FITNESS DATA HUB

A DBMS Lab Class Project Report

Submitted to the APJ Abdul Kalam Technological University in partial fulfilment of requirements for the award of degree

Bachelor of Technology
in
Information Technology
by

Siva Nandu S (TRV21IT059)
Vishnu V P (TRV21IT064)
Sidharth S (IDK21IT035)

DEPARTMENT OF INFORMATION TECHNOLOGY

GOVERNMENT ENGINEERING COLLEGE, BARTONHILL,

THIRUVANANTHAPURAM – 695035 KERALA

June 2023

23-05-2023

CERTIFICATE OF COMPLETION

This is to certify that the group project on "**Database Management System (DBMS)**" has been successfully completed by:

- Siva Nandu S (TRV21IT059)
- Vishnu V P (TRV21IT064)
- Sidharth S (IDK21IT035)

as a part of their DBMS Group Project at **Government Engineering College, Barton Hill**.

The group project focused on the application and implementation of various concepts related to Database Management System. The team demonstrated excellent teamwork, problem-solving skills, and proficiency in designing and developing a database system.

The project report showcases their in-depth understanding of DBMS concepts, including database design, normalization, SQL queries, and database administration. Their dedication, hard work, and commitment throughout the project duration are commendable.

This certificate acknowledges their outstanding performance, technical competence, and contribution to the field of Database Management System.

Congratulations on the successful completion of the group project!

Jiphi T S

Assistant Professor

Information Technology

Government Engineering College, Barton Hill, Trivandrum

ABSTRACT

Fitness Data Hub

This document presents the abstract for the DBMS group project titled "Fitness Data Hub," submitted by Siva Nandu S, Vishnu V P, and Sidharth S. The project revolves around the management and utilization of data in a gym environment.

The objective of the project is to develop a comprehensive database management system (DBMS) that effectively handles the various aspects of gym data, including member profiles, workout routines, equipment inventory, classes, and fitness goals. The Fitness Data Hub aims to provide a centralized platform for efficient data organization, analysis, and reporting, empowering gym administrators, trainers, and members with valuable insights.

The project encompasses several key components, including database design, data collection and management, analysis and reporting, user interface development, and data security. The database design phase involves the creation of a well-structured schema that accommodates the diverse gym data and ensures data integrity. Data collection and management procedures are implemented to ensure accurate and consistent data entry, enabling reliable analysis and reporting.

The Fitness Data Hub project emphasizes the importance of data analysis in the gym environment. By utilizing SQL queries and data analysis techniques, the system enables users to extract meaningful information from the gym data. This information can be used to make informed decisions regarding member engagement, class scheduling, equipment utilization, and overall gym operations.

To facilitate user interaction, the project includes the development of a user-friendly interface that allows authorized users to access and interact with the database seamlessly. The interface provides convenient features for data retrieval, input, and visualization, enhancing the usability and efficiency of the system.

Data security is a critical aspect of the project. The Fitness Data Hub incorporates measures to ensure the confidentiality and integrity of the gym data, including access controls, encryption techniques, and adherence to data protection regulations.

Overall, the Fitness Data Hub project aims to streamline gym operations, enhance member engagement, and facilitate data-driven decision-making. By leveraging the power of a robust DBMS, the project offers a comprehensive solution for managing and utilizing data in a gym setting. The contributions made by Siva Nandu S, Vishnu V P, and Sidharth S in the successful completion of the project are greatly acknowledged.

Keywords: DBMS, Fitness Data Hub, gym data, database design, data collection, data management, data analysis, reporting, user interface, data security.

ACKNOWLEDGEMENT

We would like to express our sincere gratitude and appreciation to all those who have contributed to the successful completion of our group project on "Database Management System (DBMS)".

First and foremost, we would like to thank Prof. Jiphi T S, our project guide, for his valuable guidance, support, and expertise throughout the project. His insightful suggestions, timely feedback, and constant encouragement played a pivotal role in shaping our project and enhancing our understanding of DBMS concepts.

We extend our heartfelt thanks to our fellow team members, Siva Nandu S, Vishnu V P and Sidharth S, for their unwavering commitment, teamwork, and collaborative efforts in completing this project. Their dedication, hard work, and technical expertise were instrumental in overcoming challenges and achieving project objectives.

We are grateful to our friends and classmates who provided assistance and valuable inputs during the development and testing phases of the project. Their feedback and constructive criticism helped us refine our work and improve the overall quality of the project.

We would also like to express our gratitude to the staff and faculty members of the Information Technology for providing us with the necessary resources, infrastructure, and academic support during the course of our project.

Last but not least, we would like to thank our families for their unwavering support, understanding, and encouragement throughout our academic journey.

We acknowledge that this project would not have been possible without the collective efforts, guidance, and support of all the individuals mentioned above.

Thank you once again to everyone who contributed to our group project on DBMS. We are proud of our accomplishments and the knowledge gained during this experience.

Sincerely,

Siva Nandu S

Vishnu V P

Sidharth S

CONTENT

1. Introduction
2. Objective
3. E R Diagram
4. Relational Schema
5. Implementation
6. Output Screenshots
7. Conclusion
8. References

Introduction

Welcome to the introduction page of our DBMS group project, "Fitness Data Hub." We, Siva Nandu S, Vishnu V P, and Sidharth S, have collaborated to design and develop a comprehensive database management system specifically tailored for managing data in a gym environment. In this project, we aim to address the challenges faced by gyms in efficiently organizing and utilizing their data to enhance operational efficiency and provide valuable insights.

In today's fast-paced world, maintaining a healthy lifestyle has become a priority for many individuals. Gyms play a vital role in promoting fitness and well-being by offering various exercise programs, classes, and personalized training sessions. However, the management of gym-related data, including member information, trainer schedules, equipment inventory, and class registrations, can be a complex and time-consuming task.

The "Fitness Data Hub" project aims to streamline and optimize the management of gym data by creating a centralized database management system. This system will provide a user-friendly interface for gym administrators, trainers, and members to efficiently store, access, and analyze data, resulting in improved operations and better decision-making.

Objective

The objective of our DBMS group project, "Fitness Data Hub," is to design and implement a comprehensive database management system for a fictional gym. The primary goal of this project is to provide the gym management team with an efficient platform to collect, manage, analyze, and leverage data for enhanced gym operations, member engagement, and fitness program optimization.

Key Objectives:

1. Database Design: Develop a well-structured and normalized database schema that efficiently captures and organizes essential information related to gym operations, members, trainers, equipment, classes, and fitness goals.

2. Data Collection and Management: Establish a systematic approach to collect and manage various types of data, including member profiles, attendance records, equipment usage and class schedules.

3. Analysis and Reporting: Implement robust data analysis capabilities to generate meaningful insights and reports for gym administrators, trainers, and management. This includes analyzing member trends, class popularity, peak hours, equipment utilization, and performance tracking.

4. Member Engagement and Personalization: Create features within the database system to track individual member progress, set goals, and provide personalized recommendations for workout routines, nutrition plans, and fitness programs.

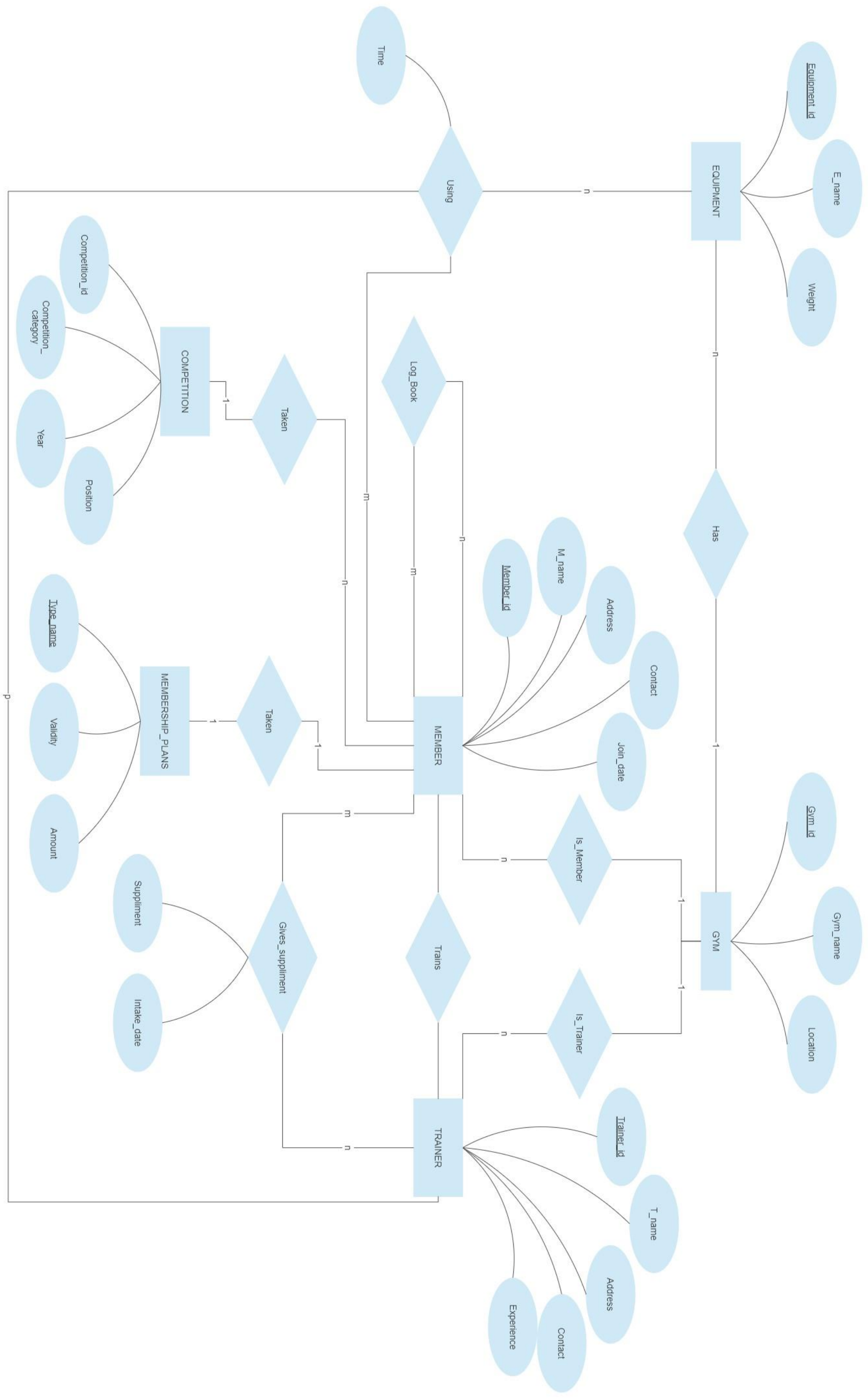
5. Efficiency and Automation: Streamline gym operations by automating routine tasks such as attendance tracking, membership renewals, and class scheduling. Integrate the database system with other systems or tools to enhance efficiency and data accuracy.

6. User-Friendly Interface: Develop a user-friendly interface that allows gym staff, trainers, and administrators to easily access, input, and retrieve data. Ensure the interface provides intuitive data visualization and reporting capabilities.

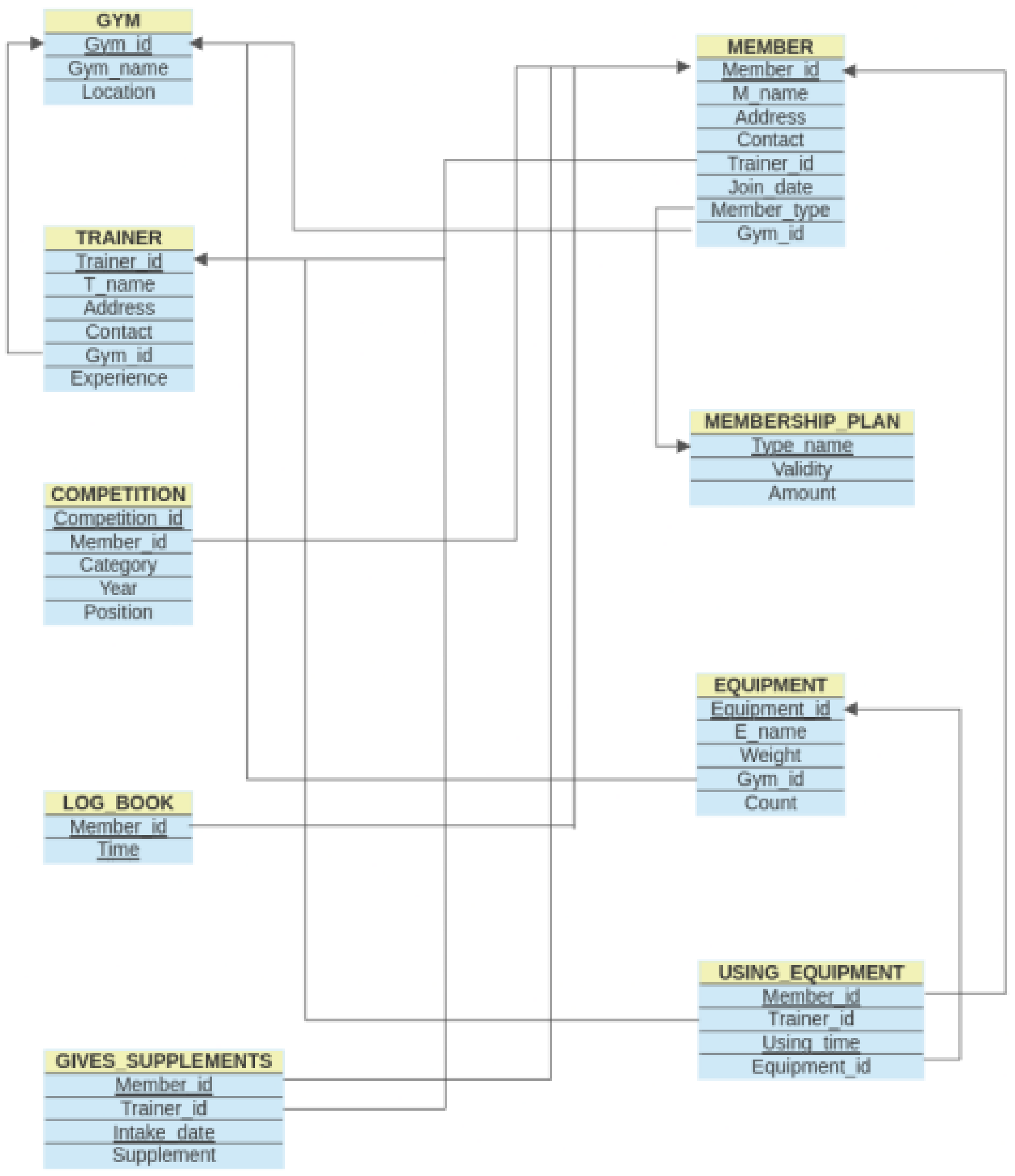
7. Scalability and Future Expansion: Design the database system with scalability in mind, enabling it to accommodate future growth and evolving requirements of the gym. Consider potential expansions, such as multi-location support or integration with external systems.

By achieving these objectives, the Fitness Data Hub project aims to empower the fictional gym with a comprehensive DBMS solution that enhances operational efficiency, member engagement, and data-driven decision-making.

ER diagram


Time
Using
EQUIPMENT
Equipment_id
E_name
Weight
Has
COMPETITION
Competition_id
Competition_category
Year
Position
Taken
Log_Book
MEMBERSHIP_PLANS
Type_name
Validity
Amount
Taken
MEMBER
Member_id
M_name
Address
Contact
Join_date
Is_Member
Trains
Is_Trainer
GYM
Gym_id
Gym_name
Location
Gives_suppliment
Suppliment
Intake_date
TRAINER
Trainer_id
T_name
Address
Contact
Experience
n
m
1
p

Relational Schema Diagram



Implementation

```sql
CREATE DATABASE fitness_data_hub;

USE fitness_data_hub;

CREATE TABLE gym(
    gym_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    gym_name VARCHAR(25) NOT NULL,
    location VARCHAR(255) NOT NULL
);

CREATE TABLE trainer(
    trainer_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    trainer_name VARCHAR(30) NOT NULL,
    address VARCHAR(100),
    contact BIGINT NOT NULL,
    experience INTEGER NOT NULL,
    gym_id INTEGER NOT NULL,
    FOREIGN KEY (gym_id) REFERENCES gym(gym_id)
);

-- (modified)
CREATE TABLE membership_plan(
    type_name VARCHAR(25) PRIMARY KEY,
    expiry_date DATE NOT NULL,
    amount INTEGER NOT NULL
);

-- (modified)
CREATE TABLE member(
    member_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    member_name VARCHAR(30) NOT NULL,
    address VARCHAR(255),
    contact BIGINT NOT NULL,
    join_date DATE NOT NULL,
    gym_id INTEGER NOT NULL,
    trainer_id INTEGER NOT NULL,
    member_type VARCHAR(25) NOT NULL,
    FOREIGN KEY (trainer_id) REFERENCES trainer(trainer_id),
    FOREIGN KEY (gym_id) REFERENCES gym(gym_id),
    FOREIGN KEY (member_type) REFERENCES membership_plan(type_name)
);

CREATE TABLE competition(
    category_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    category_name VARCHAR(25) NOT NULL,
    position INTEGER,
    year INTEGER,
    member_id INTEGER,
    FOREIGN KEY (member_id) REFERENCES member(member_id)
);

-- (modified)
CREATE TABLE equipment(
    equipment_id INTEGER PRIMARY KEY,
    equipment_name VARCHAR(30) NOT NULL,
    weight INTEGER,
    gym_id INTEGER,
    FOREIGN KEY (gym_id) REFERENCES gym(gym_id)
);

CREATE TABLE gives_supplements(
    member_id INTEGER,
    trainer_id INTEGER,
    date_of_intake DATE NOT NULL,
    supplement_name VARCHAR(30) NOT NULL,
    PRIMARY KEY (member_id,date_of_intake),
    FOREIGN KEY (trainer_id) REFERENCES trainer(trainer_id),
    FOREIGN KEY (member_id) REFERENCES member(member_id)
);

CREATE TABLE log_book(
    member_id INTEGER,
    login_date DATETIME,
    PRIMARY KEY (member_id,login_date),
    FOREIGN KEY (member_id) REFERENCES member(member_id)
);

CREATE TABLE using_equipment(
    member_id INTEGER,
    trainer_id INTEGER,
    equipment_id INTEGER,
    date_of_use DATETIME NOT NULL,
    PRIMARY KEY (member_id,date_of_use),
    FOREIGN KEY (member_id) REFERENCES member(member_id),
    FOREIGN KEY (trainer_id) REFERENCES trainer(trainer_id),
    FOREIGN KEY (equipment_id) REFERENCES equipment(equipment_id)
);
-- inserting values to table gym
INSERT INTO gym(gym_name,location) VALUES
    ('Rothman Gym','Trivandrum');

-- inserting values to table trainer
INSERT INTO trainer(trainer_name,address,contact,experience,gym_id) VALUES
    ('Michael','Palayam',9553798011,6,1),
    ('Justin','Vanchiyoor',8351280095,5,1),
    ('Maria','Thampanoor',8769611599,4,1),
    ('Rajesh','Pettah',7255480246,3,1),
    ('Jagath','Kowdiar',9971077633,2,1),
    ('Jennifer','Nedumangad',7643856016 ,1,1);

-- modified table membership_plan
ALTER TABLE membership_plan RENAME COLUMN expiry_date to validity;
ALTER TABLE membership_plan MODIFY COLUMN validity INT NOT NULL;

-- inserting values to table membership_plan
INSERT INTO membership_plan VALUES
    ('Platinum',12,15000),
    ('Gold',6,8000),
    ('Silver',3,5000 ),
    ('Bronze',1, 2000),
    ('Expired',0,0);


-- modified table equipment
DROP TABLE using_equipment;
ALTER TABLE equipment MODIFY equipment_id INT(11) AUTO_INCREMENT;
CREATE TABLE using_equipment(
    member_id INTEGER,
    trainer_id INTEGER,
    equipment_id INTEGER,
    date_of_use DATETIME NOT NULL,
    PRIMARY KEY (member_id,date_of_use),
    FOREIGN KEY (member_id) REFERENCES member(member_id),
    FOREIGN KEY (trainer_id) REFERENCES trainer(trainer_id),
    FOREIGN KEY (equipment_id) REFERENCES equipment(equipment_id)
);

-- modified table equipment
DELETE FROM equipment;
ALTER TABLE equipment ADD column equipment_count INTEGER;
DELETE FROM equipment;
INSERT INTO equipment(equipment_name,weight,equipment_count,gym_id) VALUES
    ('Dumbbell',2,6,1),
    ('Dumbbell',5,6,1),
    ('Dumbbell',10,6,1),
    ('Kettlebell',8,3,1),
    ('Kettlebell',12,3,1),
    ('Kettlebell',16,3,1),
    ('Punching Bag',1,2,1),
    ('Treadmill',NULL,8,1),
    ('Skipping rope',NULL,5,1),
    ('Smith machine',NULL,3,1),
    ('Bench press machine',NULL,3,1),
    ('Leg press machine',NULL,2,1),
    ('Lats pulley',NULL,2,1),
    ('Pull up bars',NULL,4,1),
    ('Barbell',NULL,5,1),
    ('EZ bar',NULL,5,1);
```

```sql
-- modified table member
ALTER TABLE member MODIFY trainer_id INTEGER;
-- inserting values to table member
INSERT INTO
member(member_name,address,contact,join_date,gym_id,trainer_id,member_type)
VALUES
    ('Rohan','Palayam',9376843054,'2023-01-01',1,1,'Platinum'),
    ('Rahul','Kowdiar',9643122032,'2023-01-02',1,2,'Gold'),
    ('Shiva',NULL,9176646363,'2023-01-03',1,3,'Silver'),
    ('Ajay','Pettah',8261428506,'2022-07-01',1,4,'Platinum'),
    ('Karthik','Chakkai',8481814241 ,'2022-08-01',1,5,'Gold'),
    ('Rayhan','Pattom',9778543651,'2021-11-17',1,6,'Bronze'),
    ('Adithya','Kochuveli',7912673384,'2020-07-26',1,NULL,'Platinum'),
    ('Anjali','Attingal',7112504113,'2023-11-21',1,2,'Silver'),
    ('Alvin','Palayam',8158252272,'2023-01-01',1,4,'Platinum'),
    ('Janet','Kattakada',9963713806,'2019-05-30',1,NULL,'Silver'),
    ('Ahmed',NULL,8184177002,'2020-06-19',1,1,'Gold'),
    ('Merin','Perurkada',7241506567,'2023-02-09',1,3,'Gold'),
    ('Tessa','Kowdiar',7525145930,'2023-08-14',1,1,'Platinum'),
    ('Ashley','Pettah',9172432533,'2020-03-23',1,NULL,'Bronze'),
    ('Abel','Pattom',8229423323,'2021-01-12',1,6,'Silver');

-- inserting values into table competition

INSERT INTO competition (category_name,position,year,member_id) VALUES
("Mens Physique",3,2023,2),
("Bodybuilding",2,2022,5),
("Classic Physique",5,2021,15),
("Womens Physique",2,2023,8),
("Mens Physique",5,2023,9),
("Classic Physique",3,2020,7),
("Bodybuilding",3,2023,13),
("Bikini Physique",1,2023,12),
("Bodybuilding",3,2021,6),
("Mens Physique",4,2022,4),
("Womens Physique",2,2020,14);

-- inserting values into table gives_supplements

INSERT INTO gives_supplements VALUES
(1,1,'2023-02-12','Creatine'),
(3,3,'2023-02-12','BCAA'),
(14,NULL,'2023-02-14','Mass-Gainer'),
(9,4,'2023-02-16','Creatine'),
(4,4,'2023-02-21','L-Arginine'),
(3,3,'2023-02-22','Ashvagandha'),
(2,2,'2023-02-22','Citrulline Malate'),
(10,NULL,'2023-03-04','Creatine'),
(6,6,'2023-03-05','L-Arginine'),
(1,1,'2023-03-06','BCAA'),
(3,3,'2023-03-06','Ashvagandha'),
(8,2,'2023-03-07','Citrulline Malate'),
(12,3,'2023-03-08','Mass-Gainer'),
(9,4,'2023-03-09','Creatine');

-- inserting values into table log_book

INSERT INTO log_book VALUES
(2,'2023-03-02 13:14:07'),
(5,'2023-03-02 13:46:34'),
(14,'2023-03-03 05:32:06'),
(7,'2023-03-03 07:34:05'),
(10,'2023-03-04 05:32:23'),
(14,'2023-03-04 14:23:24'),
(1,'2023-03-04 18:45:54'),
(11,'2023-03-04 20:34:45'),
(13,'2023-03-06 05:23:45'),
(11,'2023-03-06 18:45:54'),
(2,'2023-03-07 06:56:01'),
(8,'2023-03-07 07:34:05'),
(3,'2023-03-09 08:56:44'),
(12,'2023-03-09 18:02:00'),
(4,'2023-03-10 05:23:53'),
(13,'2023-03-12 06:34:23'),
(5,'2023-03-12 12:34:23'),
(8,'2023-03-12 12:34:23'),
(9,'2023-03-13 06:34:23'),
(12,'2023-03-13 16:14:29'),
(4,'2023-03-13 19:54:45'),
(2,'2023-03-14 06:04:27'),
(6,'2023-03-14 14:56:01'),
(15,'2023-03-14 20:34:22'),
(3,'2023-03-16 06:09:10'),
(6,'2023-03-16 08:39:23'),
(12,'2023-03-17 08:33:43'),
(9,'2023-03-18 13:45:34'),
(4,'2023-03-19 08:04:56'),
(10,'2023-03-19 15:32:23');

-- inserting values into table using_equipment

INSERT INTO using_equipment VALUES
(2,2,3,'2023-03-02 13:14:07'),
(5,5,5,'2023-03-02 13:46:34'),
(14,NULL,9,'2023-03-03 05:32:06'),
(7,NULL,4,'2023-03-03 07:34:05'),
(10,NULL ,13,'2023-03-04 05:32:23'),
(14,NULL ,15,'2023-03-04 14:23:24'),
(1, 1,1,'2023-03-04 18:45:54'),
(11,1 ,16,'2023-03-04 20:34:45'),
(13, 1,13,'2023-03-06 05:23:45'),
(11, 1,11,'2023-03-06 18:45:54'),
(2, 2,11,'2023-03-07 06:56:01'),
(8, 2,10,'2023-03-07 07:34:05'),
(3, 3,8,'2023-03-09 08:56:44'),
(12, 3,3,'2023-03-09 18:02:00'),
(4, 4,9,'2023-03-10 05:23:53'),
(13,1 ,14,'2023-03-12 06:34:23'),
(5,5,11,'2023-03-12 12:34:23'),
(8, 2,9,'2023-03-12 12:34:23'),
(9, 4,6,'2023-03-13 06:34:23'),
(12, 3,2,'2023-03-13 16:14:29'),
(4, 4,15,'2023-03-13 19:54:45'),
(2, 2,14,'2023-03-14 06:04:27'),
(6, 6,8,'2023-03-14 14:56:01'),
(15, 6,11,'2023-03-14 20:34:22'),
(3, 3,16,'2023-03-16 06:09:10'),
(6, 6,7,'2023-03-16 08:39:23'),
(12, 3,15,'2023-03-17 08:33:43'),
(9, 4,2,'2023-03-18 13:45:34'),
(4, 4,1,'2023-03-19 08:04:56'),
(10, NULL,6,'2023-03-19 15:32:23');
```

DBMS Queries :

1. Count the number of people trained by trainer trainer_name

```
SELECT trainer_name,COUNT(member_id) AS 'Number of Pupil' FROM trainer
INNER JOIN member
ON trainer.trainer_id=member.trainer_id
GROUP BY trainer_name;
```

2. List the details of people who have used equipment equipment_name on a_date

```
SELECT DATE(date_of_use) AS "Date",member_name,equipment_name FROM
using_equipment
NATURAL JOIN member
NATURAL JOIN equipment
ORDER BY date_of_use;
```

3. Display the number of people subscribed to each membership in descending order of count

```
SELECT type_name,COUNT(member_id) FROM member
INNER JOIN membership_plan
ON member.member_type=membership_plan.type_name
GROUP BY type_name;
```

4. list members along with trainer participating in competition

```
SELECT member_name,trainer_name,category_name FROM competition
INNER JOIN member ON member.member_id=competition.member_id
INNER JOIN trainer ON trainer.trainer_id=member.trainer_id;
```

5. Write a procedure to edit details of an equipment . Handle exception for *primary key*

```
DROP PROCEDURE IF EXISTS edit_equipment;
DELIMITER $$
CREATE PROCEDURE edit_equipment(id INTEGER,name
VARCHAR(25),equipment_weight INTEGER,gym INTEGER,count INTEGER)
BEGIN
DECLARE highest_count INTEGER;
SELECT MAX(equipment_id) INTO highest_count FROM equipment;
IF id > highest_count OR id < 1 THEN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No equipment available';
END IF;
UPDATE equipment SET equipment_name=name, weight=equipment_weight,
equipment_count=count WHERE equipment_id=id;
END$$
DELIMITER ;

CALL edit_equipment(22,"Kettlebell",16,1,3);
CALL edit_equipment(6,"Kettlebell",15,1,5);
```

6. Write a procedure to edit the membership plans to rejection after a time

```
DROP PROCEDURE IF EXISTS membership_plan_update;
DELIMITER $$
CREATE PROCEDURE membership_plan_update()
BEGIN
DECLARE plan VARCHAR(15);
DECLARE date_of_join DATE;
DECLARE expiry INTEGER;
DECLARE id INTEGER;
DECLARE f INTEGER DEFAULT 0;
DECLARE cur CURSOR FOR SELECT member_id FROM member;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET f=1;
OPEN cur;
loop1: LOOP
FETCH cur INTO id;
IF f=1 THEN
LEAVE loop1;
END IF;
SELECT member_type INTO plan FROM member WHERE member_id=id;
SELECT join_date INTO date_of_join FROM member WHERE member_id=id;
SELECT validity INTO expiry FROM membership_plan WHERE type_name=plan;
IF month(date_of_join)-month(CURDATE()) NOT BETWEEN -1*expiry AND expiry
THEN
UPDATE member SET member_type="Expired",trainer_id=NULL WHERE member_id=id;
END IF;
END LOOP loop1;
CLOSE cur;
```

```sql
END $$
DELIMITER ;

CALL membership_plan_update();
```

7. Write a function which returns list of supplements available in the gym using cursors(comma separated)

```sql
DROP FUNCTION IF EXISTS supplements;
DELIMITER $$
CREATE FUNCTION supplements()
RETURNS TEXT
DETERMINISTIC
BEGIN
DECLARE supplement VARCHAR(20);
DECLARE supplement_list TEXT DEFAULT '';
DECLARE f INTEGER DEFAULT 0;
DECLARE cur CURSOR FOR SELECT DISTINCT(supplement_name) FROM
gives_supplements;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET f=1;
OPEN cur;
loop1: LOOP
FETCH cur INTO supplement;
IF f=1 THEN LEAVE loop1;
END IF;
SET supplement_list = CONCAT(supplement_list,supplement,', ');
END LOOP loop1;
CLOSE cur;
RETURN supplement_list;
END $$
DELIMITER ;

SELECT supplements();
```

8. Create a view of member names along with their trainer

```sql
DROP VIEW IF EXISTS member_trainer_view;
CREATE VIEW member_trainer_view AS
SELECT member_name,trainer_name FROM member
INNER JOIN trainer ON member.trainer_id = trainer.trainer_id;
```

9. Write a trigger to remove trainers with zero years of experience

```sql
DROP TRIGGER IF EXISTS zero_exp;
DELIMITER $$
CREATE TRIGGER zero_exp
BEFORE INSERT ON trainer
FOR EACH ROW
BEGIN
DELETE FROM trainer WHERE trainer_id = new.trainer_id;
END$$
DELIMITER ;
```

```sql
INSERT INTO trainer(trainer_name,address,contact,experience,gym_id) VALUES
('Roshan','Palayam',9446890901,0,1);
INSERT INTO trainer(trainer_name,address,contact,experience,gym_id) VALUES
('Kalyani','Nedumangaadu',9495676708,0,1);
```

10. Create a view of members and the suppliments they have taken and the date of date_of_intake

```sql
DROP VIEW IF EXISTS member_supplement;
CREATE VIEW member_supplement AS
SELECT member_name,supplement_name FROM member
INNER JOIN gives_supplements ON
member.member_id=gives_supplements.member_id;
```

11. Create a procedure to list the memebers who were in the competition in an year

```sql
DROP PROCEDURE IF EXISTS competition_member;
DELIMITER $$
CREATE PROCEDURE competition_member(in_year INTEGER)
BEGIN
SELECT member_name,category_name from member
INNER JOIN competition ON member.member_id = competition.member_id
WHERE year = in_year;
END$$
DELIMITER ;

CALL competition_member(2022);
```

12. Create a trigger to backup the member data to a new table

```sql
CREATE TABLE IF NOT EXISTS member_back_up(
    member_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    member_name VARCHAR(30) NOT NULL,
    join_date DATE NOT NULL,
    membership_plan VARCHAR(20) NOT NULL
);

DROP TRIGGER IF EXISTS member_back_up;
DELIMITER $$
CREATE TRIGGER member_back_up
BEFORE INSERT ON member
FOR EACH ROW
BEGIN
INSERT INTO member_back_up(member_name,join_date,membership_plan)
VALUES (new.member_name,new.join_date,new.member_type);
END$$
DELIMITER ;

INSERT INTO
member(member_name,address,contact,join_date,gym_id,trainer_id,member_type)
VALUES
```

```
('Jebin','Kowdiar',9564821356,'2023-04-25',1,4,'Silver'),
('Gopika','Pappanamkodu',9223290903,'2023-04-02',1,6,'Gold');
```

13. List the name of members who havenot used any of the equipments

```
SELECT member_name FROM member
WHERE member_id NOT IN (SELECT DISTINCT(member_id) FROM using_equipment);
```

14. Create a New User with only read operation provilage for all tables

```
CREATE USER 'viewer'@'localhost' IDENTIFIED BY 'pass';
GRANT SELECT ON fitness_data_hub.* TO 'viewer'@'localhost' WITH GRANT
OPTION;
```

15. List the names of members who have won medals in any category and order them by position

```
SELECT category_name , member_name, position,year
FROM competition NATURAL JOIN member
WHERE position <=3
ORDER BY position ;
```

16. Count the number of people that came to Gym on 4th March 2023

```
SELECT count(distinct member_id)
FROM log_book
WHERE DATE(login_date) = '2023-03-04';
```

17. Write a function to determine the supplement that is most used in the gym using cursor

```
DROP FUNCTION IF EXISTS most_used_supplement;
DELIMITER $$
CREATE FUNCTION most_used_supplement()
RETURNS VARCHAR(30)
DETERMINISTIC
BEGIN
DECLARE flag INT DEFAULT 0;
DECLARE current_element VARCHAR(30);
DECLARE current_count INT;
DECLARE max_element VARCHAR(30);
DECLARE max_count INT DEFAULT 0;
DECLARE cur CURSOR FOR SELECT supplement_name, count(*) FROM
gives_supplements GROUP BY supplement_name;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag = 1;
OPEN cur;
FETCH cur INTO current_element , current_count;
WHILE flag < 1 DO
IF current_count > max_count THEN
SET max_count = current_count;
SET max_element = current_element;
END IF;
FETCH cur INTO current_element , current_count;
```

```
END WHILE;
CLOSE cur;
RETURN max_element;
END $$
DELIMITER ;

SELECT most_used_supplement();
```

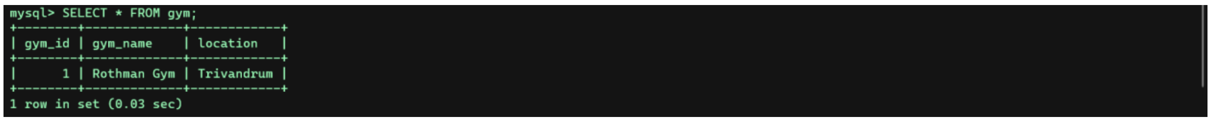
18. Write a procedure to add attribute 'salary' for trainers to table trainer depending on their experience

```
DROP PROCEDURE IF EXISTS make_salary;
DELIMITER $$
CREATE PROCEDURE make_salary()
BEGIN
ALTER TABLE trainer ADD salary BIGINT;
UPDATE trainer set salary = experience*5000;
END $$
DELIMITER ;

CALL make_salary();
```

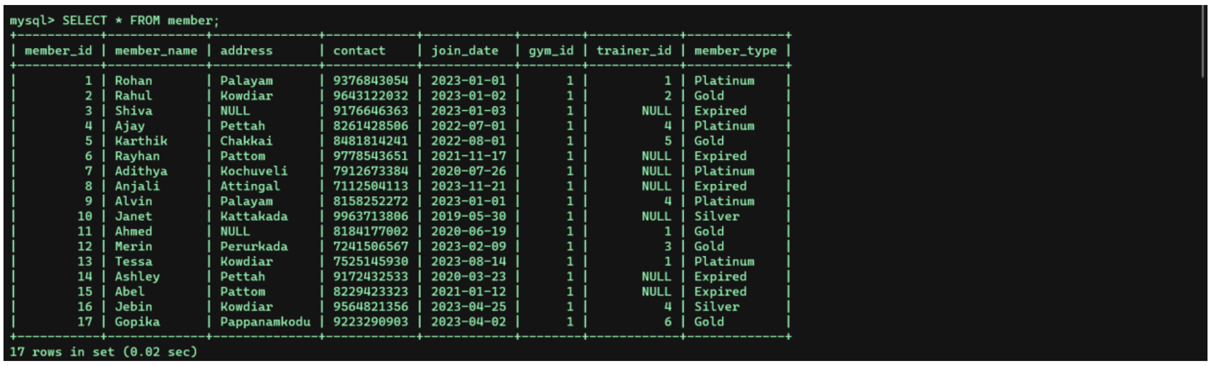
19. Write a function to calculate current income to the gym

```
DROP FUNCTION IF EXISTS calculate_current_income;
DELIMITER $$
CREATE FUNCTION calculate_current_income()
RETURNS INT
DETERMINISTIC
BEGIN
DECLARE amt INT;
DECLARE cnt INT;
DECLARE total INT DEFAULT 0;
DECLARE flag INT DEFAULT 0;
DECLARE cur CURSOR FOR SELECT amount , count(*) FROM membership_plan INNER
JOIN member  ON type_name = member_type GROUP BY amount;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag = 1;
OPEN cur;
FETCH cur INTO amt,cnt;
WHILE flag < 1 DO
SET total = total + amt*cnt;
FETCH cur INTO amt,cnt;
END WHILE;
CLOSE cur;
RETURN total;
END $$
DELIMITER ;

SELECT calculate_current_income();
```

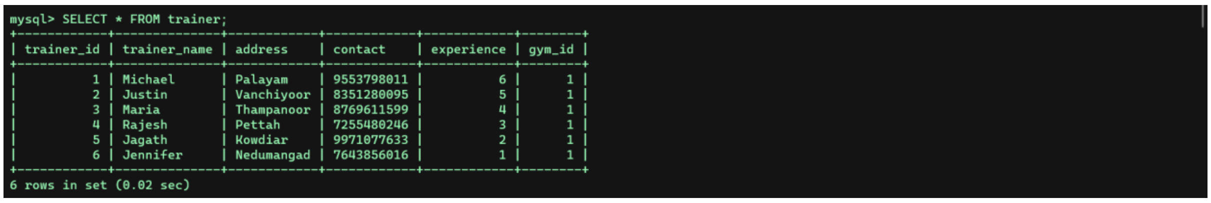
Output Screenshots



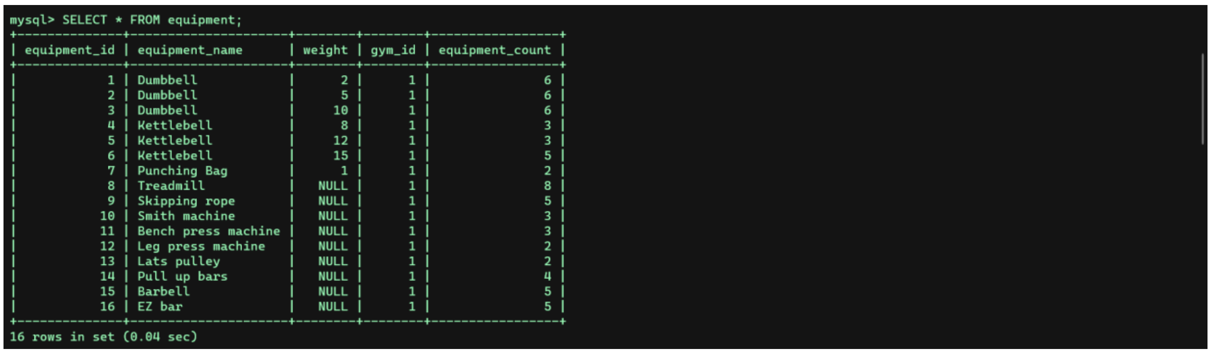
```
mysql> SELECT * FROM gym;
+--------+-------------+------------+
| gym_id | gym_name    | location   |
+--------+-------------+------------+
|      1 | Rothman Gym | Trivandrum |
+--------+-------------+------------+
1 row in set (0.03 sec)
```



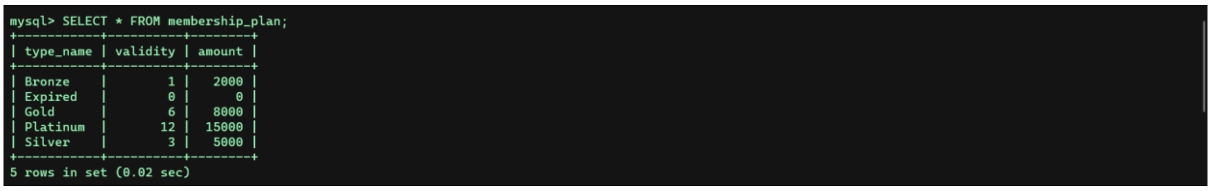
```
mysql> SELECT * FROM member;
+-----------+-------------+--------------+------------+------------+--------+------------+-------------+
| member_id | member_name | address      | contact    | join_date  | gym_id | trainer_id | member_type |
+-----------+-------------+--------------+------------+------------+--------+------------+-------------+
|         1 | Rohan       | Palayam      | 9376843054 | 2023-01-01 |      1 |          1 | Platinum    |
|         2 | Rahul       | Kowdiar      | 9643122032 | 2023-01-02 |      1 |          2 | Gold        |
|         3 | Shiva       | NULL         | 9176646363 | 2023-01-03 |      1 |       NULL | Expired     |
|         4 | Ajay        | Pettah       | 8261428506 | 2022-07-01 |      1 |          4 | Platinum    |
|         5 | Karthik     | Chakkai      | 8481814241 | 2022-08-01 |      1 |          5 | Gold        |
|         6 | Rayhan      | Pattom       | 9778543651 | 2021-11-17 |      1 |       NULL | Expired     |
|         7 | Adithya     | Kochuveli    | 7912673384 | 2020-07-26 |      1 |       NULL | Platinum    |
|         8 | Anjali      | Attingal     | 7112504113 | 2023-11-21 |      1 |       NULL | Expired     |
|         9 | Alvin       | Palayam      | 8158252272 | 2023-01-01 |      1 |          4 | Platinum    |
|        10 | Janet       | Kattakada    | 9963713806 | 2019-05-30 |      1 |       NULL | Silver      |
|        11 | Ahmed       | NULL         | 8184177002 | 2020-06-19 |      1 |          1 | Gold        |
|        12 | Merin       | Perurkada    | 7241506567 | 2023-02-09 |      1 |          3 | Gold        |
|        13 | Tessa       | Kowdiar      | 7525145930 | 2023-08-14 |      1 |          1 | Platinum    |
|        14 | Ashley      | Pettah       | 9172432533 | 2020-03-23 |      1 |       NULL | Expired     |
|        15 | Abel        | Pattom       | 8229423323 | 2021-01-12 |      1 |       NULL | Expired     |
|        16 | Jebin       | Kowdiar      | 9564821356 | 2023-04-25 |      1 |          4 | Silver      |
|        17 | Gopika      | Pappanamkodu | 9223290903 | 2023-04-02 |      1 |          6 | Gold        |
+-----------+-------------+--------------+------------+------------+--------+------------+-------------+
17 rows in set (0.02 sec)
```



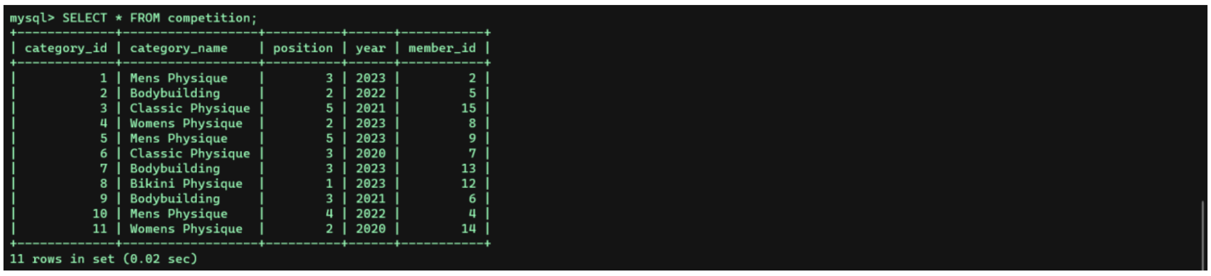
```
mysql> SELECT * FROM trainer;
+------------+--------------+------------+------------+------------+--------+
| trainer_id | trainer_name | address    | contact    | experience | gym_id |
+------------+--------------+------------+------------+------------+--------+
|          1 | Michael      | Palayam    | 9553798011 |          6 |      1 |
|          2 | Justin       | Vanchiyoor | 8351280095 |          5 |      1 |
|          3 | Maria        | Thampanoor | 8769611599 |          4 |      1 |
|          4 | Rajesh       | Pettah     | 7255480246 |          3 |      1 |
|          5 | Jagath       | Kowdiar    | 9971077633 |          2 |      1 |
|          6 | Jennifer     | Nedumangad | 7643856016 |          1 |      1 |
+------------+--------------+------------+------------+------------+--------+
6 rows in set (0.02 sec)
```



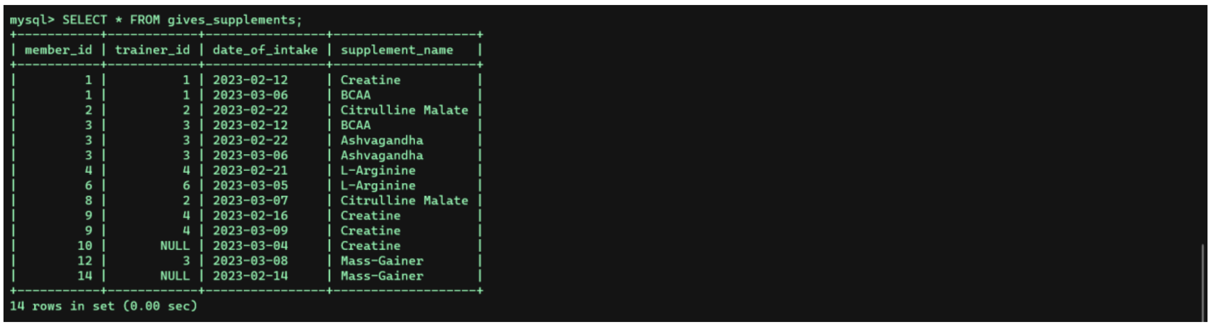
```
mysql> SELECT * FROM equipment;
+--------------+---------------------+--------+--------+-----------------+
| equipment_id | equipment_name      | weight | gym_id | equipment_count |
+--------------+---------------------+--------+--------+-----------------+
|            1 | Dumbbell            |      2 |      1 |               6 |
|            2 | Dumbbell            |      5 |      1 |               6 |
|            3 | Dumbbell            |     10 |      1 |               6 |
|            4 | Kettlebell          |      8 |      1 |               3 |
|            5 | Kettlebell          |     12 |      1 |               3 |
|            6 | Kettlebell          |     15 |      1 |               5 |
|            7 | Punching Bag        |      1 |      1 |               2 |
|            8 | Treadmill           |   NULL |      1 |               8 |
|            9 | Skipping rope       |   NULL |      1 |               5 |
|           10 | Smith machine       |   NULL |      1 |               3 |
|           11 | Bench press machine |   NULL |      1 |               3 |
|           12 | Leg press machine   |   NULL |      1 |               2 |
|           13 | Lats pulley         |   NULL |      1 |               2 |
|           14 | Pull up bars        |   NULL |      1 |               4 |
|           15 | Barbell             |   NULL |      1 |               5 |
|           16 | EZ bar              |   NULL |      1 |               5 |
+--------------+---------------------+--------+--------+-----------------+
16 rows in set (0.04 sec)
```



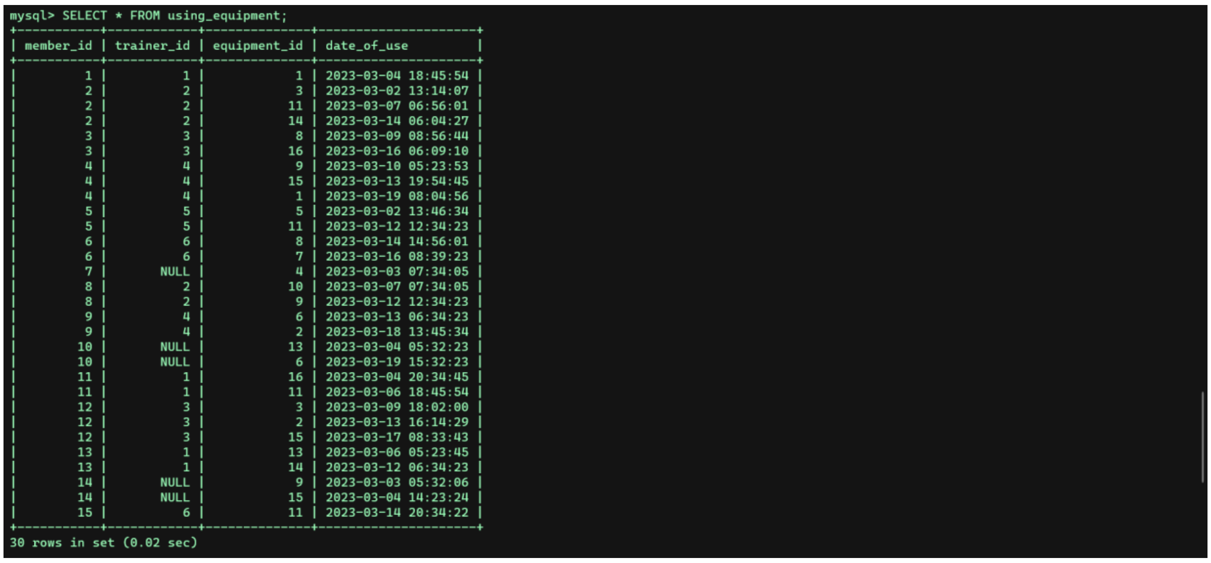
```
mysql> SELECT * FROM membership_plan;
+-----------+----------+--------+
| type_name | validity | amount |
+-----------+----------+--------+
| Bronze    |        1 |   2000 |
| Expired   |        0 |      0 |
| Gold      |        6 |   8000 |
| Platinum  |       12 |  15000 |
| Silver    |        3 |   5000 |
+-----------+----------+--------+
5 rows in set (0.02 sec)
```



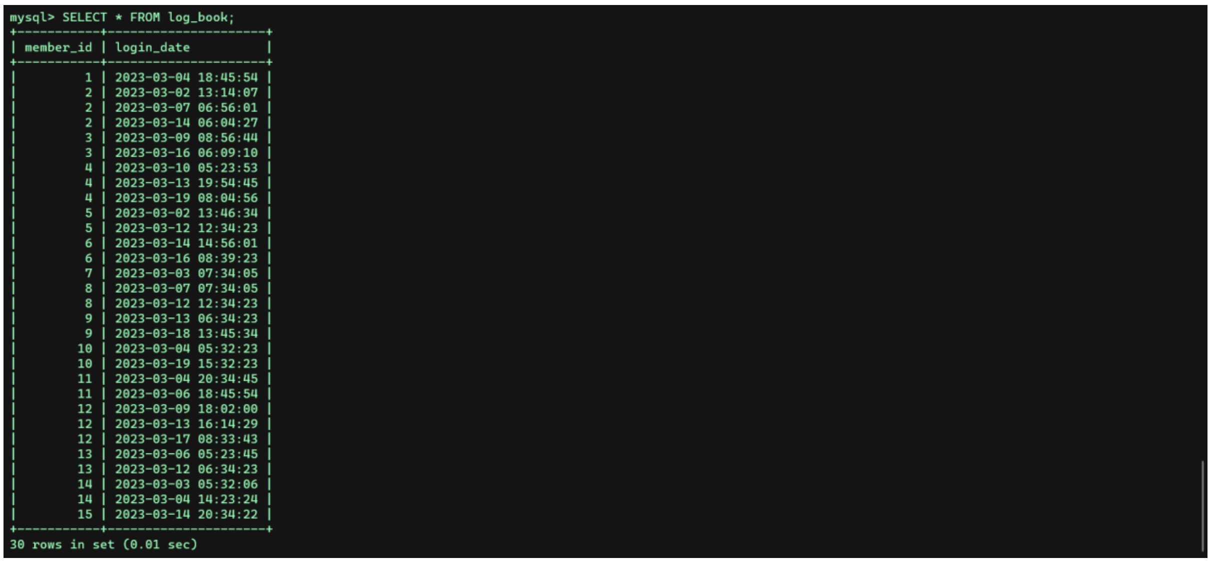
```
mysql> SELECT * FROM competition;
+-------------+------------------+----------+------+-----------+
| category_id | category_name    | position | year | member_id |
+-------------+------------------+----------+------+-----------+
|           1 | Mens Physique    |        3 | 2023 |         2 |
|           2 | Bodybuilding     |        2 | 2022 |         5 |
|           3 | Classic Physique |        5 | 2021 |        15 |
|           4 | Womens Physique  |        2 | 2023 |         8 |
|           5 | Mens Physique    |        5 | 2023 |         9 |
|           6 | Classic Physique |        3 | 2020 |         7 |
|           7 | Bodybuilding     |        3 | 2023 |        13 |
|           8 | Bikini Physique  |        1 | 2023 |        12 |
|           9 | Bodybuilding     |        3 | 2021 |         6 |
|          10 | Mens Physique    |        4 | 2022 |         4 |
|          11 | Womens Physique  |        2 | 2020 |        14 |
+-------------+------------------+----------+------+-----------+
11 rows in set (0.02 sec)
```



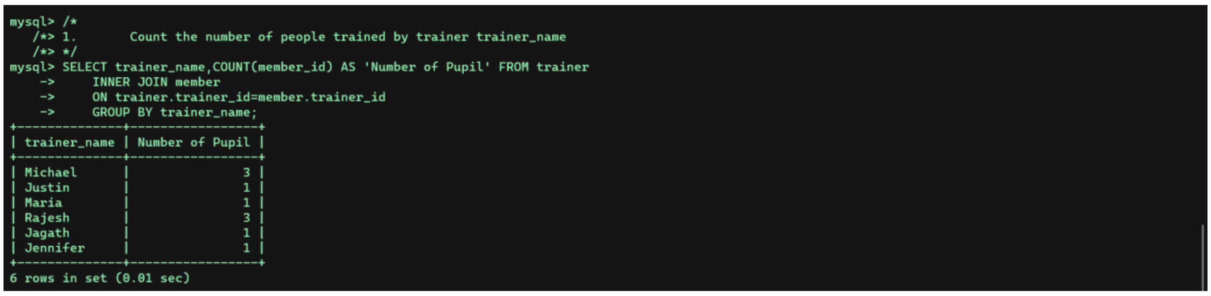
```
mysql> SELECT * FROM gives_supplements;
+-----------+------------+----------------+--------------------+
| member_id | trainer_id | date_of_intake | supplement_name    |
+-----------+------------+----------------+--------------------+
|         1 |          1 | 2023-02-12     | Creatine           |
|         1 |          1 | 2023-03-06     | BCAA               |
|         2 |          2 | 2023-02-22     | Citrulline Malate  |
|         3 |          3 | 2023-02-12     | BCAA               |
|         3 |          3 | 2023-02-22     | Ashvagandha        |
|         3 |          3 | 2023-03-06     | Ashvagandha        |
|         4 |          4 | 2023-02-21     | L-Arginine         |
|         6 |          6 | 2023-03-05     | L-Arginine         |
|         8 |          2 | 2023-03-07     | Citrulline Malate  |
|         9 |          4 | 2023-02-16     | Creatine           |
|         9 |          4 | 2023-03-09     | Creatine           |
|        10 |       NULL | 2023-03-04     | Creatine           |
|        12 |          3 | 2023-03-08     | Mass-Gainer        |
|        14 |       NULL | 2023-02-14     | Mass-Gainer        |
+-----------+------------+----------------+--------------------+
14 rows in set (0.00 sec)
```



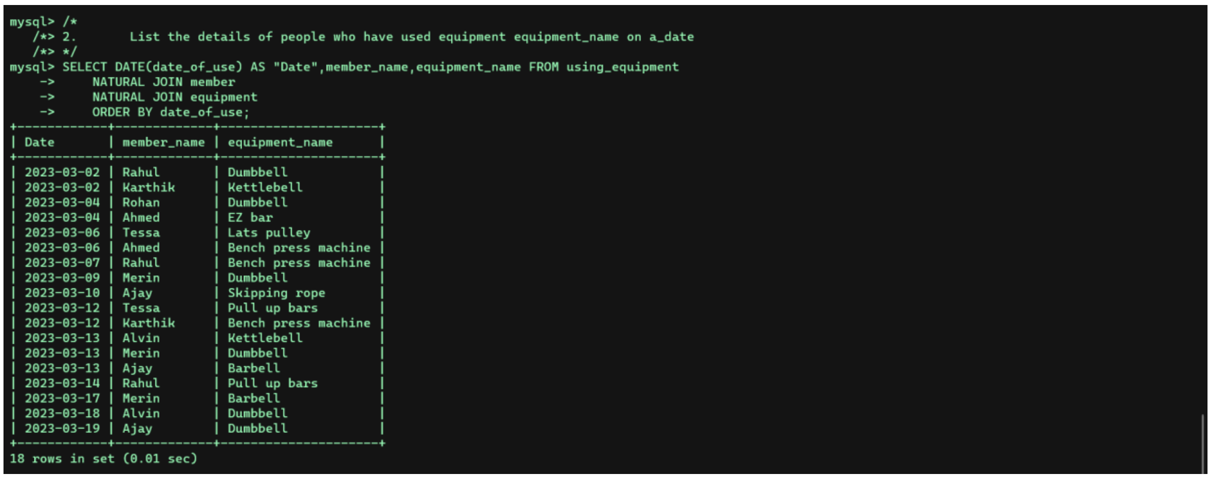
```
mysql> SELECT * FROM using_equipment;
+-----------+------------+--------------+---------------------+
| member_id | trainer_id | equipment_id | date_of_use         |
+-----------+------------+--------------+---------------------+
|         1 |          1 |            1 | 2023-03-04 18:45:54 |
|         2 |          2 |            3 | 2023-03-02 13:14:07 |
|         2 |          2 |           11 | 2023-03-07 06:56:01 |
|         2 |          2 |           14 | 2023-03-14 06:04:27 |
|         3 |          3 |            8 | 2023-03-09 08:56:44 |
|         3 |          3 |           16 | 2023-03-16 06:09:10 |
|         4 |          4 |            9 | 2023-03-10 05:23:53 |
|         4 |          4 |           15 | 2023-03-13 19:54:45 |
|         4 |          4 |            1 | 2023-03-19 08:04:56 |
|         5 |          5 |            5 | 2023-03-02 13:46:34 |
|         5 |          5 |           11 | 2023-03-12 12:34:23 |
|         6 |          6 |            8 | 2023-03-14 14:56:01 |
|         6 |          6 |            7 | 2023-03-16 08:39:23 |
|         7 |       NULL |            4 | 2023-03-03 07:34:05 |
|         8 |          2 |           10 | 2023-03-07 07:34:05 |
|         8 |          2 |            9 | 2023-03-12 12:34:23 |
|         9 |          4 |            6 | 2023-03-13 06:34:23 |
|         9 |          4 |            2 | 2023-03-18 13:45:34 |
|        10 |       NULL |           13 | 2023-03-04 05:32:23 |
|        10 |       NULL |            6 | 2023-03-19 15:32:23 |
|        11 |          1 |           16 | 2023-03-04 20:34:45 |
|        11 |          1 |           11 | 2023-03-06 18:45:54 |
|        12 |          3 |            3 | 2023-03-09 18:02:00 |
|        12 |          3 |            2 | 2023-03-13 16:14:29 |
|        12 |          3 |           15 | 2023-03-17 08:33:43 |
|        13 |          1 |           13 | 2023-03-06 05:23:45 |
|        13 |          1 |           14 | 2023-03-12 06:34:23 |
|        14 |       NULL |            9 | 2023-03-03 05:32:06 |
|        14 |       NULL |           15 | 2023-03-04 14:23:24 |
|        15 |          6 |           11 | 2023-03-14 20:34:22 |
+-----------+------------+--------------+---------------------+
30 rows in set (0.02 sec)
```



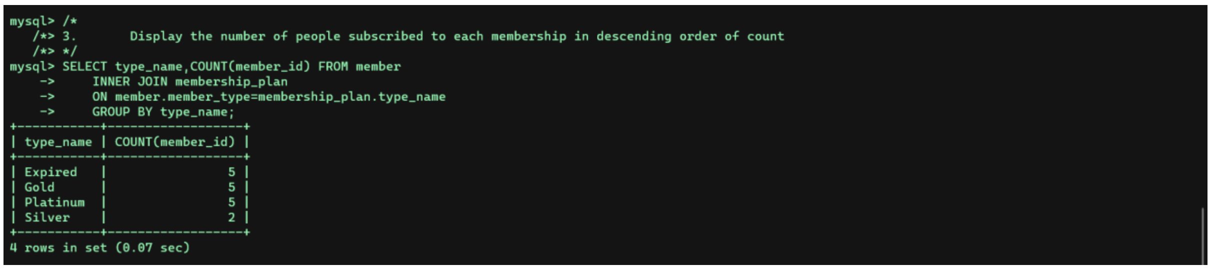
```
mysql> SELECT * FROM log_book;
+-----------+---------------------+
| member_id | login_date          |
+-----------+---------------------+
|         1 | 2023-03-04 18:45:54 |
|         2 | 2023-03-02 13:14:07 |
|         2 | 2023-03-07 06:56:01 |
|         2 | 2023-03-14 06:04:27 |
|         3 | 2023-03-09 08:56:44 |
|         3 | 2023-03-16 06:09:10 |
|         4 | 2023-03-10 05:23:53 |
|         4 | 2023-03-13 19:54:45 |
|         4 | 2023-03-19 08:04:56 |
|         5 | 2023-03-02 13:46:34 |
|         5 | 2023-03-12 12:34:23 |
|         6 | 2023-03-14 14:56:01 |
|         6 | 2023-03-16 08:39:23 |
|         7 | 2023-03-03 07:34:05 |
|         8 | 2023-03-07 07:34:05 |
|         8 | 2023-03-12 12:34:23 |
|         9 | 2023-03-13 06:34:23 |
|         9 | 2023-03-18 13:45:34 |
|        10 | 2023-03-04 05:32:23 |
|        10 | 2023-03-19 15:32:23 |
|        11 | 2023-03-04 20:34:45 |
|        11 | 2023-03-06 18:45:54 |
|        12 | 2023-03-09 18:02:00 |
|        12 | 2023-03-13 16:14:29 |
|        12 | 2023-03-17 08:33:43 |
|        13 | 2023-03-06 05:23:45 |
|        13 | 2023-03-12 06:34:23 |
|        14 | 2023-03-03 05:32:06 |
|        14 | 2023-03-04 14:23:24 |
|        15 | 2023-03-14 20:34:22 |
+-----------+---------------------+
30 rows in set (0.01 sec)
```



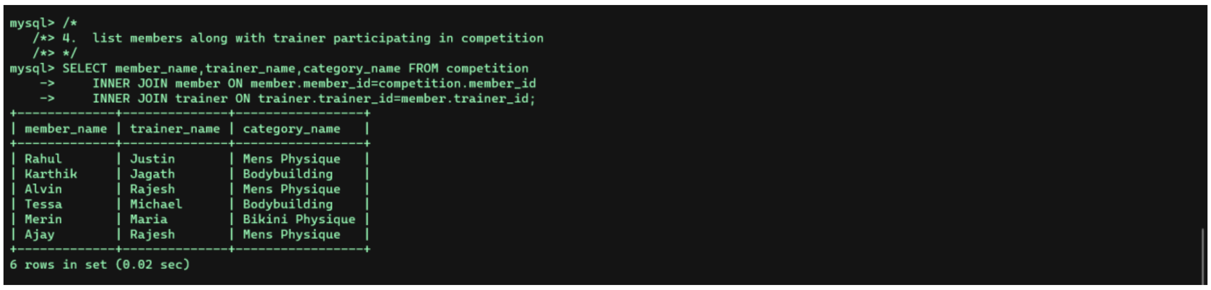
```
mysql> /*
   /*> 1.       Count the number of people trained by trainer trainer_name
   /*> */
mysql> SELECT trainer_name,COUNT(member_id) AS 'Number of Pupil' FROM trainer
    ->     INNER JOIN member
    ->     ON trainer.trainer_id=member.trainer_id
    ->     GROUP BY trainer_name;
+--------------+-----------------+
| trainer_name | Number of Pupil |
+--------------+-----------------+
| Michael      |               3 |
| Justin       |               1 |
| Maria        |               1 |
| Rajesh       |               3 |
| Jagath       |               1 |
| Jennifer     |               1 |
+--------------+-----------------+
6 rows in set (0.01 sec)
```



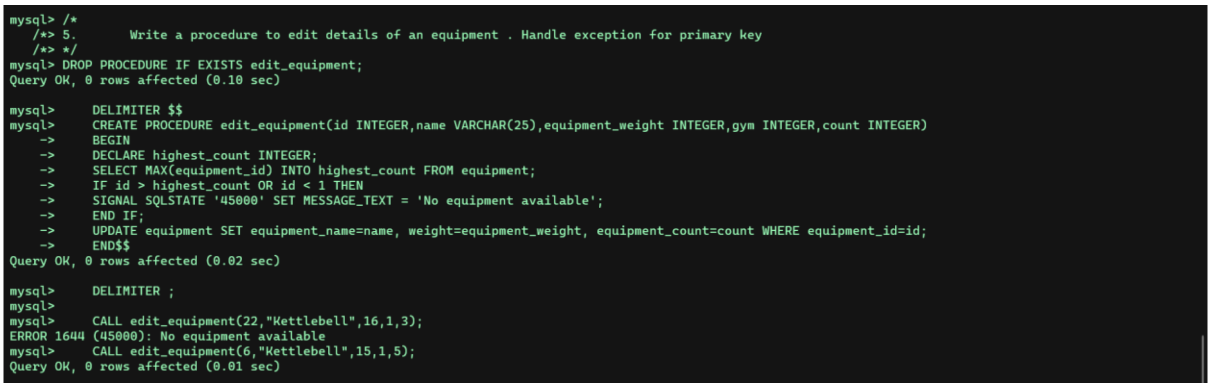
```
mysql> /*
   /*> 2.       List the details of people who have used equipment equipment_name on a_date
   /*> */
mysql> SELECT DATE(date_of_use) AS "Date",member_name,equipment_name FROM using_equipment
    ->     NATURAL JOIN member
    ->     NATURAL JOIN equipment
    ->     ORDER BY date_of_use;
+------------+-------------+---------------------+
| Date       | member_name | equipment_name      |
+------------+-------------+---------------------+
| 2023-03-02 | Rahul       | Dumbbell            |
| 2023-03-02 | Karthik     | Kettlebell          |
| 2023-03-04 | Rohan       | Dumbbell            |
| 2023-03-04 | Ahmed       | EZ bar              |
| 2023-03-06 | Tessa       | Lats pulley         |
| 2023-03-06 | Ahmed       | Bench press machine |
| 2023-03-07 | Rahul       | Bench press machine |
| 2023-03-09 | Merin       | Dumbbell            |
| 2023-03-10 | Ajay        | Skipping rope       |
| 2023-03-12 | Tessa       | Pull up bars        |
| 2023-03-12 | Karthik     | Bench press machine |
| 2023-03-13 | Alvin       | Kettlebell          |
| 2023-03-13 | Merin       | Dumbbell            |
| 2023-03-13 | Ajay        | Barbell             |
| 2023-03-14 | Rahul       | Pull up bars        |
| 2023-03-17 | Merin       | Barbell             |
| 2023-03-18 | Alvin       | Dumbbell            |
| 2023-03-19 | Ajay        | Dumbbell            |
+------------+-------------+---------------------+
18 rows in set (0.01 sec)
```



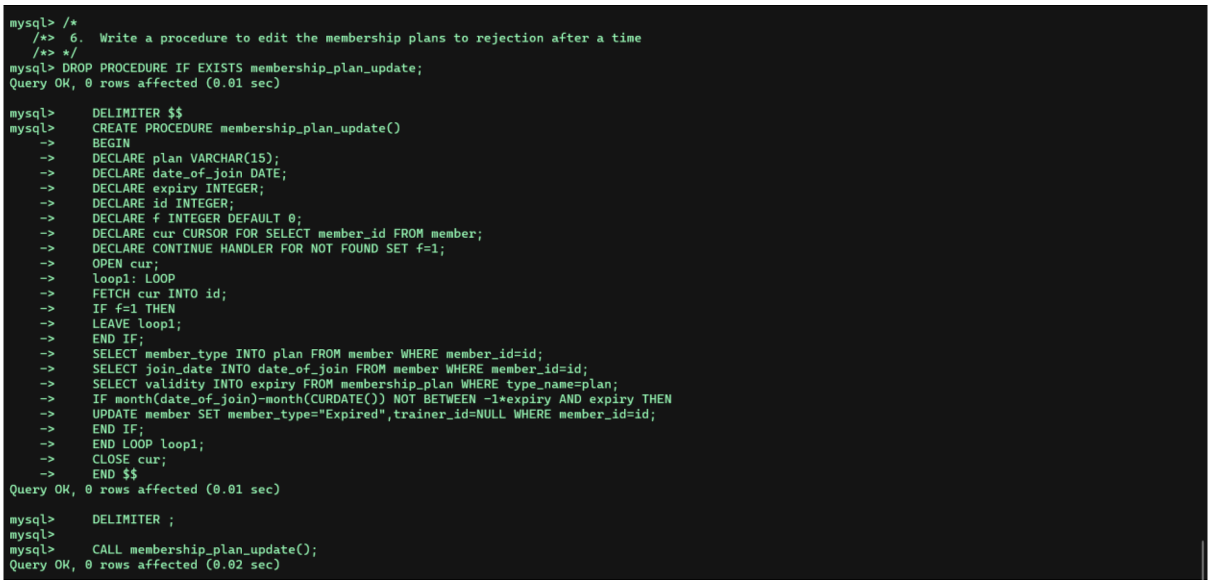
```
mysql> /*
   /*> 3.       Display the number of people subscribed to each membership in descending order of count
   /*> */
mysql> SELECT type_name,COUNT(member_id) FROM member
    ->     INNER JOIN membership_plan
    ->     ON member.member_type=membership_plan.type_name
    ->     GROUP BY type_name;
+-----------+------------------+
| type_name | COUNT(member_id) |
+-----------+------------------+
| Expired   |                5 |
| Gold      |                5 |
| Platinum  |                5 |
| Silver    |                2 |
+-----------+------------------+
4 rows in set (0.07 sec)
```



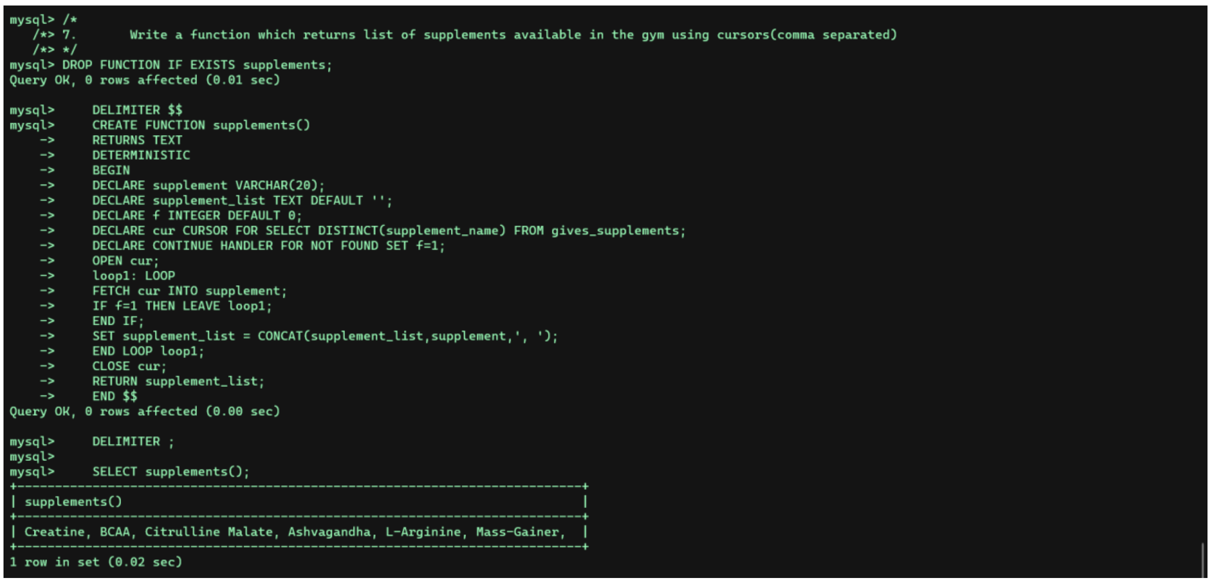
```
mysql> /*
   /*> 4.  list members along with trainer participating in competition
   /*> */
mysql> SELECT member_name,trainer_name,category_name FROM competition
    ->     INNER JOIN member ON member.member_id=competition.member_id
    ->     INNER JOIN trainer ON trainer.trainer_id=member.trainer_id;
+-------------+--------------+-----------------+
| member_name | trainer_name | category_name   |
+-------------+--------------+-----------------+
| Rahul       | Justin       | Mens Physique   |
| Karthik     | Jagath       | Bodybuilding    |
| Alvin       | Rajesh       | Mens Physique   |
| Tessa       | Michael      | Bodybuilding    |
| Merin       | Maria        | Bikini Physique |
| Ajay        | Rajesh       | Mens Physique   |
+-------------+--------------+-----------------+
6 rows in set (0.02 sec)
```



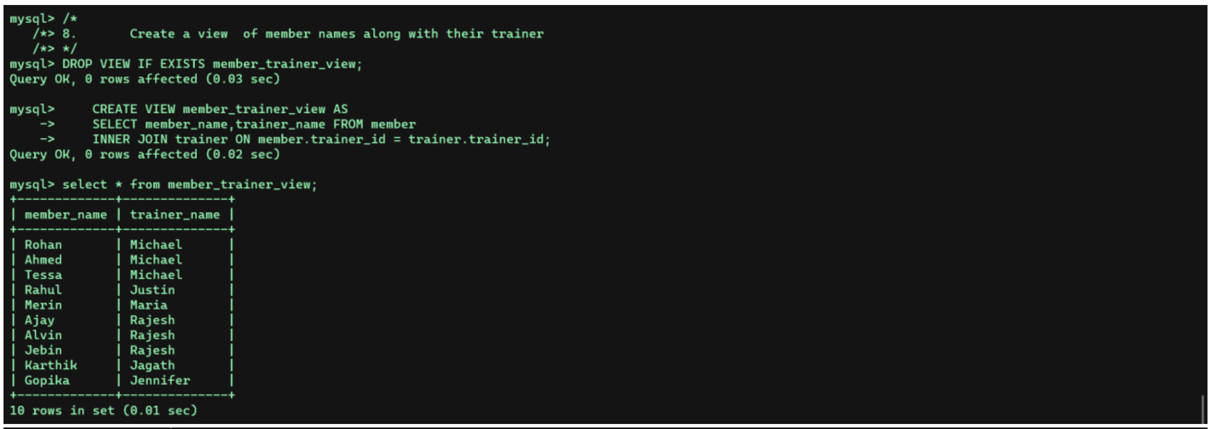
```
mysql> /*
   /*> 5.       Write a procedure to edit details of an equipment . Handle exception for primary key
   /*> */
mysql> DROP PROCEDURE IF EXISTS edit_equipment;
Query OK, 0 rows affected (0.10 sec)

mysql>     DELIMITER $$
mysql>     CREATE PROCEDURE edit_equipment(id INTEGER,name VARCHAR(25),equipment_weight INTEGER,gym INTEGER,count INTEGER)
    ->     BEGIN
    ->     DECLARE highest_count INTEGER;
    ->     SELECT MAX(equipment_id) INTO highest_count FROM equipment;
    ->     IF id > highest_count OR id < 1 THEN
    ->     SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No equipment available';
    ->     END IF;
    ->     UPDATE equipment SET equipment_name=name, weight=equipment_weight, equipment_count=count WHERE equipment_id=id;
    ->     END$$
Query OK, 0 rows affected (0.02 sec)

mysql>     DELIMITER ;
mysql>
mysql>     CALL edit_equipment(22,"Kettlebell",16,1,3);
ERROR 1644 (45000): No equipment available
mysql>     CALL edit_equipment(6,"Kettlebell",15,1,5);
Query OK, 0 rows affected (0.01 sec)
```



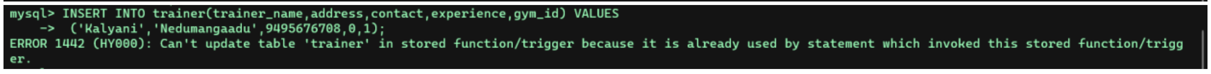
```
mysql> /*
   /*>  6.  Write a procedure to edit the membership plans to rejection after a time
   /*> */
mysql> DROP PROCEDURE IF EXISTS membership_plan_update;
Query OK, 0 rows affected (0.01 sec)

mysql>     DELIMITER $$
mysql>     CREATE PROCEDURE membership_plan_update()
    ->     BEGIN
    ->     DECLARE plan VARCHAR(15);
    ->     DECLARE date_of_join DATE;
    ->     DECLARE expiry INTEGER;
    ->     DECLARE id INTEGER;
    ->     DECLARE f INTEGER DEFAULT 0;
    ->     DECLARE cur CURSOR FOR SELECT member_id FROM member;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET f=1;
    ->     OPEN cur;
    ->     loop1: LOOP
    ->     FETCH cur INTO id;
    ->     IF f=1 THEN
    ->     LEAVE loop1;
    ->     END IF;
    ->     SELECT member_type INTO plan FROM member WHERE member_id=id;
    ->     SELECT join_date INTO date_of_join FROM member WHERE member_id=id;
    ->     SELECT validity INTO expiry FROM membership_plan WHERE type_name=plan;
    ->     IF month(date_of_join)-month(CURDATE()) NOT BETWEEN -1*expiry AND expiry THEN
    ->     UPDATE member SET member_type="Expired",trainer_id=NULL WHERE member_id=id;
    ->     END IF;
    ->     END LOOP loop1;
    ->     CLOSE cur;
    ->     END $$
Query OK, 0 rows affected (0.01 sec)

mysql>     DELIMITER ;
mysql>
mysql>     CALL membership_plan_update();
Query OK, 0 rows affected (0.02 sec)
```



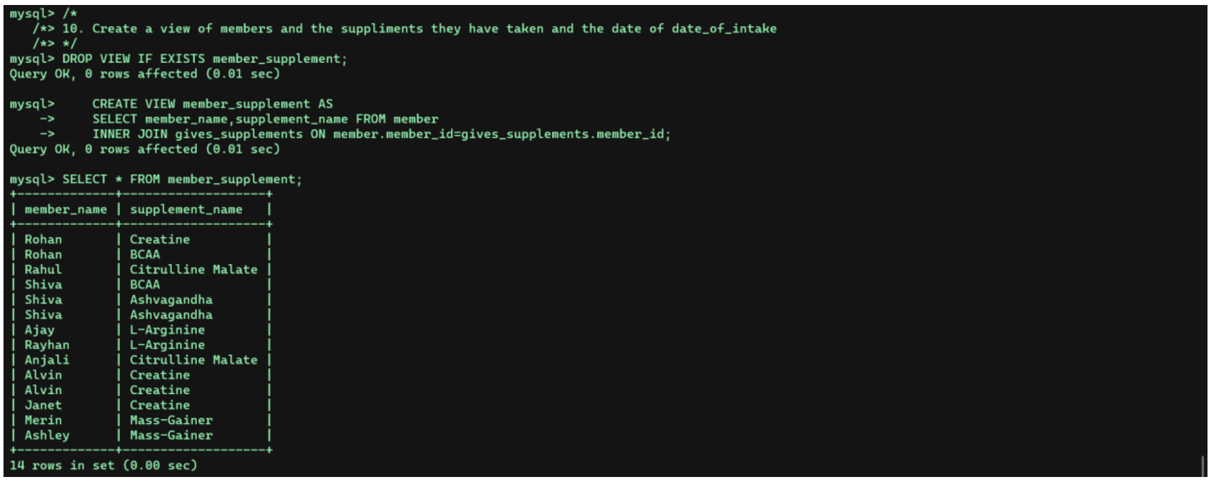
```
mysql> /*
   /*> 7.      Write a function which returns list of supplements available in the gym using cursors(comma separated)
   /*> */
mysql> DROP FUNCTION IF EXISTS supplements;
Query OK, 0 rows affected (0.01 sec)

mysql>     DELIMITER $$
mysql>     CREATE FUNCTION supplements()
    ->     RETURNS TEXT
    ->     DETERMINISTIC
    ->     BEGIN
    ->     DECLARE supplement VARCHAR(20);
    ->     DECLARE supplement_list TEXT DEFAULT '';
    ->     DECLARE f INTEGER DEFAULT 0;
    ->     DECLARE cur CURSOR FOR SELECT DISTINCT(supplement_name) FROM gives_supplements;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET f=1;
    ->     OPEN cur;
    ->     loop1: LOOP
    ->     FETCH cur INTO supplement;
    ->     IF f=1 THEN LEAVE loop1;
    ->     END IF;
    ->     SET supplement_list = CONCAT(supplement_list,supplement,', ');
    ->     END LOOP loop1;
    ->     CLOSE cur;
    ->     RETURN supplement_list;
    ->     END $$
Query OK, 0 rows affected (0.00 sec)

mysql>     DELIMITER ;
mysql>
mysql>     SELECT supplements();
+-----------------------------------------------------------------------------+
| supplements()                                                               |
+-----------------------------------------------------------------------------+
| Creatine, BCAA, Citrulline Malate, Ashvagandha, L-Arginine, Mass-Gainer,   |
+-----------------------------------------------------------------------------+
1 row in set (0.02 sec)
```



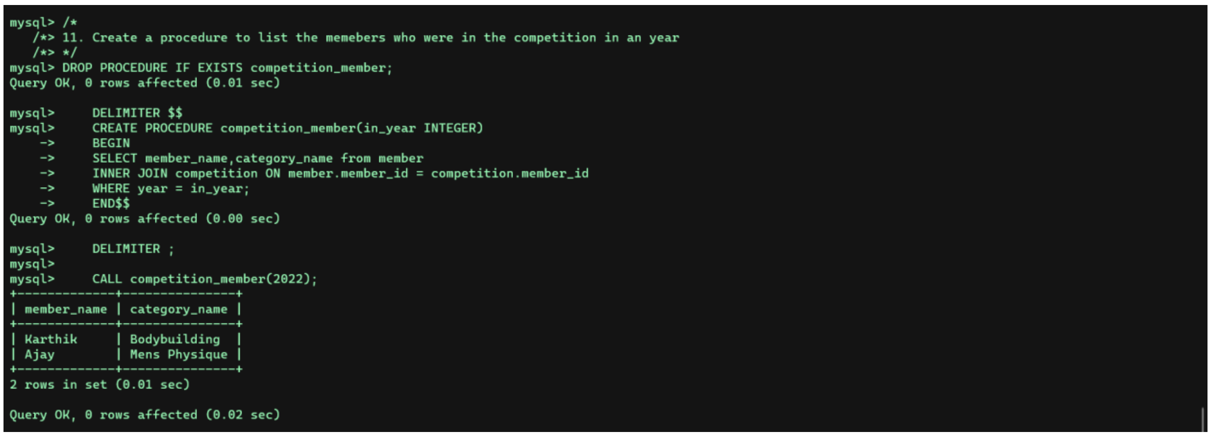
```
mysql> /*
   /*> 8.      Create a view  of member names along with their trainer
   /*> */
mysql> DROP VIEW IF EXISTS member_trainer_view;
Query OK, 0 rows affected (0.03 sec)

mysql>     CREATE VIEW member_trainer_view AS
    ->     SELECT member_name,trainer_name FROM member
    ->     INNER JOIN trainer ON member.trainer_id = trainer.trainer_id;
Query OK, 0 rows affected (0.02 sec)

mysql> select * from member_trainer_view;
+-------------+--------------+
| member_name | trainer_name |
+-------------+--------------+
| Rohan       | Michael      |
| Ahmed       | Michael      |
| Tessa       | Michael      |
| Rahul       | Justin       |
| Merin       | Maria        |
| Ajay        | Rajesh       |
| Alvin       | Rajesh       |
| Jebin       | Rajesh       |
| Karthik     | Jagath       |
| Gopika      | Jennifer     |
+-------------+--------------+
10 rows in set (0.01 sec)
```



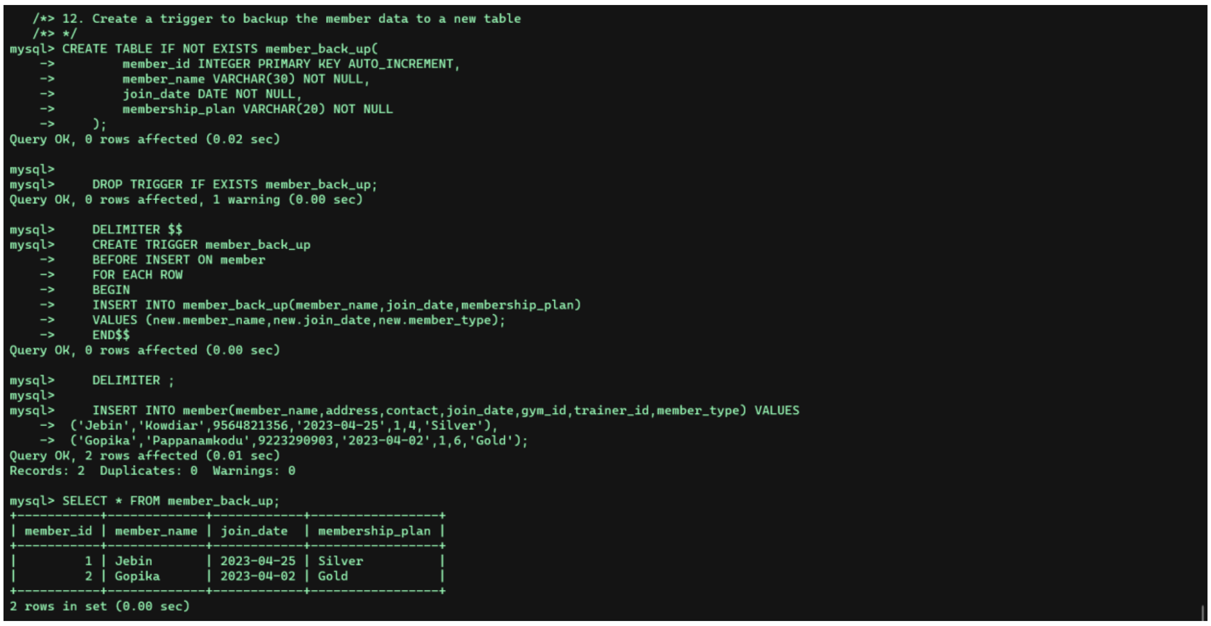
```
mysql> INSERT INTO trainer(trainer_name,address,contact,experience,gym_id) VALUES
    -> ('Kalyani','Nedumangaadu',9495676708,0,1);
ERROR 1442 (HY000): Can't update table 'trainer' in stored function/trigger because it is already used by statement which invoked this stored function/trigg
er.
```



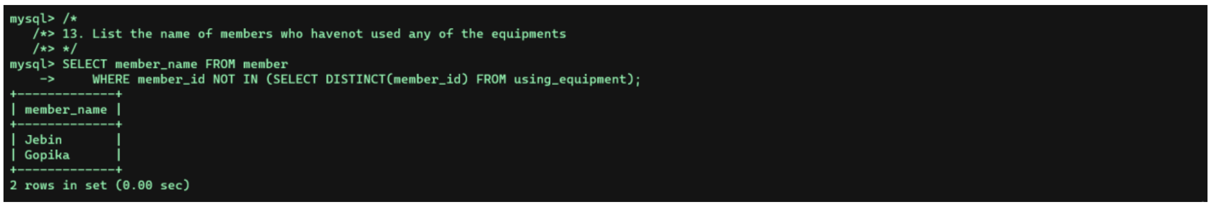
```
mysql> /*
   /*> 10. Create a view of members and the suppliments they have taken and the date of date_of_intake
   /*> */
mysql> DROP VIEW IF EXISTS member_supplement;
Query OK, 0 rows affected (0.01 sec)

mysql>     CREATE VIEW member_supplement AS
    ->     SELECT member_name,supplement_name FROM member
    ->     INNER JOIN gives_supplements ON member.member_id=gives_supplements.member_id;
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM member_supplement;
+-------------+------------------+
| member_name | supplement_name  |
+-------------+------------------+
| Rohan       | Creatine         |
| Rohan       | BCAA             |
| Rahul       | Citrulline Malate |
| Shiva       | BCAA             |
| Shiva       | Ashvagandha      |
| Shiva       | Ashvagandha      |
| Ajay        | L-Arginine       |
| Rayhan      | L-Arginine       |
| Anjali      | Citrulline Malate |
| Alvin       | Creatine         |
| Alvin       | Creatine         |
| Janet       | Creatine         |
| Merin       | Mass-Gainer      |
| Ashley      | Mass-Gainer      |
+-------------+------------------+
14 rows in set (0.00 sec)
```



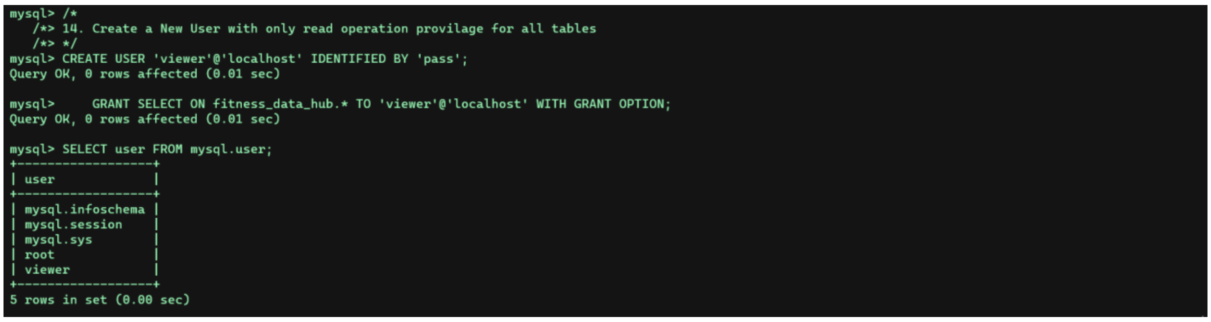
```
mysql> /*
   /*> 11. Create a procedure to list the memebers who were in the competition in an year
   /*> */
mysql> DROP PROCEDURE IF EXISTS competition_member;
Query OK, 0 rows affected (0.01 sec)

mysql>     DELIMITER $$
mysql>     CREATE PROCEDURE competition_member(in_year INTEGER)
    ->     BEGIN
    ->     SELECT member_name,category_name from member
    ->     INNER JOIN competition ON member.member_id = competition.member_id
    ->     WHERE year = in_year;
    ->     END$$
Query OK, 0 rows affected (0.00 sec)

mysql>     DELIMITER ;
mysql>
mysql>     CALL competition_member(2022);
+-------------+---------------+
| member_name | category_name |
+-------------+---------------+
| Karthik     | Bodybuilding  |
| Ajay        | Mens Physique |
+-------------+---------------+
2 rows in set (0.01 sec)

Query OK, 0 rows affected (0.02 sec)
```



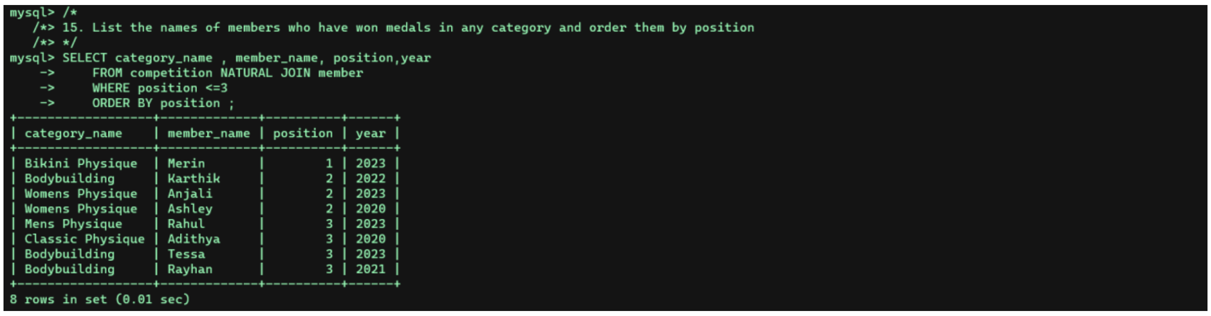
```
   /*> 12. Create a trigger to backup the member data to a new table
   /*> */
mysql> CREATE TABLE IF NOT EXISTS member_back_up(
    ->         member_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    ->         member_name VARCHAR(30) NOT NULL,
    ->         join_date DATE NOT NULL,
    ->         membership_plan VARCHAR(20) NOT NULL
    ->     );
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql>     DROP TRIGGER IF EXISTS member_back_up;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql>     DELIMITER $$
mysql>     CREATE TRIGGER member_back_up
    ->     BEFORE INSERT ON member
    ->     FOR EACH ROW
    ->     BEGIN
    ->     INSERT INTO member_back_up(member_name,join_date,membership_plan)
    ->     VALUES (new.member_name,new.join_date,new.member_type);
    ->     END$$
Query OK, 0 rows affected (0.00 sec)

mysql>     DELIMITER ;
mysql>
mysql>     INSERT INTO member(member_name,address,contact,join_date,gym_id,trainer_id,member_type) VALUES
    ->  ('Jebin','Kowdiar',9564821356,'2023-04-25',1,4,'Silver'),
    ->  ('Gopika','Pappanamkodu',9223290903,'2023-04-02',1,6,'Gold');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM member_back_up;
+-----------+-------------+------------+-----------------+
| member_id | member_name | join_date  | membership_plan |
+-----------+-------------+------------+-----------------+
|         1 | Jebin       | 2023-04-25 | Silver          |
|         2 | Gopika      | 2023-04-02 | Gold            |
+-----------+-------------+------------+-----------------+
2 rows in set (0.00 sec)
```



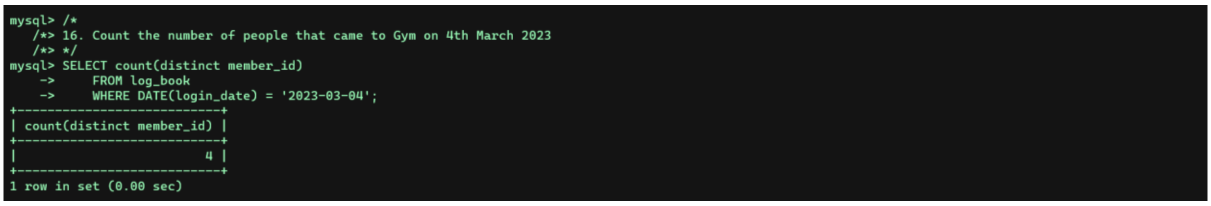
```
mysql> /*
   /*> 13. List the name of members who havenot used any of the equipments
   /*> */
mysql> SELECT member_name FROM member
    ->     WHERE member_id NOT IN (SELECT DISTINCT(member_id) FROM using_equipment);
+-------------+
| member_name |
+-------------+
| Jebin       |
| Gopika      |
+-------------+
2 rows in set (0.00 sec)
```



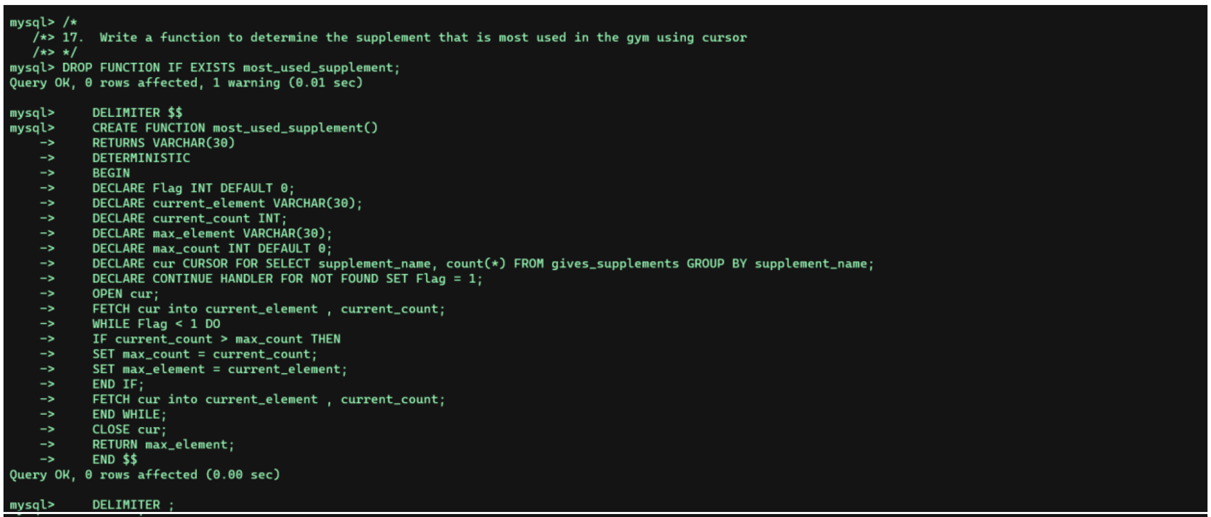
```
mysql> /*
   /*> 14. Create a New User with only read operation provilage for all tables
   /*> */
mysql> CREATE USER 'viewer'@'localhost' IDENTIFIED BY 'pass';
Query OK, 0 rows affected (0.01 sec)

mysql>     GRANT SELECT ON fitness_data_hub.* TO 'viewer'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT user FROM mysql.user;
+------------------+
| user             |
+------------------+
| mysql.infoschema |
| mysql.session    |
| mysql.sys        |
| root             |
| viewer           |
+------------------+
5 rows in set (0.00 sec)
```



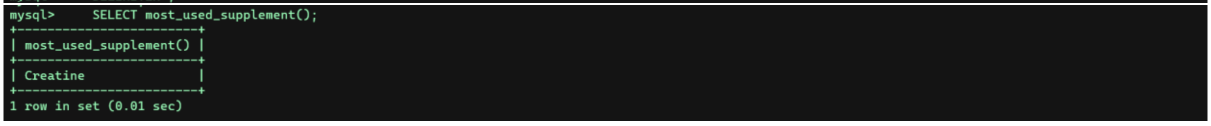
```
mysql> /*
   /*> 15. List the names of members who have won medals in any category and order them by position
   /*> */
mysql> SELECT category_name , member_name, position,year
    ->     FROM competition NATURAL JOIN member
    ->     WHERE position <=3
    ->     ORDER BY position ;
+------------------+-------------+----------+------+
| category_name    | member_name | position | year |
+------------------+-------------+----------+------+
| Bikini Physique  | Merin       |        1 | 2023 |
| Bodybuilding     | Karthik     |        2 | 2022 |
| Womens Physique  | Anjali      |        2 | 2023 |
| Womens Physique  | Ashley      |        2 | 2020 |
| Mens Physique    | Rahul       |        3 | 2023 |
| Classic Physique | Adithya     |        3 | 2020 |
| Bodybuilding     | Tessa       |        3 | 2023 |
| Bodybuilding     | Rayhan      |        3 | 2021 |
+------------------+-------------+----------+------+
8 rows in set (0.01 sec)
```



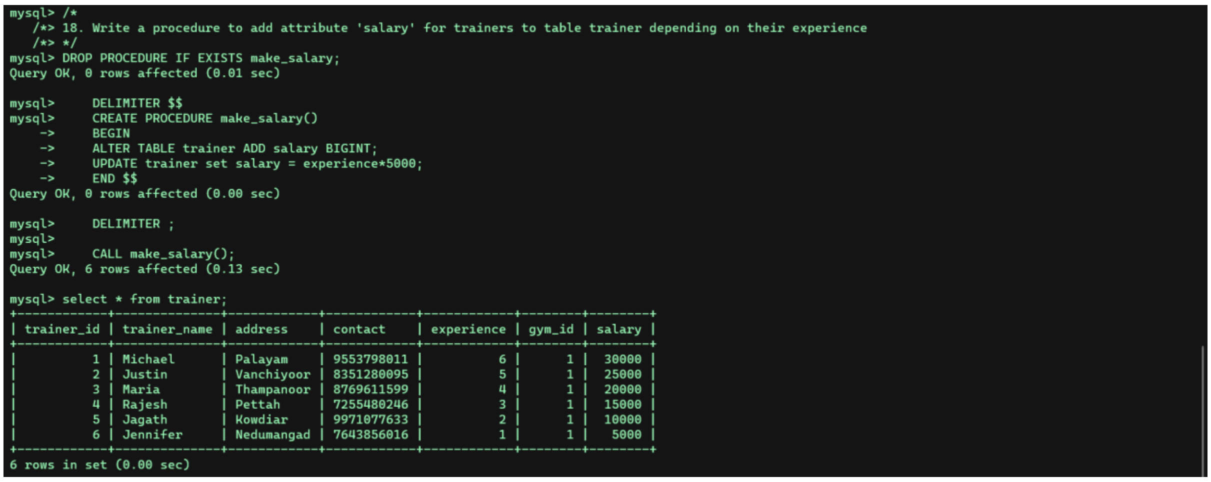
```
mysql> /*
   /*> 16. Count the number of people that came to Gym on 4th March 2023
   /*> */
mysql> SELECT count(distinct member_id)
    ->     FROM log_book
    ->     WHERE DATE(login_date) = '2023-03-04';
+---------------------------+
| count(distinct member_id) |
+---------------------------+
|                         4 |
+---------------------------+
1 row in set (0.00 sec)
```



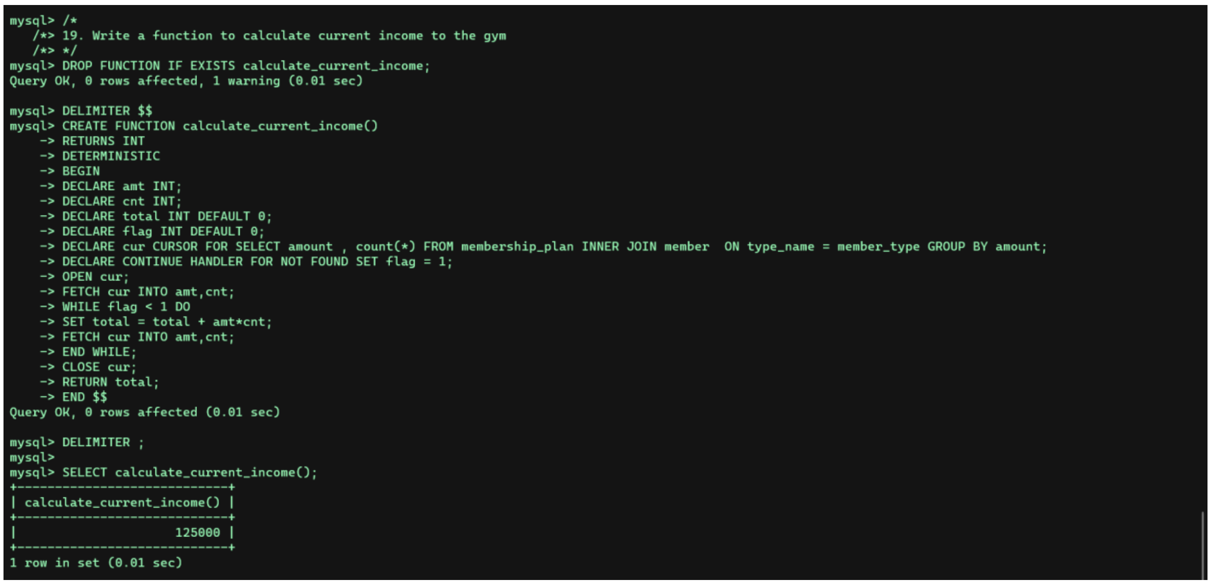
```
mysql> /*
   /*> 17.  Write a function to determine the supplement that is most used in the gym using cursor
   /*> */
mysql> DROP FUNCTION IF EXISTS most_used_supplement;
Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql>     DELIMITER $$
mysql>     CREATE FUNCTION most_used_supplement()
    ->     RETURNS VARCHAR(30)
    ->     DETERMINISTIC
    ->     BEGIN
    ->     DECLARE Flag INT DEFAULT 0;
    ->     DECLARE current_element VARCHAR(30);
    ->     DECLARE current_count INT;
    ->     DECLARE max_element VARCHAR(30);
    ->     DECLARE max_count INT DEFAULT 0;
    ->     DECLARE cur CURSOR FOR SELECT supplement_name, count(*) FROM gives_supplements GROUP BY supplement_name;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET Flag = 1;
    ->     OPEN cur;
    ->     FETCH cur into current_element , current_count;
    ->     WHILE Flag < 1 DO
    ->     IF current_count > max_count THEN
    ->     SET max_count = current_count;
    ->     SET max_element = current_element;
    ->     END IF;
    ->     FETCH cur into current_element , current_count;
    ->     END WHILE;
    ->     CLOSE cur;
    ->     RETURN max_element;
    ->     END $$
Query OK, 0 rows affected (0.00 sec)

mysql>     DELIMITER ;
```



```
mysql>     SELECT most_used_supplement();
+------------------------+
| most_used_supplement() |
+------------------------+
| Creatine               |
+------------------------+
1 row in set (0.01 sec)
```



```
mysql> /*
   /*> 18. Write a procedure to add attribute 'salary' for trainers to table trainer depending on their experience
   /*> */
mysql> DROP PROCEDURE IF EXISTS make_salary;
Query OK, 0 rows affected (0.01 sec)

mysql>     DELIMITER $$
mysql>     CREATE PROCEDURE make_salary()
    ->     BEGIN
    ->     ALTER TABLE trainer ADD salary BIGINT;
    ->     UPDATE trainer set salary = experience*5000;
    ->     END $$
Query OK, 0 rows affected (0.00 sec)

mysql>     DELIMITER ;
mysql>
mysql>     CALL make_salary();
Query OK, 6 rows affected (0.13 sec)

mysql> select * from trainer;
+------------+--------------+------------+------------+------------+--------+--------+
| trainer_id | trainer_name | address    | contact    | experience | gym_id | salary |
+------------+--------------+------------+------------+------------+--------+--------+
|          1 | Michael      | Palayam    | 9553798011 |          6 |      1 |  30000 |
|          2 | Justin       | Vanchiyoor | 8351280095 |          5 |      1 |  25000 |
|          3 | Maria        | Thampanoor | 8769611599 |          4 |      1 |  20000 |
|          4 | Rajesh       | Pettah     | 7255480246 |          3 |      1 |  15000 |
|          5 | Jagath       | Kowdiar    | 9971077633 |          2 |      1 |  10000 |
|          6 | Jennifer     | Nedumangad | 7643856016 |          1 |      1 |   5000 |
+------------+--------------+------------+------------+------------+--------+--------+
6 rows in set (0.00 sec)
```



```
mysql> /*
   /*> 19. Write a function to calculate current income to the gym
   /*> */
mysql> DROP FUNCTION IF EXISTS calculate_current_income;
Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> DELIMITER $$
mysql> CREATE FUNCTION calculate_current_income()
    -> RETURNS INT
    -> DETERMINISTIC
    -> BEGIN
    -> DECLARE amt INT;
    -> DECLARE cnt INT;
    -> DECLARE total INT DEFAULT 0;
    -> DECLARE flag INT DEFAULT 0;
    -> DECLARE cur CURSOR FOR SELECT amount , count(*) FROM membership_plan INNER JOIN member  ON type_name = member_type GROUP BY amount;
    -> DECLARE CONTINUE HANDLER FOR NOT FOUND SET flag = 1;
    -> OPEN cur;
    -> FETCH cur INTO amt,cnt;
    -> WHILE flag < 1 DO
    -> SET total = total + amt*cnt;
    -> FETCH cur INTO amt,cnt;
    -> END WHILE;
    -> CLOSE cur;
    -> RETURN total;
    -> END $$
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql> SELECT calculate_current_income();
+----------------------------+
| calculate_current_income() |
+----------------------------+
|                     125000 |
+----------------------------+
1 row in set (0.01 sec)
```

Conclusion

In conclusion, our group project, Fitness Data Hub, has successfully addressed the challenges and requirements of managing data in a fictional gym through the implementation of a comprehensive database management system (DBMS). Throughout this project, we have achieved the following key milestones and outcomes:

1. Database Design: We meticulously designed the database schema, considering the specific needs of the fictional gym. The schema efficiently captures and organizes essential information such as member profiles, attendance records, workout routines, equipment inventory, classes, and fitness goals.

2. Data Collection and Management: We implemented robust mechanisms for data collection, ensuring the accuracy, consistency, and integrity of the gym-related data. By systematically collecting and managing various types of data, including member details, workout sessions, and equipment usage, we have established a solid foundation for analysis and reporting.

3. Analysis and Reporting: Through the utilization of powerful SQL queries and data analysis techniques, we have gained valuable insights from the gym data. These insights enable gym administrators, trainers, and management to make informed decisions regarding member engagement, class scheduling, equipment utilization, and overall gym operations.

4. User-Friendly Interface: We developed an intuitive and user-friendly interface that allows authorized users to access and interact with the database seamlessly. The interface provides convenient data retrieval, input, and visualization capabilities, enabling efficient monitoring and management of gym-related activities.

5. Member Engagement and Personalization: By incorporating features for member progress tracking, goal setting, and personalized recommendations, we have enhanced member engagement and satisfaction. The system empowers trainers to create tailored workout routines, nutrition plans, and fitness programs to help members achieve their goals effectively.

Throughout the project, our team members, Siva Nandu S, Vishnu V P, and Sidharth S, have collaborated effectively, leveraging our individual strengths and expertise to deliver a comprehensive DBMS solution tailored to the needs of the fictional gym. We would like to express our gratitude to our project guide and mentor for their valuable guidance and support throughout the project.

In conclusion, the Fitness Data Hub project has successfully provided the fictional gym with a powerful tool for managing and leveraging gym-related data. The implemented DBMS solution enables data-driven decision-making, enhances member engagement, and contributes to the overall success of the gym. We are proud of our achievements and look forward to the positive impact this project will have on the fictional gym and its members.

References

1. Elmasri, R., Navathe, S. B. (2016). Fundamentals of Database Systems. Pearson.

2. Date, C. J. (2003). An Introduction to Database Systems. Addison-Wesley.

3. Ramakrishnan, R., Gehrke, J. (2003). Database Management Systems. McGraw-Hill.

4. Connolly, T., Begg, C. (2014). Database Systems: A Practical Approach to Design, Implementation, and Management. Pearson.

5. Silberschatz, A., Korth, H. F., Sudarshan, S. (2013). Database System Concepts. McGraw-Hill.

6. Codd, E. F. (1970). A Relational Model of Data for Large Shared Data Banks. Communications of the ACM, 13(6), 377-387.

7. Garcia-Molina, H., Ullman, J. D., Widom, J. (2008). Database Systems: The Complete Book. Pearson.

8. Oracle Database Documentation: https://docs.oracle.com/en/database/

9. MySQL Documentation: https://dev.mysql.com/doc/

Note: This list is not exhaustive, and additional resources may have been consulted during the project development.