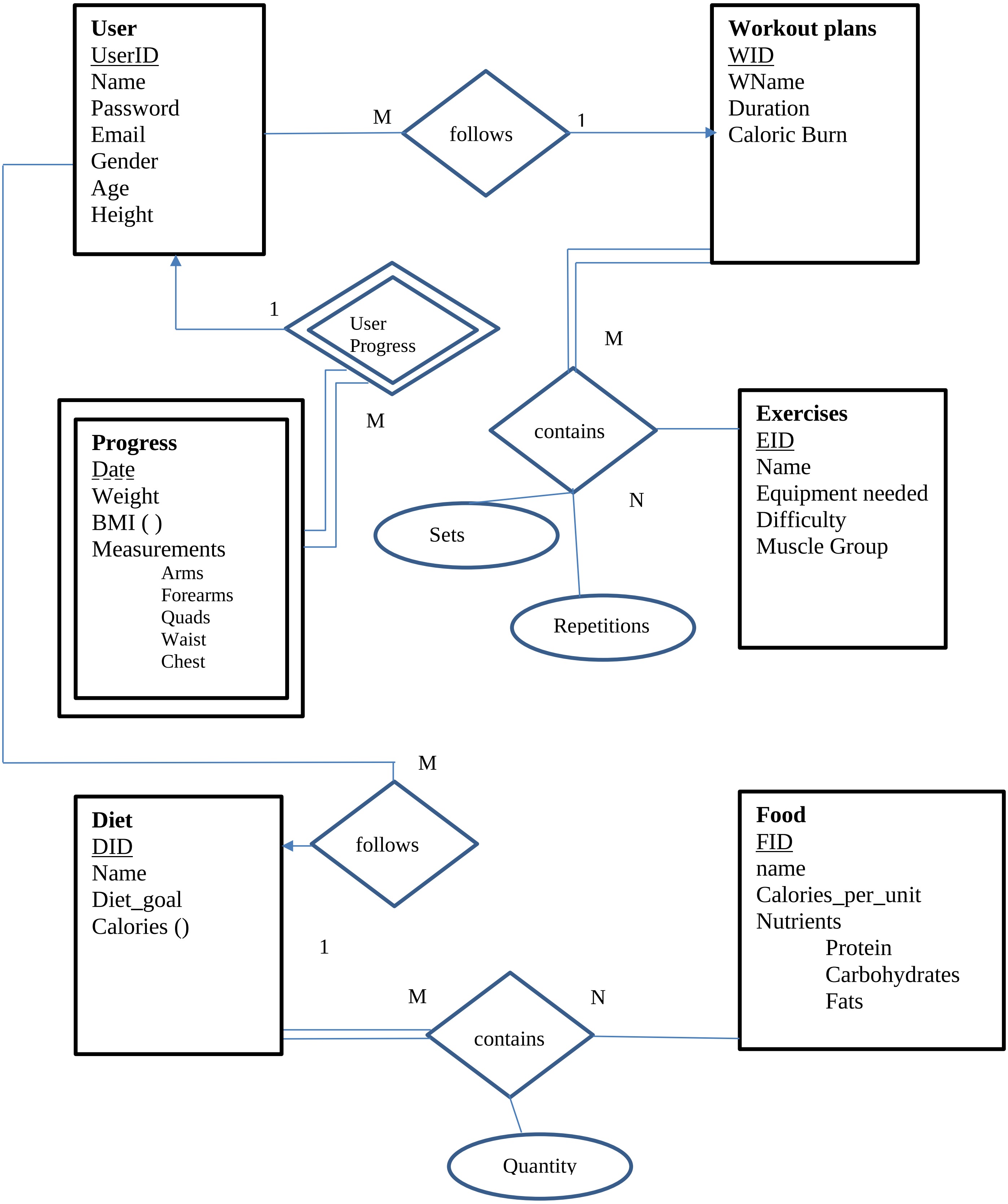
a. Input:
    i. (Filters) duration, difficulty levels etc.

b. Processing:
    i. Based on entered workout level, goals and other filters workouts are selected from workout table

c. Output:
    i. Appropriate Workouts are displayed

Chapter 4

4.1 ER Diagram

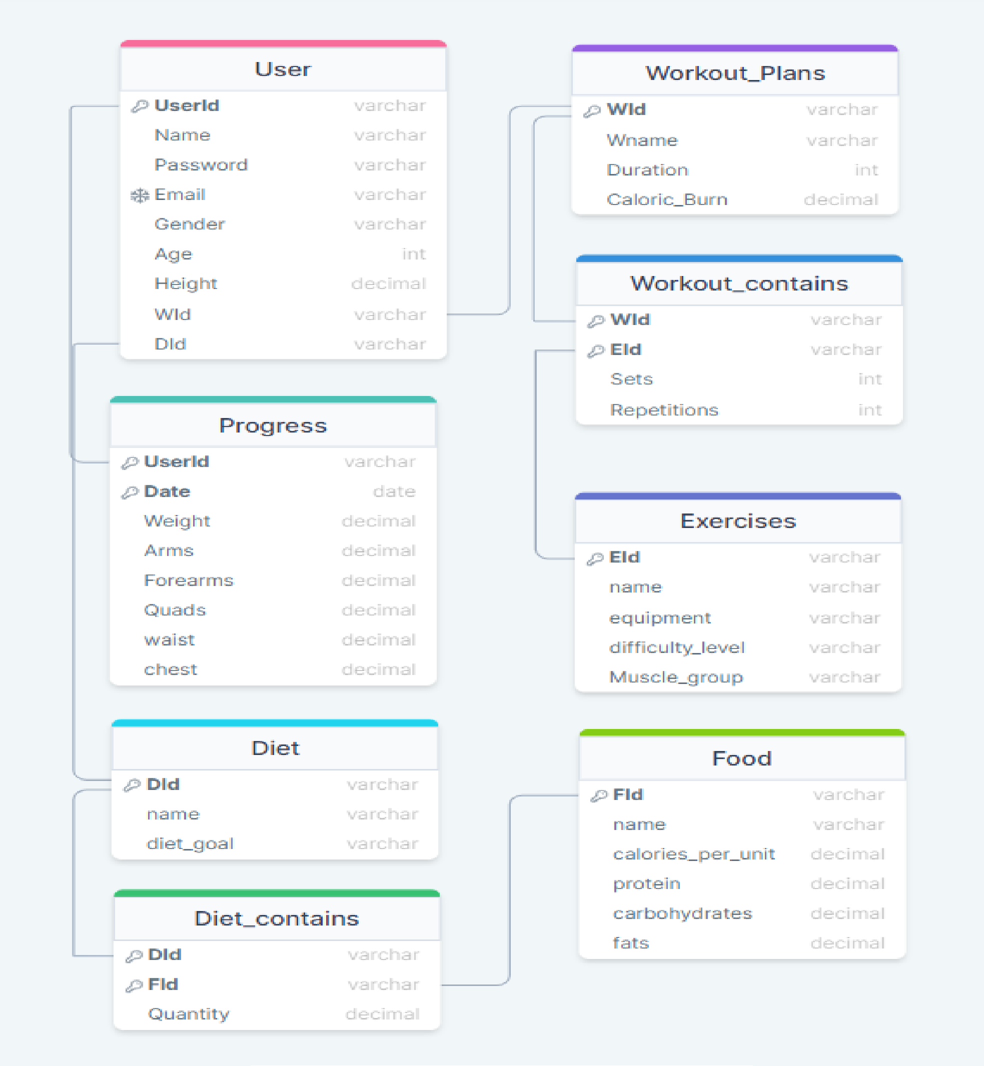


(1) ER Diagram

4.2 Schema Diagram

Relational Schema

- User (UserId, Name, Password, Email, Gender, Age, Height, WId, DId)
- Workout_plans (WId, WName, Duration, Caloric Burn)
- Exercises (EId, name, equipment, difficulty, muscle_group)
- Workout_contains (WId, EId, sets, repetitions)
- Progress (UserId, Date, weight, Arms, Forearms, Quads, waist, chest)
- Diet (Did, name, diet_goal)
- Food (foodId ,name ,calories_per_unit, protein, carbohydrates, fats)
- Diet_contains (DId, Fid, Quantity)



(2) Schema Diagram

4.3 Data Dictionary

USER:

Column Name	**DataType**	**Constraint**	**Constraint Name**
userid	Varchar	primary key starts wih U	Pk_user Startswith_U
name	Varchar		
password	Varchar	Length > 4	Pwd_len
email	Varchar	Unique	Unq_email
gender	Varchar	In ‘M’, ‘F’	Chk_gender
age	Number		
height	Number		
wid	Varchar	References workout_plans	Fk_workout_plans
did	Varchar	References diet	Fk_diet

WORKOUT_PLANS:

Column Name	**DataType**	**Constraint**	**Constraint Name**
wid	Varchar	primary key starts wih W	Pk_workout_plans Startswith_W
Wname	Varchar		
duration	Number		
Caloric_burn	Number		

WORKOUT_CONTAINS:

Column Name	**DataType**	**Constraint**	**Constraint Name**
Wid	Varchar	Primary key references workout_plans	Pk_workout_contains Fk_contains_workout
Eid	Varchar	Primary key References Exercise	Pk_workout_contains Fk_contains_exercise
duration	Number		
Caloric_burn	Number		

EXERCISES:

Column Name	DataType	Constraint	Constraint Name
Eid	Varchar	Primary key Starts with E	Pk_exercise Startswith_E
name	Varchar		
Equipment_needed	Varchar		
Difficulty_level	Varchar	In easy, medium ,hard	Chk_difficulty
Muscle_group	Varchar	In chest , biceps , triceps , quads , lats , upper back , lower back , calves , hamstrings , forearms , traps , abs , shoulder	Chk_muscles

PROGRESS:

Column Name	DataType	Constraint	Constraint Name
Userid	Varchar	Primary key references user	Pk_progress Fk_progress_user
Date	Date	Primary key	Pk_progress
Weight	Number		
Arms	Number		
Forearms	Number		
Quads	Number		
Waist	Number		
Chest	Number		

DIET:

Column Name	DataType	Constraint	Constraint Name
Did	Varchar	Primary key Starts with D	Pk_diet Startswith_D
name	Varchar		
Diet_goal	Varchar	in maintain , loose , gain	

FOOD:

Column Name	DataType	Constraint	Constraint Name
Fid	Varchar	Primary key Starts with F	Pk_food Startswith_F
Name	Varchar		
Calories_per_unit	Number		
protein	Number		
carbohydrates	Number		
fats	Number		

DIET CONTAINS:

Column Name	DataType	Constraint	Constraint Name
Did	Varchar	Primary key References diet	Pk_diet_contains Fk_contains_diet
Fid	Varchar	Primary key References food	Pk_diet_contains Fk_contains_food
Quantity	Number		

4.4 Relational Model Implementation

```
CREATE TABLE workout_plans (
   wid VARCHAR(20) PRIMARY KEY,
   wname VARCHAR(100),
   duration INT,
   caloric_burn INT,
   difficulty VARCHAR(100),
   CONSTRAINT Startswith_W CHECK (wid LIKE 'W%')
);

CREATE TABLE exercises (
   eid VARCHAR(20) PRIMARY KEY,
   name VARCHAR(100),
   equipment_needed VARCHAR(100),
   difficulty_level VARCHAR(10) CHECK (difficulty_level IN ('easy', 'medium', 'hard')),
   muscle_group VARCHAR(20) CHECK (muscle_group IN ('chest', 'biceps', 'triceps', 'quads', 
'lats', 'upper back', 'lower back', 'calves', 'hamstrings', 'forearms', 'traps', 'abs', 'shoulder')),
   CONSTRAINT Startswith_E CHECK (eid LIKE 'E%')
);

CREATE TABLE workout_contains (
   wid VARCHAR(20),
   eid VARCHAR(20),
   duration INT,
   caloric_burn INT,
   PRIMARY KEY (wid, eid),
   FOREIGN KEY (wid) REFERENCES workout_plans(wid),
   FOREIGN KEY (eid) REFERENCES exercises(eid)
);

CREATE TABLE diet (
   did VARCHAR(20) PRIMARY KEY,
   name VARCHAR(100),
   diet_goal VARCHAR(20) CHECK (diet_goal IN ('maintain', 'loose', 'gain')),
   CONSTRAINT Startswith_D CHECK (did LIKE 'D%')
);


CREATE TABLE food (
   fid VARCHAR(20) PRIMARY KEY,
   name VARCHAR(100),
   calories_per_unit INT,
   protein DECIMAL(5,2),
   carbohydrates DECIMAL(5,2),
   fats DECIMAL(5,2),
   CONSTRAINT Startswith_F CHECK (fid LIKE 'F%')
);

CREATE TABLE diet_contains (
   did VARCHAR(20),
   fid VARCHAR(20),
   quantity INT,
   PRIMARY KEY (did, fid),
   FOREIGN KEY (did) REFERENCES diet(did),
   FOREIGN KEY (fid) REFERENCES food(fid)
);

CREATE TABLE users (
   userid VARCHAR(20) PRIMARY KEY,
   name VARCHAR(100),
   password VARCHAR(100),
   email VARCHAR(100) UNIQUE,
   gender CHAR(1) CHECK (gender IN ('M', 'F')),
   age INT,
   height DECIMAL(5,2),
   wid VARCHAR(20),
   did VARCHAR(20),
   CONSTRAINT Startswith_U CHECK (userid LIKE 'U%'),
   CONSTRAINT Pwd_len CHECK (LENGTH(password) > 4),
   FOREIGN KEY (wid) REFERENCES workout_plans(wid),
   FOREIGN KEY (did) REFERENCES diet(did)
);

CREATE TABLE progress (
   userid VARCHAR(20),
   date DATE,
   weight DECIMAL(5,2),
   arms DECIMAL(5,2),
   forearms DECIMAL(5,2),
   quads DECIMAL(5,2),
   waist DECIMAL(5,2),
   chest DECIMAL(5,2),
   PRIMARY KEY (userid, date),
   FOREIGN KEY (userid) REFERENCES users(userid));
```

Chapter 5
Implementation

5.1 Queries

- To get the exercises present in a given workout :

```
SELECT e.name
FROM workout_contains wc
JOIN exercises e ON wc.eid = e.eid
WHERE wc.wid = 'W010';
```

- To get food items present in a diet :

```
SELECT f.name
FROM food f
JOIN diet_contains dc ON f.fid = dc.fid
WHERE dc.did = 'D010';
```

- To retrieve workouts followed by each user :

```
SELECT u.name, w.wname
FROM workout_plans w
JOIN users u ON w.wid = u.wid;
```

- To retrieve diet followed by each user :

```
SELECT u.name, d.name
FROM diet d
JOIN users u ON d.did = u.did;
```

- Filtering exercises by difficulty level :

```
SELECT name
FROM exercises
WHERE difficulty_level = 'easy';
```

- Filtering diet by diet goal:

```
SELECT name
FROM diet
WHERE diet_goal = 'gain';
```

- To retrieve frequency of exercises across workouts:

```
SELECT e.name, COUNT(e.name)
FROM exercises e
JOIN workout_contains wc ON wc.eid = e.eid
GROUP BY e.name;
```

- Exercises followed by user consuming a particular food item:

```
SELECT e.name
FROM exercises e
JOIN workout_contains wc ON wc.eid = e.eid
JOIN users u ON u.wid = wc.wid
JOIN diet_contains dc ON dc.did = u.did
JOIN food f ON f.fid = dc.fid
WHERE f.name = 'Salmon';
```

- To retrieve frequency of food items across diets:

```
SELECT f.name, COUNT(f.name)
FROM food f
JOIN diet_contains dc ON dc.fid = f.fid
GROUP BY f.name;
```

- To select frequency of workouts being followed by different users:

```
SELECT wname, COUNT(wname)
FROM workout_plans
NATURAL JOIN users
GROUP BY wname;
```

- Frequency of diets being followed by different users:

```
SELECT d.name, COUNT(d.name)
FROM diet d
JOIN users u ON d.did = u.did
GROUP BY d.name;
```

5.2 Stored Procedures

1> To add new user record to users table when a new user signs up:

```
DELIMITER //
CREATE PROCEDURE new_user(
   IN uname VARCHAR(100),      -- Input parameter for user's name
   IN upassword VARCHAR(100),  -- Input parameter for user's password
   IN uemail VARCHAR(100),     -- Input parameter for user's email
   IN ugender VARCHAR(1),      -- Input parameter for user's gender
   IN uage INT,                -- Input parameter for user's age
   IN uheight DECIMAL(5,2)     -- Input parameter for user's height
)
BEGIN
   DECLARE cnt INT;            -- Variable to store count of existing users
   DECLARE userid VARCHAR(20); -- Variable to store the generated user ID

   -- Count the number of existing users
   SELECT COUNT(*) INTO cnt FROM users;

   -- Generate a unique user ID based on the count of existing users
   SET userid = CONCAT('U', LPAD(CONVERT(cnt, CHAR), 3, '0'));

   -- Insert the new user record into the users table
   INSERT INTO users (userid, name, password, email, gender, age, height)
   VALUES (userid, uname, upassword, uemail, ugender, uage, uheight);
END //
DELIMITER ;
```

2> To add new progress record to progress table when input by user:

```
DELIMITER //
CREATE PROCEDURE new_progress(
   IN userid VARCHAR(100),  -- Input parameter for user ID
   IN weight DECIMAL(5,2),  -- Input parameter for weight
   IN arms DECIMAL(5,2),    -- Input parameter for arm measurement
   IN forearms DECIMAL(5,2),-- Input parameter for forearm measurement
   IN quads DECIMAL(5,2),   -- Input parameter for quadriceps measurement
   IN waist DECIMAL(5,2),   -- Input parameter for waist measurement
   IN chest DECIMAL(5,2)    -- Input parameter for chest measurement
)
BEGIN
   -- Insert new progress record into the progress table
   INSERT INTO progress
   VALUES (userid, CURDATE(), weight, arms, forearms, quads, waist, chest);
END //
DELIMITER ;
```

3> To check if calculated BMI is in normal range:

```
DELIMITER //
CREATE PROCEDURE BMI_range(
   IN BMI DECIMAL(5,2)  -- Input parameter for BMI
)
BEGIN
   -- Check if BMI is underweight, overweight, or obese
   IF BMI < 18.5 THEN
      SELECT 'underweight';
   ELSEIF BMI > 30 THEN
      SELECT 'obese';
   ELSEIF BMI > 24.9 THEN
      SELECT 'overweight';
   END IF;
END //
DELIMITER ;
```

5.3 Stored Functions

1> To calculate total calories in diet:

```
DELIMITER &&
CREATE FUNCTION total_calories(v_did VARCHAR(255)) RETURNS
DECIMAL DETERMINISTIC
BEGIN
   DECLARE total_cal DECIMAL DEFAULT 0;

   DECLARE done INT DEFAULT FALSE;
   DECLARE cur_calories_per_unit DECIMAL;

   -- Cursor to iterate over food items in the diet
   DECLARE cur CURSOR FOR
      SELECT f.calories_per_unit
      FROM food f
      JOIN diet_contains dc ON dc.fid = f.fid
      WHERE dc.did = v_did;

   -- Handler for when no more rows are found
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

   OPEN cur;
   read_loop: LOOP
      FETCH cur INTO cur_calories_per_unit;
      IF done THEN
         LEAVE read_loop;
      END IF;

      -- Calculate total calories
      SET total_cal = total_cal + cur_calories_per_unit;
   END LOOP;
   CLOSE cur;

   RETURN total_cal;
END &&
DELIMITER ;
```

2> To calculate current BMI from height and weight information:

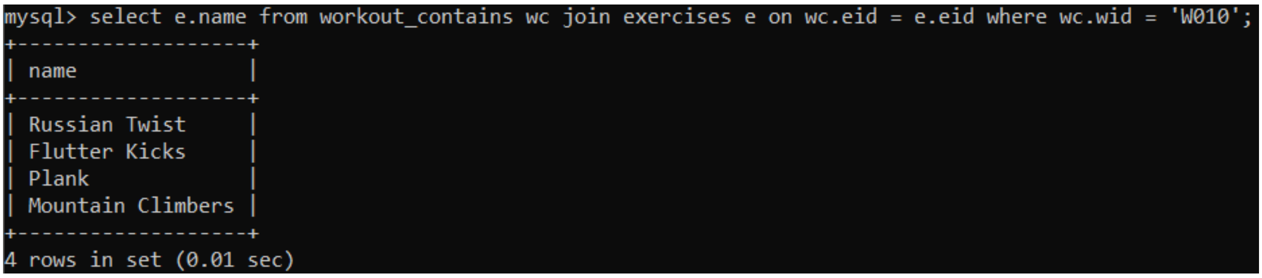
```
DELIMITER &&
CREATE FUNCTION bmi(v_weight DECIMAL , v_height DECIMAL)
RETURNS DECIMAL DETERMINISTIC
BEGIN
   -- Calculate BMI using weight (in kg) and height (in cm)
   RETURN v_weight /((v_height / 100) * (v_height / 100));
END &&
DELIMITER ;
```

3>To get workout difficulty level based on the exercises it contains:

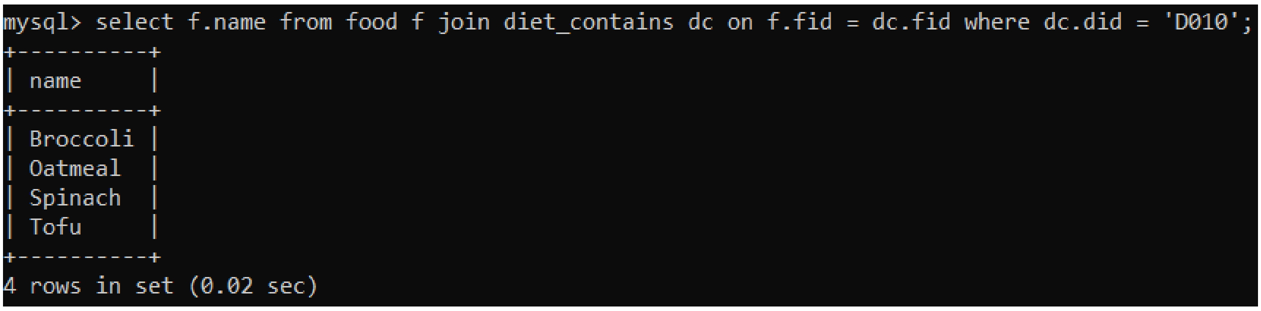
```
DELIMITER &&
CREATE FUNCTION workout_difficulty(inp_wid VARCHAR(20)) RETURNS
VARCHAR(20) DETERMINISTIC
BEGIN
   DECLARE diff VARCHAR(100);
   DECLARE ec INT;
   DECLARE mc INT;
   DECLARE hc INT;
   DECLARE m INT;

   -- Count exercises of each difficulty level in the workout
   SELECT COUNT(*) INTO ec FROM exercises e JOIN workout_contains wc
ON e.eid = wc.eid WHERE wc.wid = inp_wid AND e.difficulty_level = 'easy';
   SELECT COUNT(*) INTO mc FROM exercises e JOIN workout_contains wc
ON e.eid = wc.eid WHERE wc.wid = inp_wid AND e.difficulty_level = 'medium';
   SELECT COUNT(*) INTO hc FROM exercises e JOIN workout_contains wc
ON e.eid = wc.eid WHERE wc.wid = inp_wid AND e.difficulty_level = 'hard';

   -- Find the maximum count to determine the workout difficulty
   SET m = GREATEST(ec, mc, hc);

   -- Return the difficulty level
   IF m = ec THEN
      RETURN 'easy';
   ELSEIF m = mc THEN
      RETURN 'medium';
   ELSEIF m = hc THEN
      RETURN 'hard';
   END IF;
END &&
DELIMITER ;
```

Chapter 6
Result

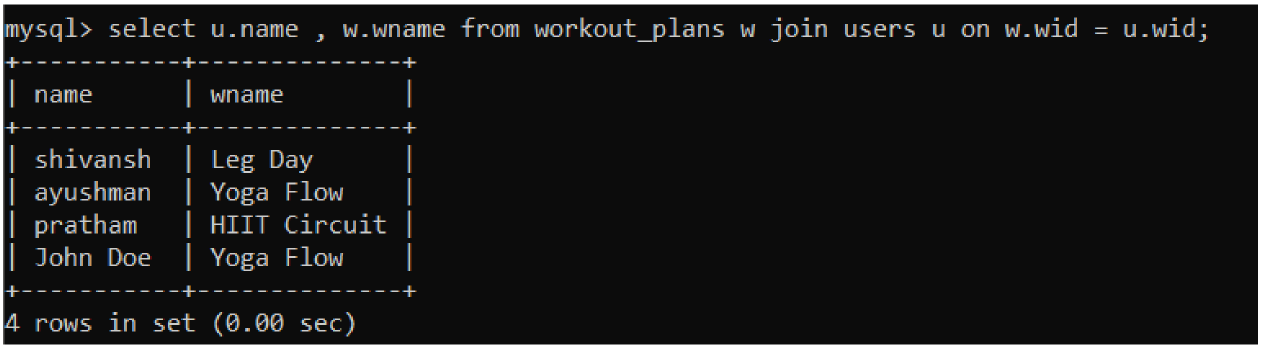
Queries



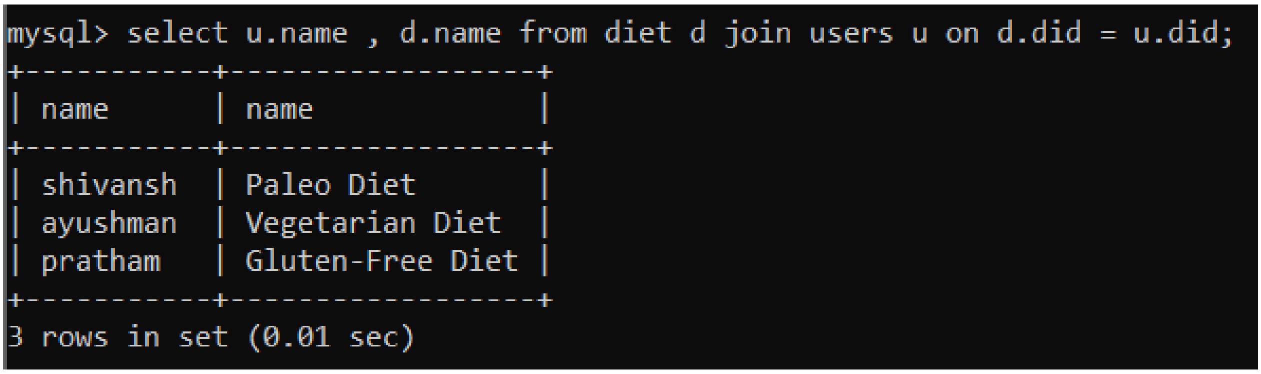
```
mysql> select e.name from workout_contains wc join exercises e on wc.eid = e.eid where wc.wid = 'W010';
+-------------------+
| name              |
+-------------------+
| Russian Twist     |
| Flutter Kicks     |
| Plank             |
| Mountain Climbers |
+-------------------+
4 rows in set (0.01 sec)
```



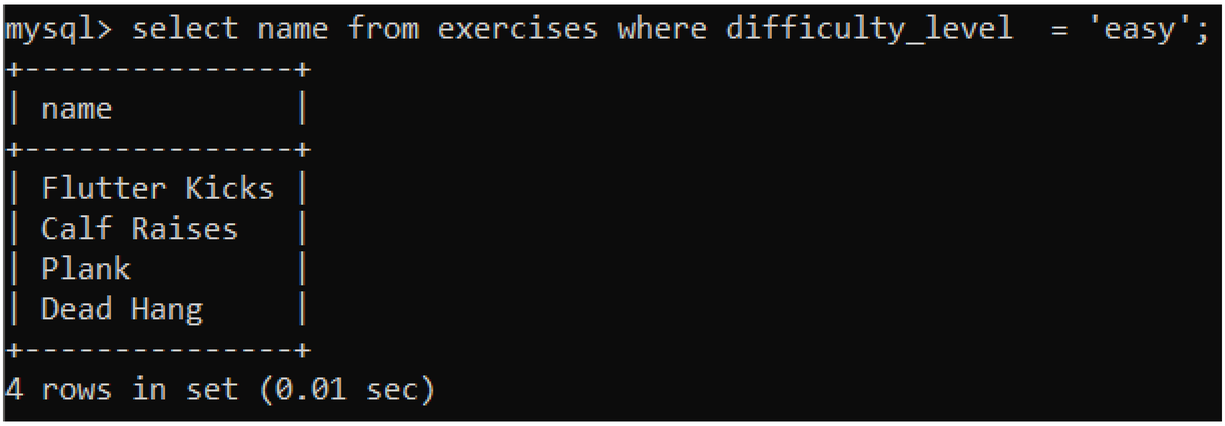
```
mysql> select f.name from food f join diet_contains dc on f.fid = dc.fid where dc.did = 'D010';
+----------+
| name     |
+----------+
| Broccoli |
| Oatmeal  |
| Spinach  |
| Tofu     |
+----------+
4 rows in set (0.02 sec)
```



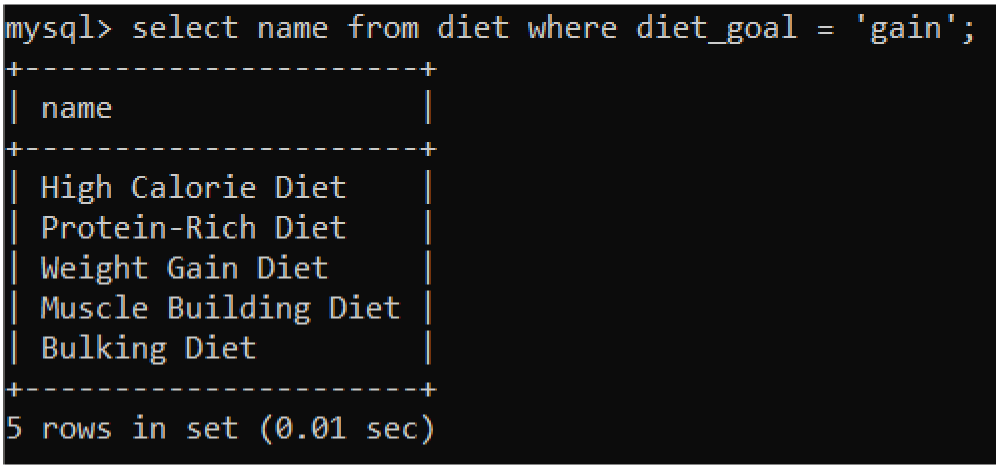
```
mysql> select u.name , w.wname from workout_plans w join users u on w.wid = u.wid;
+----------+--------------+
| name     | wname        |
+----------+--------------+
| shivansh | Leg Day      |
| ayushman | Yoga Flow    |
| pratham  | HIIT Circuit |
| John Doe | Yoga Flow    |
+----------+--------------+
4 rows in set (0.00 sec)
```



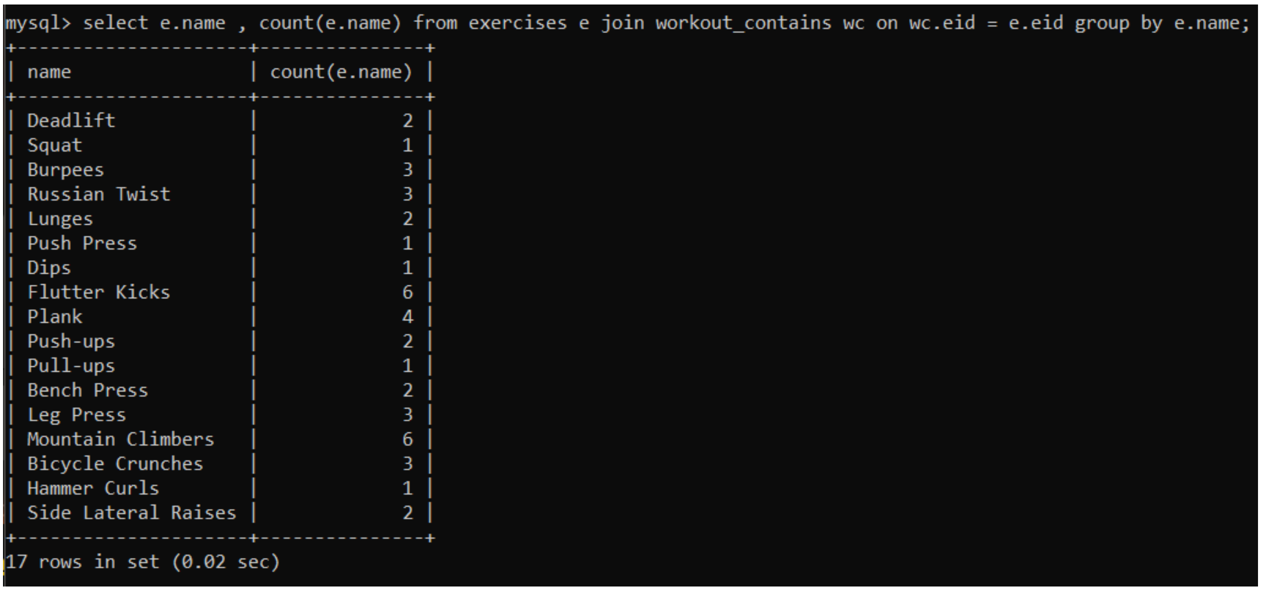
```
mysql> select u.name , d.name from diet d join users u on d.did = u.did;
+----------+------------------+
| name     | name             |
+----------+------------------+
| shivansh | Paleo Diet       |
| ayushman | Vegetarian Diet  |
| pratham  | Gluten-Free Diet |
+----------+------------------+
3 rows in set (0.01 sec)
```



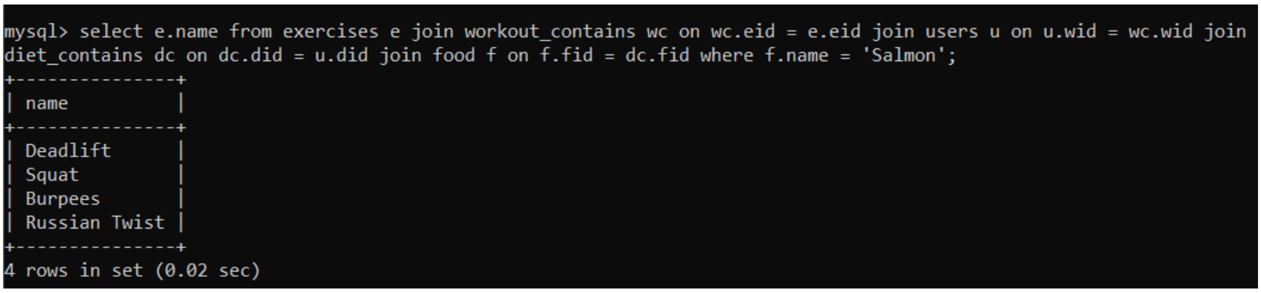
```
mysql> select name from exercises where difficulty_level  = 'easy';
+---------------+
| name          |
+---------------+
| Flutter Kicks |
| Calf Raises   |
| Plank         |
| Dead Hang     |
+---------------+
4 rows in set (0.01 sec)
```



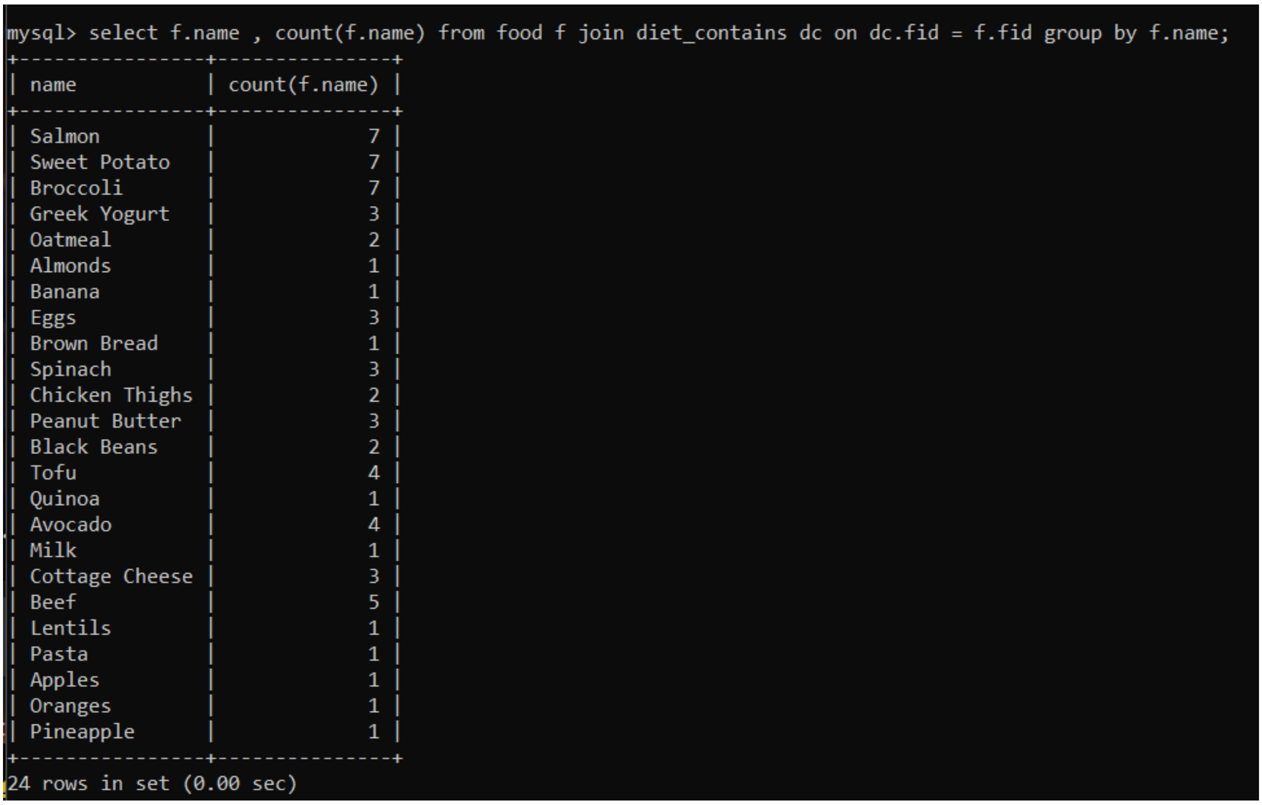
```
mysql> select name from diet where diet_goal = 'gain';
+----------------------+
| name                 |
+----------------------+
| High Calorie Diet    |
| Protein-Rich Diet    |
| Weight Gain Diet     |
| Muscle Building Diet |
| Bulking Diet         |
+----------------------+
5 rows in set (0.01 sec)
```



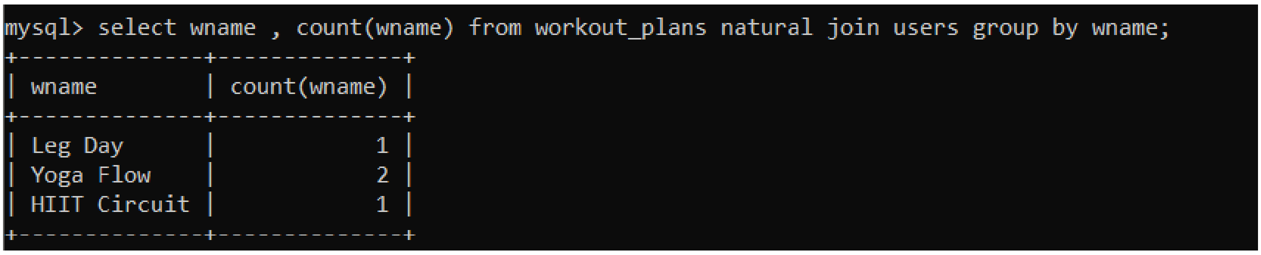
```
mysql> select e.name , count(e.name) from exercises e join workout_contains wc on wc.eid = e.eid group by e.name;
+---------------------+---------------+
| name                | count(e.name) |
+---------------------+---------------+
| Deadlift            |             2 |
| Squat               |             1 |
| Burpees             |             3 |
| Russian Twist       |             3 |
| Lunges              |             2 |
| Push Press          |             1 |
| Dips                |             1 |
| Flutter Kicks       |             6 |
| Plank               |             4 |
| Push-ups            |             2 |
| Pull-ups            |             1 |
| Bench Press         |             2 |
| Leg Press           |             3 |
| Mountain Climbers   |             6 |
| Bicycle Crunches    |             3 |
| Hammer Curls        |             1 |
| Side Lateral Raises |             2 |
+---------------------+---------------+
17 rows in set (0.02 sec)
```



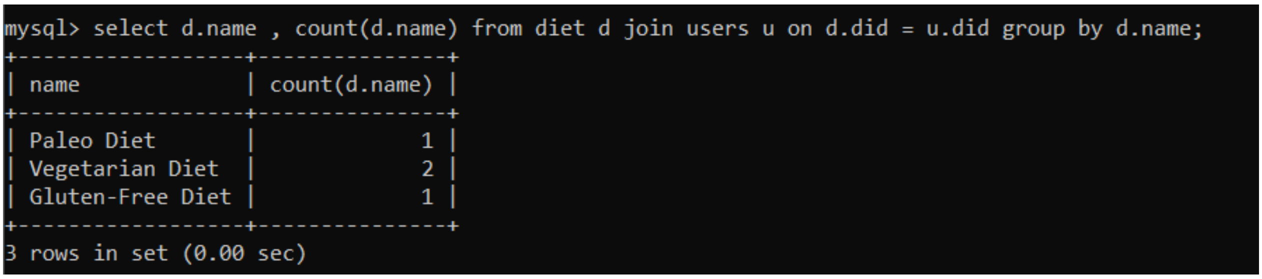
```
mysql> select e.name from exercises e join workout_contains wc on wc.eid = e.eid join users u on u.wid = wc.wid join
diet_contains dc on dc.did = u.did join food f on f.fid = dc.fid where f.name = 'Salmon';
+---------------+
| name          |
+---------------+
| Deadlift      |
| Squat         |
| Burpees       |
| Russian Twist |
+---------------+
4 rows in set (0.02 sec)
```



```
mysql> select f.name , count(f.name) from food f join diet_contains dc on dc.fid = f.fid group by f.name;
+----------------+---------------+
| name           | count(f.name) |
+----------------+---------------+
| Salmon         |             7 |
| Sweet Potato   |             7 |
| Broccoli       |             7 |
| Greek Yogurt   |             3 |
| Oatmeal        |             2 |
| Almonds        |             1 |
| Banana         |             1 |
| Eggs           |             3 |
| Brown Bread    |             1 |
| Spinach        |             3 |
| Chicken Thighs |             2 |
| Peanut Butter  |             3 |
| Black Beans    |             2 |
| Tofu           |             4 |
| Quinoa         |             1 |
| Avocado        |             4 |
| Milk           |             1 |
| Cottage Cheese |             3 |
| Beef           |             5 |
| Lentils        |             1 |
| Pasta          |             1 |
| Apples         |             1 |
| Oranges        |             1 |
| Pineapple      |             1 |
+----------------+---------------+
24 rows in set (0.00 sec)
```

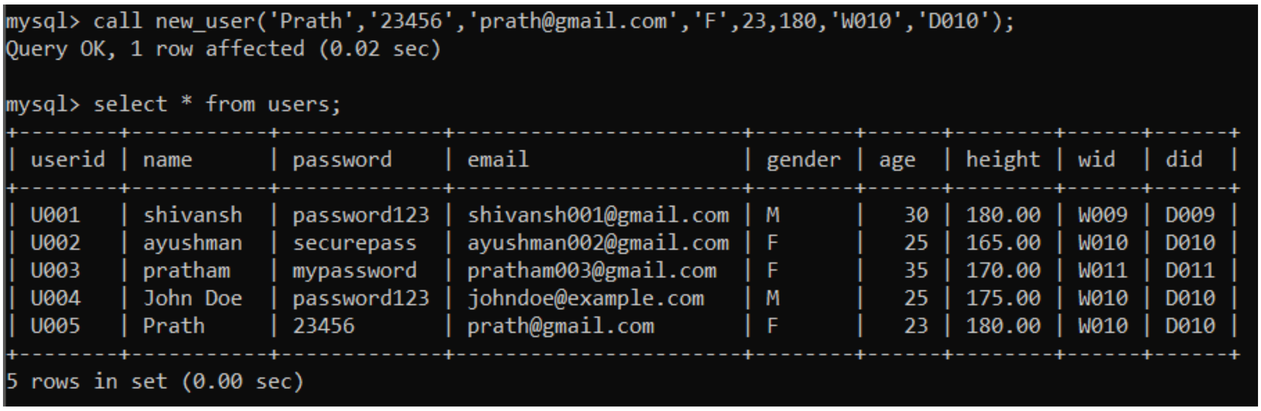


```
mysql> select wname , count(wname) from workout_plans natural join users group by wname;
+--------------+--------------+
| wname        | count(wname) |
+--------------+--------------+
| Leg Day      |            1 |
| Yoga Flow    |            2 |
| HIIT Circuit |            1 |
+--------------+--------------+
```

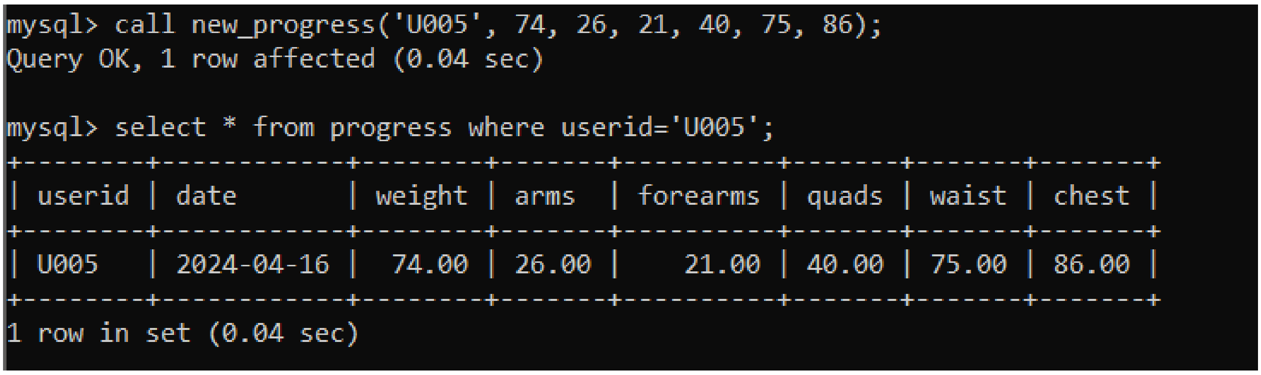


```
mysql> select d.name , count(d.name) from diet d join users u on d.did = u.did group by d.name;
+------------------+---------------+
| name             | count(d.name) |
+------------------+---------------+
| Paleo Diet       |             1 |
| Vegetarian Diet  |             2 |
| Gluten-Free Diet |             1 |
+------------------+---------------+
3 rows in set (0.00 sec)
```

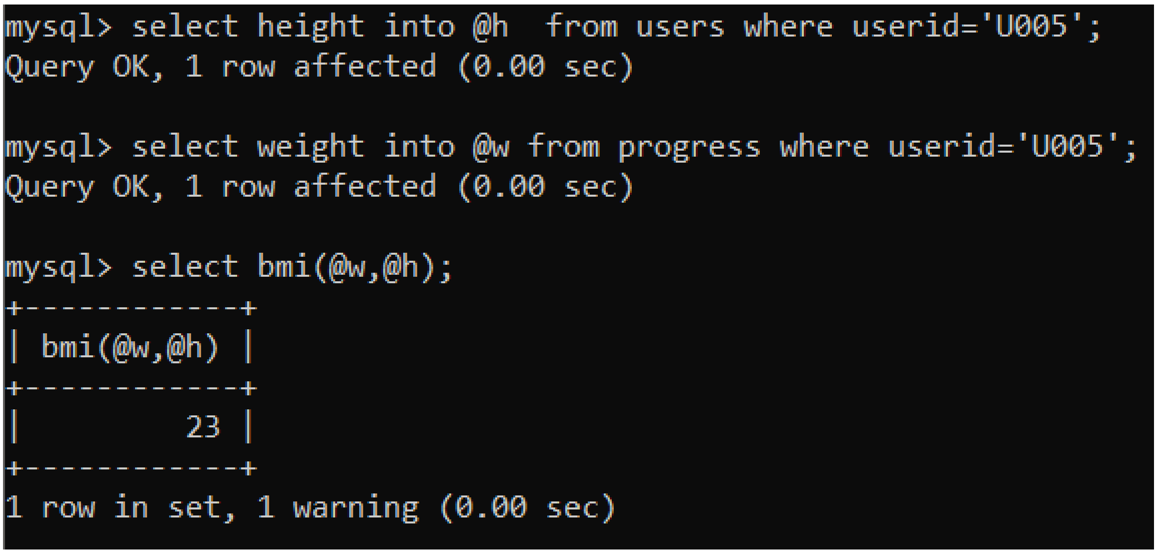
Procedures and Functions



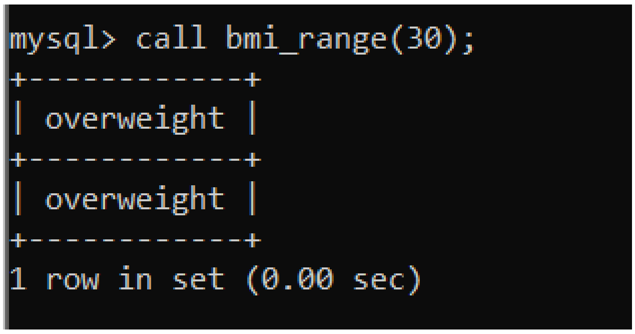
```
mysql> call new_user('Prath','23456','prath@gmail.com','F',23,180,'W010','D010');
Query OK, 1 row affected (0.02 sec)

mysql> select * from users;
+--------+----------+-------------+----------------------+--------+------+--------+------+------+
| userid | name     | password    | email                | gender | age  | height | wid  | did  |
+--------+----------+-------------+----------------------+--------+------+--------+------+------+
| U001   | shivansh | password123 | shivansh001@gmail.com | M      |   30 | 180.00 | W009 | D009 |
| U002   | ayushman | securepass  | ayushman002@gmail.com | F      |   25 | 165.00 | W010 | D010 |
| U003   | pratham  | mypassword  | pratham003@gmail.com | F      |   35 | 170.00 | W011 | D011 |
| U004   | John Doe | password123 | johndoe@example.com  | M      |   25 | 175.00 | W010 | D010 |
| U005   | Prath    | 23456       | prath@gmail.com      | F      |   23 | 180.00 | W010 | D010 |
+--------+----------+-------------+----------------------+--------+------+--------+------+------+
5 rows in set (0.00 sec)
```



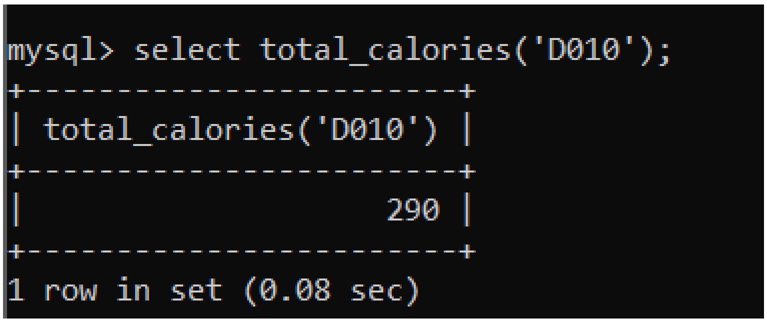
```
mysql> call new_progress('U005', 74, 26, 21, 40, 75, 86);
Query OK, 1 row affected (0.04 sec)

mysql> select * from progress where userid='U005';
+--------+------------+--------+-------+----------+-------+-------+-------+
| userid | date       | weight | arms  | forearms | quads | waist | chest |
+--------+------------+--------+-------+----------+-------+-------+-------+
| U005   | 2024-04-16 |  74.00 | 26.00 |    21.00 | 40.00 | 75.00 | 86.00 |
+--------+------------+--------+-------+----------+-------+-------+-------+
1 row in set (0.04 sec)
```



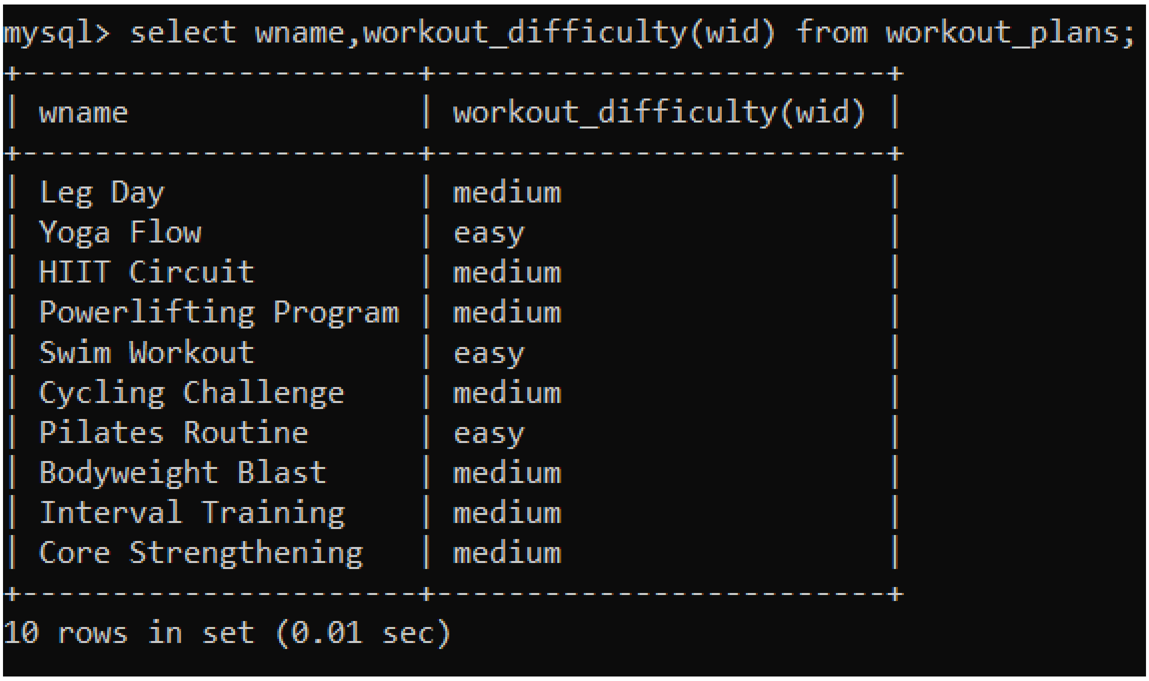
```
mysql> select height into @h  from users where userid='U005';
Query OK, 1 row affected (0.00 sec)

mysql> select weight into @w from progress where userid='U005';
Query OK, 1 row affected (0.00 sec)

mysql> select bmi(@w,@h);
+------------+
| bmi(@w,@h) |
+------------+
|         23 |
+------------+
1 row in set, 1 warning (0.00 sec)
```



```
mysql> call bmi_range(30);
+------------+
| overweight |
+------------+
| overweight |
+------------+
1 row in set (0.00 sec)
```

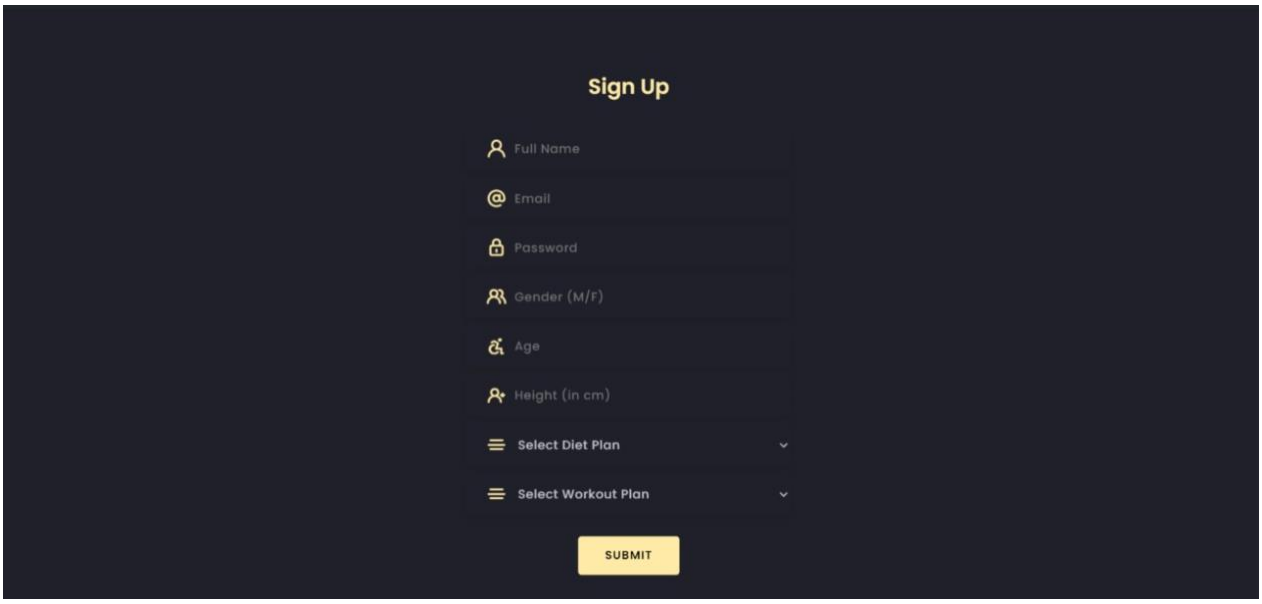


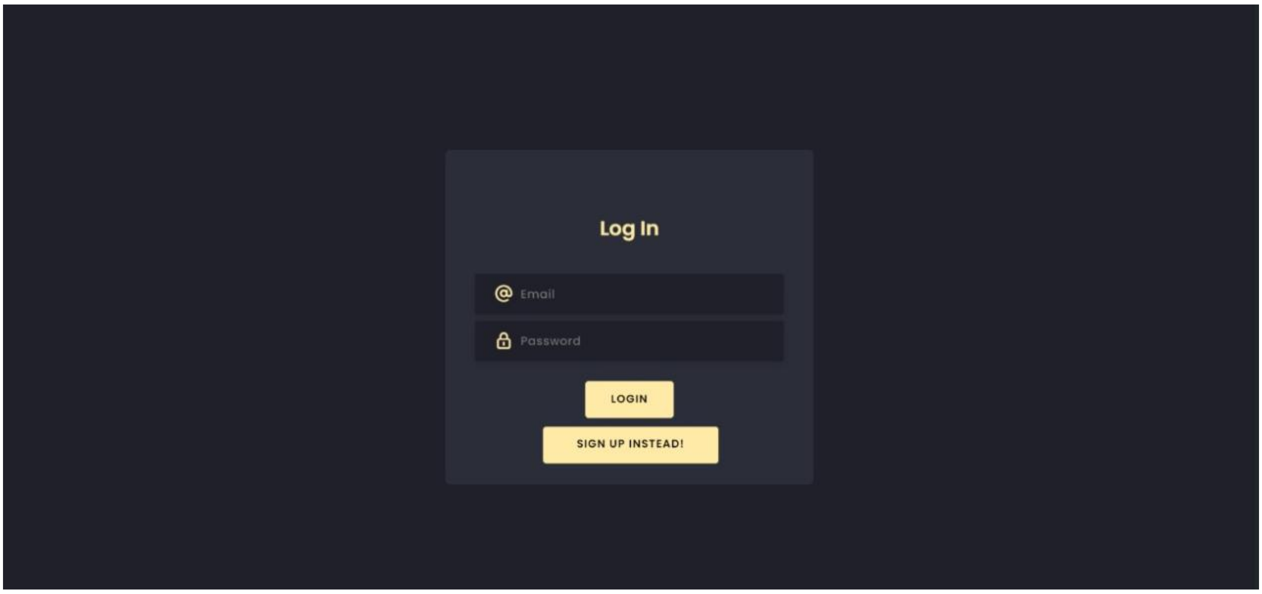
```
mysql> select total_calories('D010');
+------------------------+
| total_calories('D010') |
+------------------------+
|                    290 |
+------------------------+
1 row in set (0.08 sec)
```

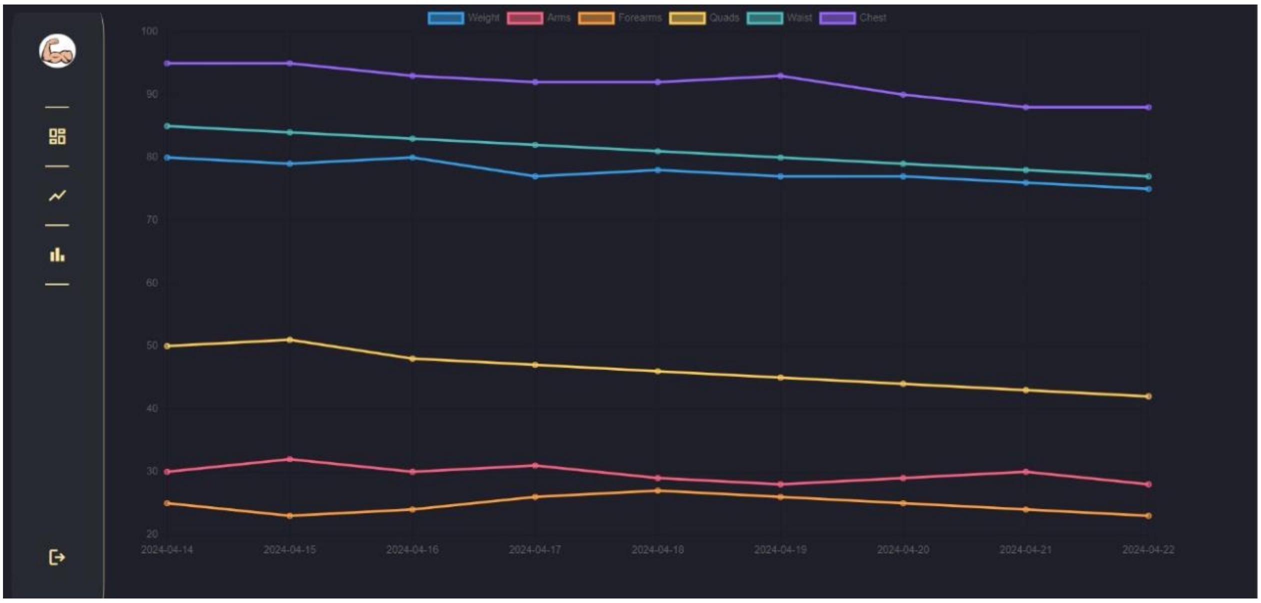


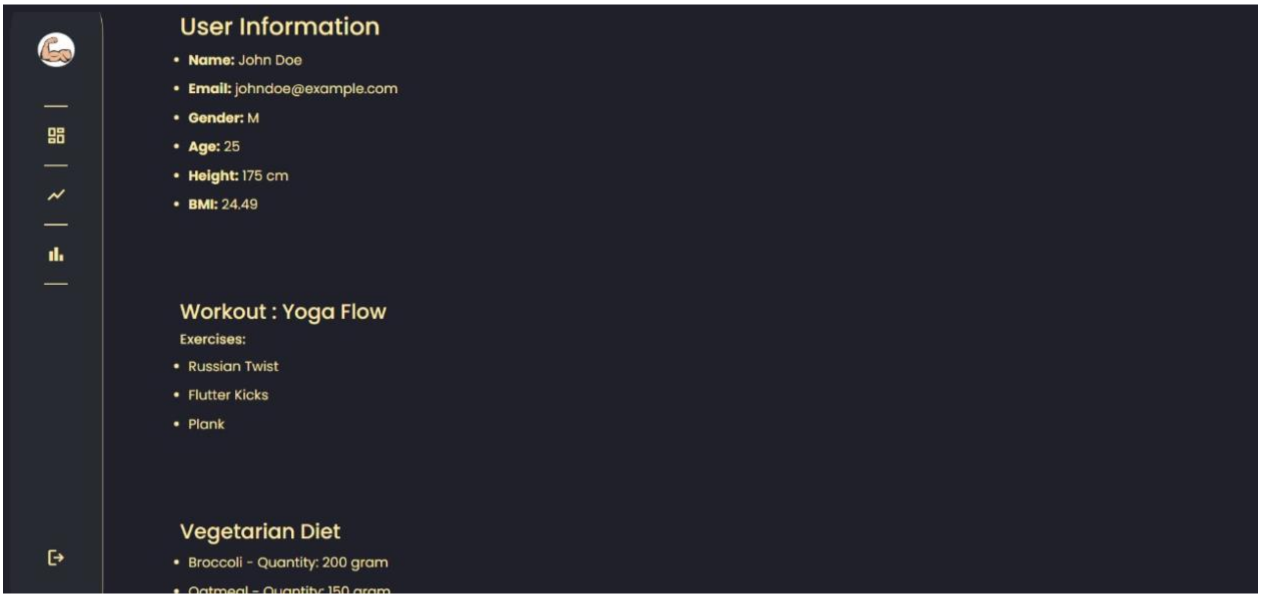
```
mysql> select wname,workout_difficulty(wid) from workout_plans;
+----------------------+-------------------------+
| wname                | workout_difficulty(wid) |
+----------------------+-------------------------+
| Leg Day              | medium                  |
| Yoga Flow            | easy                    |
| HIIT Circuit         | medium                  |
| Powerlifting Program | medium                  |
| Swim Workout         | easy                    |
| Cycling Challenge    | medium                  |
| Pilates Routine      | easy                    |
| Bodyweight Blast     | medium                  |
| Interval Training    | medium                  |
| Core Strengthening   | medium                  |
+----------------------+-------------------------+
10 rows in set (0.01 sec)
```

Front End


Sign Up
Full Name
Email
Password
Gender (M/F)
Age
Height (in cm)
Select Diet Plan
Select Workout Plan
SUBMIT


Log In
Email
Password
LOGIN
SIGN UP INSTEAD!





User Information

- **Name:** John Doe
- **Email:** johndoe@example.com
- **Gender:** M
- **Age:** 25
- **Height:** 175 cm
- **BMI:** 24.49

Workout : Yoga Flow

Exercises:

- Russian Twist
- Flutter Kicks
- Plank

Vegetarian Diet

- Broccoli - Quantity: 200 gram
- Oatmeal - Quantity: 150 gram

Chapter 7
Conclusion and Future Work

In conclusion, the development of the workout planner database project has provided a comprehensive solution for organizing and managing fitness routines effectively. By leveraging modern database technologies, we have created a platform that offers users the ability to plan, track, and analyze their workouts with ease.

Throughout the project, we have focused on several key objectives:

1. User-Friendly Interface: We attempted to create an intuitive and user-friendly interface that allows users of all fitness levels to easily navigate the platform and plan their workouts efficiently.

2. Comprehensive Functionality: Our database encompasses a wide range of features, including exercise libraries, workout plans and progress tracking ensuring that users have all the tools they need to achieve their fitness goals.

3. Scalability and Flexibility: The database architecture is designed to be scalable and flexible, accommodating future expansions and updates as the platform evolves to meet the changing needs of users and the fitness industry.

By addressing these objectives, we believe that our workout planner database project offers significant value to fitness enthusiasts, trainers, and professionals alike. Whether users are looking to lose weight, build muscle, or improve their overall health and fitness, our platform provides the tools and resources necessary to support their journey.

Looking ahead, we are committed to continually enhancing and refining the workout planner to improve our user interface to deliver an even more comprehensive and personalized experience for our users. We will incorporate user feedback, integrate new features, and stay abreast of emerging technologies to ensure that our platform remains at the forefront of the fitness industry.

In conclusion, the workout planner database project represents a significant step forward in empowering individuals to take control of their fitness journey and achieve their goals in a structured and sustainable manner.

Each Team Member Contribution

Ayushman Ranjan	abstract, chapter 1, chapter 4.1, chapter 4.4, insert values
Samyak Singh	chapter 2, chapter 4.1, chapter 4.2, chapter 4.3, chapter 5.1, chapter 5.3
Pratham Ghosh	front end, chapter 3, chapter 4.1
Shivansh Parekh	chapter 4.1, chapter 4.2, chapter 4.3, chapter 5.2, chapter 6
Abeer Sethia	Frontend, Chapter 3, Chapter 4.1, chapter 6, chapter 7