**Case Study: Fitness Center Management System**

**Section 1: Python Standalone Console Application**

Design and implement a standalone console application for a Fitness Center Management System using Python. The application should utilize collections, object-oriented programming (OOP), and exception handling to manage memberships, trainers, classes, and attendance records.

**Requirements:**

1. **Membership Management:**

* Implement the functionality to add, update, and delete membership records.
* Each membership should have attributes such as membership\_id, member\_name, member\_contact, membership\_type, and membership\_expiry.

1. **Trainer Management:**

* Implement the functionality to manage trainers.
* Each trainer should have attributes such as trainer\_id, name, specialty, contact\_info, and availability.

1. **Class Management:**

* Implement the functionality to handle fitness classes.
* Each class should have attributes such as class\_id, class\_name, trainer\_id, schedule, and capacity.

1. **Attendance Management:**

* Implement the functionality to record class attendance.
* Each attendance record should have attributes such as attendance\_id, member\_id, class\_id, and attendance\_date.

**Business Functionalities:**

1. **Manage Memberships:**

* Create a class Membership with attributes membership\_id, member\_name, member\_contact, membership\_type, and membership\_expiry.
* Implement methods to add a new membership, update membership details, and delete a membership from the system.

1. **Manage Trainers:**

* Create a class Trainer with attributes trainer\_id, name, specialty, contact\_info, and availability.
* Implement methods to add a new trainer, update trainer details, and delete a trainer.

1. **Manage Classes:**

* Create a class FitnessClass with attributes class\_id, class\_name, trainer\_id, schedule, and capacity.
* Implement methods to add a new class, update class details, and delete a class.

1. **Manage Attendance:**

* Create a class Attendance with attributes attendance\_id, member\_id, class\_id, and attendance\_date.
* Implement methods to record attendance, update attendance records, and delete attendance records.

**Section 2: MySQL Database Management**

Design a MySQL database schema to support the Fitness Center Management System and provide solutions for the problem statements.

**Table Structures:**

1. **Memberships Table:**

* membership\_id: INT, Primary Key
* member\_name: VARCHAR(255)
* member\_contact: VARCHAR(20)
* membership\_type: VARCHAR(50)
* membership\_expiry: DATE

1. **Trainers Table:**

* trainer\_id: INT, Primary Key
* name: VARCHAR(255)
* specialty: VARCHAR(50)
* contact\_info: VARCHAR(255)
* availability: TEXT

1. **Classes Table:**

* class\_id: INT, Primary Key
* class\_name: VARCHAR(100)
* trainer\_id: INT, Foreign Key References Trainers(trainer\_id)
* schedule: DATETIME
* capacity: INT

1. **Attendance Table:**

* attendance\_id: INT, Primary Key
* member\_id: INT, Foreign Key References Memberships(membership\_id)
* class\_id: INT, Foreign Key References Classes(class\_id)
* attendance\_date: DATE

**Problem Statements:**

* Write a query to find the total number of members whose memberships expire in the next month.
* Write a query to find the trainers who are available for a specific specialty.
* Write a query to find the classes that are fully booked.
* Write a query to find the members who have attended the most classes in the past month.
* Write a query to find the schedule and details of classes conducted by a specific trainer.