Problem Statement:

Managing a gym becomes increasingly complex as the number of members grows and a variety of membership plans and services are introduced. With members joining and leaving regularly, and daily check-ins to track, it becomes difficult to maintain accurate records using manual methods or basic spreadsheet tools. These outdated systems often lead to errors, data loss, double entries, or delays in updating attendance and payment records. They also lack the ability to generate useful reports or summaries that management can use for decision-making. As a result, staff may struggle to provide efficient service, and members may experience inconvenience due to mismanaged schedules or payment issues. Implementing a well-designed database system helps automate key operations such as member registration, attendance tracking, and payment management. It ensures that data is consistent, up to date, and easily accessible. Moreover, it simplifies reporting tasks and enhances the overall efficiency and reliability of gym operations, leading to better service for members and smoother day-to-day management for staff.

Objective:

The primary objective of this project is to design and implement a comprehensive relational database management system (RDBMS) tailored specifically for the operations of a gym or fitness center. With the rise in fitness awareness and the growing number of gym memberships, it becomes essential to have a system that can handle various aspects of gym management efficiently and securely.

This Gym Management System is aimed at replacing manual, paper-based, or spreadsheet-dependent methods with a centralized, digital solution that ensures accuracy, ease of access, and better decision-making capabilities for the gym's administration

The system encompasses four major functional domains:

Membership Plan Management:

The database is structured to manage multiple membership plans that differ by duration, pricing, and included benefits. These plans could range from basic access to premium packages with personal training, classes, or diet consultations. Each plan is stored with comprehensive details such as its name, price, duration in months, special benefits, and the creation date. This module ensures that the gym can easily modify, track, and assign plans to members based on their preferences.

Member Registration and Tracking:

Every individual who joins the gym is stored as a member in the database with unique identification. Their personal details, contact information, membership plan, and start/end dates are recorded. The system also ensures data integrity through constraints like unique email addresses to prevent duplication. This module makes it possible to track member lifecycles, identify expired memberships, and generate engagement reports.

Attendance Monitoring:

The database allows the gym to maintain a daily attendance log for each member. For every visit, the member's presence or absence is recorded against a specific date. This enables the management to analyze attendance trends, identify inactive members, and even create incentives for regular participation. The attendance module also supports report generation for daily, weekly, or monthly check-ins.

Payment and Financial Records:

Managing finances is a critical part of gym operations. This system supports the recording of payments made by each member including the payment date, amount paid, method (e.g., Cash, Credit Card, Bank Transfer), and current status (Completed or Pending). It helps in tracking outstanding dues, generating revenue reports, and maintaining complete financial transparency. This module is essential for auditing purposes and financial forecasting.

In addition to the operational modules, the system is designed with scalability and flexibility in mind. The database follows the principles of normalization to reduce redundancy, improve query performance, and ensure data consistency. The use of primary and foreign keys enforces relational integrity, and the structured design allows for easy integration with front-end applications such as websites or mobile apps for member self-service.

Furthermore, the Gym Management System includes a suite of analytical queries to generate insights into business performance. Examples include total revenue calculation, identifying members with pending payments, attendance summaries, and plan usage statistics. These insights empower management to make data-driven decisions for improving services and increasing customer satisfaction.

In conclusion, the objective is not just to build a database but to create a dynamic, scalable, and intelligent system that automates and optimizes gym operations, enhances member experience, and supports the strategic growth of the fitness center.

Key Requirements:

1. **Member Management:**
   - Store personal information such as name, date of birth, and contact details.
   - Maintain registration dates for each member.
   - Ensure each member has a unique email address to avoid duplication.
2. **Membership Plan Management:**

- Define various membership plans (e.g., Basic, Standard, Premium).
- Include duration, pricing, and benefits for each plan.
- Allow easy association of members with their selected plans.

3. **Attendance Tracking:**

- Record daily attendance status (e.g., Present, Absent) for each member.
- Maintain attendance history for reporting and performance tracking.
- Prevent duplicate entries for the same member on the same date.

4. **Payment Processing:**

- Log payment transactions, including payment date, amount, and method (e.g., Cash, Credit Card, Bank Transfer).
- Track payment status (e.g., Completed, Pending) for each member.
- Associate payments with the respective membership plan and duration.

5. **Reporting and Analytics:**

- Generate meaningful reports such as:
    - Total revenue collected
    - Members with pending payments
    - Attendance statistics for a specific time frame
    - Active vs. expired memberships
    - Members attending regularly vs. irregularly
    - Plan-wise revenue distribution
    - Payment trends over time

6. **System Integrity:**

- Enforce data integrity using primary and foreign keys.
- Apply constraints to validate inputs and prevent inconsistencies.
- Ensure smooth linking between members, plans, attendance, and payments for consistent database operations.

  . m ,zx;s,

Goals & Benefits:

- Efficient record keeping and automation of gym operations
- Accurate tracking of attendance and payments
- Improved data consistency and reliability
- Support for real-time analytics and business decisions
- Scalability for future gym expansion

Tools and Technologies:

- **Query Language:**
  SQL (Structured Query Language) - Used for schema creation, data manipulation, and complex querying.
- **Development Environment:**
  MySQL Workbench - Used for database design, writing SQL queries, and visualizing table relationships.

Database Schema with Table Descriptions:

1. MembershipPlans:

Stores information about different gym membership plans.

Column Name	Data Type	Description
plan_id	INT, PK, AI	Unique ID for each membership plan.
plan_name	VARCHAR(100)	Name of the membership plan.
duration	INT	Duration of the plan in months.
price	DECIMAL(10,2)	Cost of the membership.
benefits	TEXT	Description of the plan features.
created_at	DATE	Date the plan was created.

2. Members:

Holds member personal and contact details.

Column Name	Data Type	Description
member_id	INT, PK, AI	Unique identifier for each member.
first_name	VARCHAR(50)	Member's first name.
last_name	VARCHAR(50)	Member's last name.
date_of_birth	DATE	Date of birth.
contact_number	VARCHAR(15)	Mobile number.
email	VARCHAR(100)	Unique email address.
membership_plan_id	INT, FK	Linked to MembershipPlans.plan_id.
start_date	DATE	Membership start date.
end_date	DATE	Membership end date.

3. AttendanceRecords

Records attendance for each member on a daily basis.

Column Name	Data Type	Description
attendance_id	INT, PK, AI	Unique attendance record ID.
member_id	INT, FK	Linked to Members.member_id.
attendance_date	DATE	Date of attendance.
status	ENUM	Attendance status: Present or Absent.

4. Payments

Tracks all payments made by gym members.

Column Name	Data Type	Description
payment_id	INT, PK, AI	Unique payment transaction ID.
member_id	INT, FK	Linked to Members.member_id.

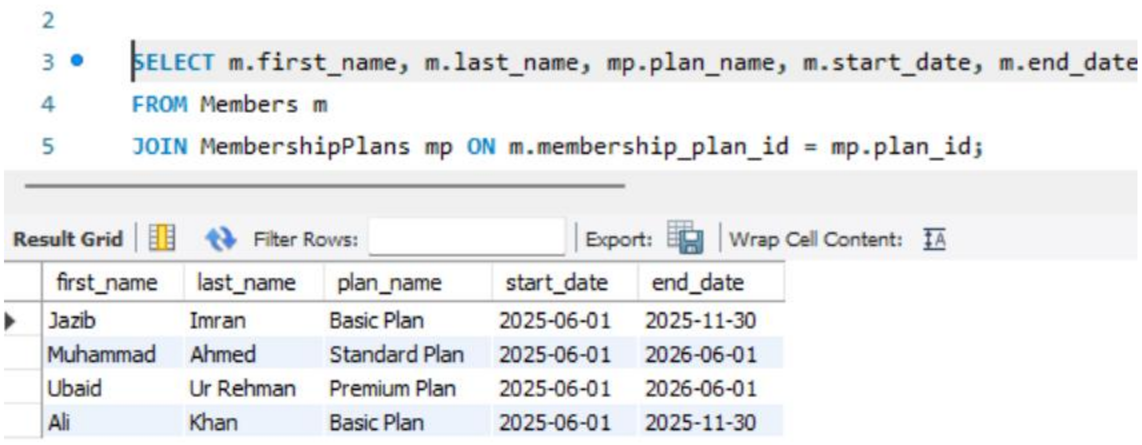
payment_date	DATE	Date of payment.
amount	DECIMAL(10,2)	Amount paid.
payment_method	ENUM	Method used: Cash, Credit Card, Bank Transfer.
payment_status	ENUM	Payment status: Completed or Pending.

Constraints Applied (PK, FK, etc.):

- Primary Key (PK): Each table has a primary key for uniquely identifying rows.
- Foreign Key (FK): Used in Members, AttendanceRecords, and Payments to enforce relationships with referenced tables.
- Auto Increment (AI): Primary keys are auto-incremented to assign unique IDs automatically.
- UNIQUE Constraint: Applied on the email column in the Members table to avoid duplicate email entries.
- ENUM Constraint: Used in AttendanceRecords and Payments to restrict the values of attendance status and payment method/status.

Screenshots of queries and outputs:

1:

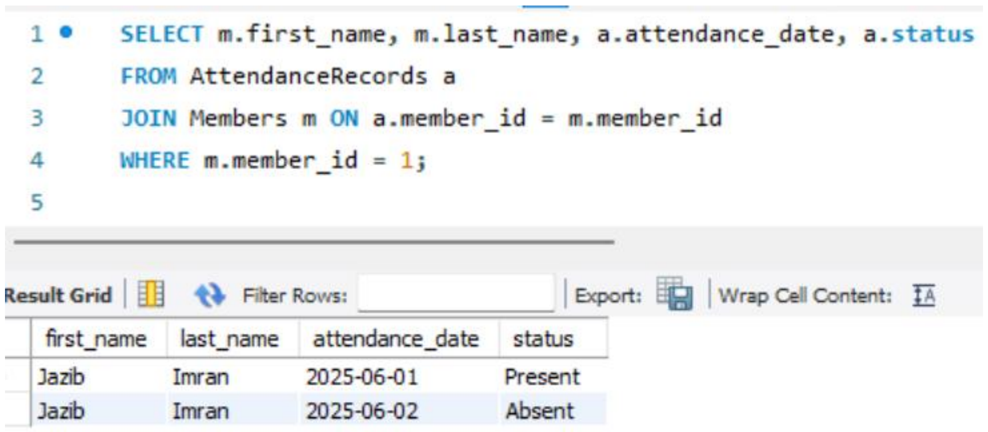


```
SELECT m.first_name, m.last_name, mp.plan_name, m.start_date, m.end_date
FROM Members m
JOIN MembershipPlans mp ON m.membership_plan_id = mp.plan_id;
```

first_name	last_name	plan_name	start_date	end_date
Jazib	Imran	Basic Plan	2025-06-01	2025-11-30
Muhammad	Ahmed	Standard Plan	2025-06-01	2026-06-01
Ubaid	Ur Rehman	Premium Plan	2025-06-01	2026-06-01
Ali	Khan	Basic Plan	2025-06-01	2025-11-30

This query retrieves the first and last names of members along with their membership plan name, start date, and end date. It joins the Members and MembershipPlans tables using the foreign key membership_plan_id.

2:

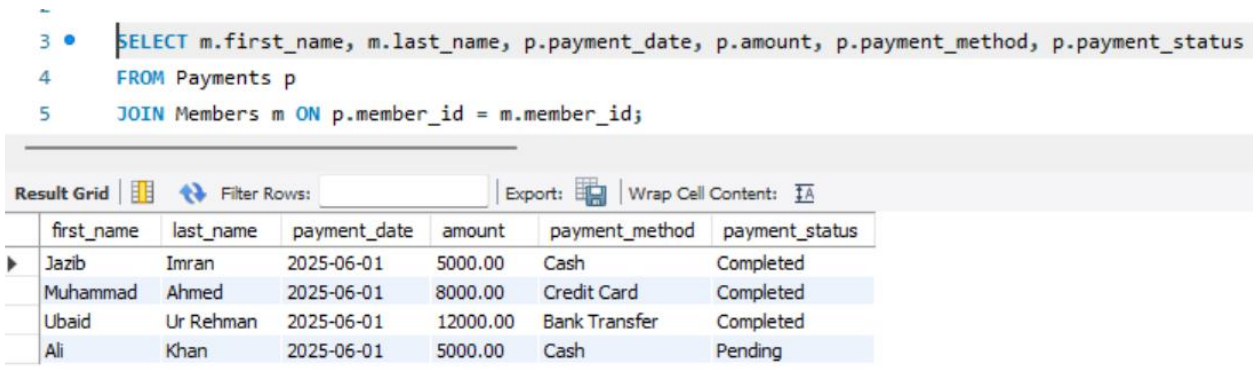


```
SELECT m.first_name, m.last_name, a.attendance_date, a.status
FROM AttendanceRecords a
JOIN Members m ON a.member_id = m.member_id
WHERE m.member_id = 1;
```

first_name	last_name	attendance_date	status
Jazib	Imran	2025-06-01	Present
Jazib	Imran	2025-06-02	Absent

This query shows the attendance records (date and status) for the member with member_id = 1. It joins the AttendanceRecords and Members tables to include the member's first and last names.

3:

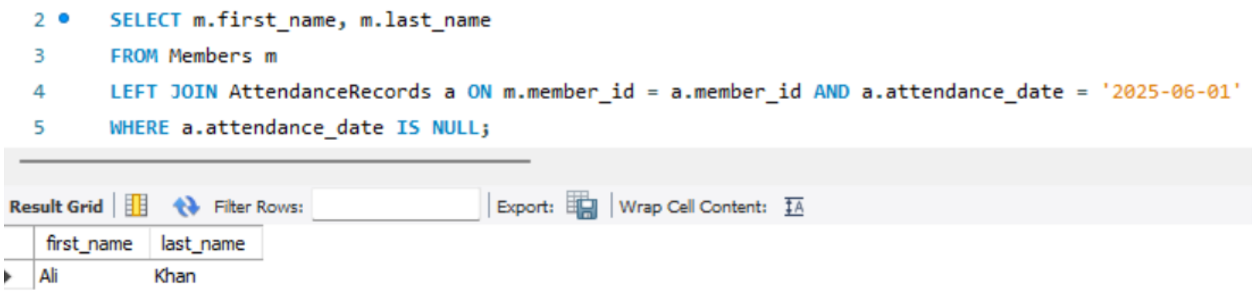


```
SELECT m.first_name, m.last_name, p.payment_date, p.amount, p.payment_method, p.payment_status
FROM Payments p
JOIN Members m ON p.member_id = m.member_id;
```

first_name	last_name	payment_date	amount	payment_method	payment_status
Jazib	Imran	2025-06-01	5000.00	Cash	Completed
Muhammad	Ahmed	2025-06-01	8000.00	Credit Card	Completed
Ubaid	Ur Rehman	2025-06-01	12000.00	Bank Transfer	Completed
Ali	Khan	2025-06-01	5000.00	Cash	Pending

This query retrieves payment details (date, amount, method, status) along with the first and last names of members. It joins the Payments and Members tables based on member_id.

4:



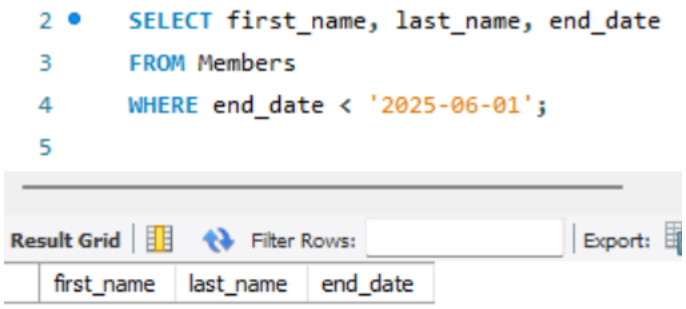
```sql
SELECT m.first_name, m.last_name
FROM Members m
LEFT JOIN AttendanceRecords a ON m.member_id = a.member_id AND a.attendance_date = '2025-06-01'
WHERE a.attendance_date IS NULL;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name
Ali	Khan

This query returns the names of members who did **not** attend the gym on **2025-06-01**. It uses a LEFT JOIN and filters records where no matching attendance entry exists for that date.

5:



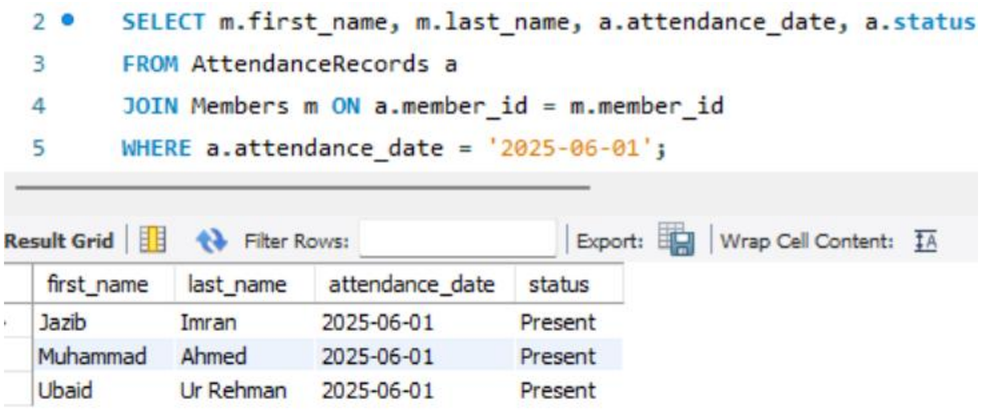
```sql
SELECT first_name, last_name, end_date
FROM Members
WHERE end_date < '2025-06-01';
```

Result Grid | Filter Rows: | Export:

first_name	last_name	end_date

This query retrieves the first name, last name, and membership end date of members whose memberships expired **before June 1, 2025**.

6:



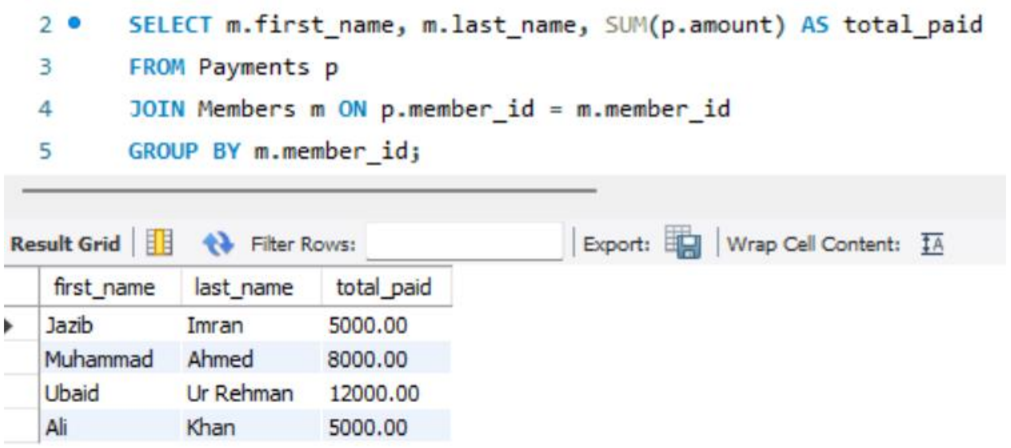
```sql
SELECT m.first_name, m.last_name, a.attendance_date, a.status
FROM AttendanceRecords a
JOIN Members m ON a.member_id = m.member_id
WHERE a.attendance_date = '2025-06-01';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	attendance_date	status
Jazib	Imran	2025-06-01	Present
Muhammad	Ahmed	2025-06-01	Present
Ubaid	Ur Rehman	2025-06-01	Present

This query displays the names of members along with their attendance status on **2025-06-01**. It joins the AttendanceRecords and Members tables based on member_id.

7:

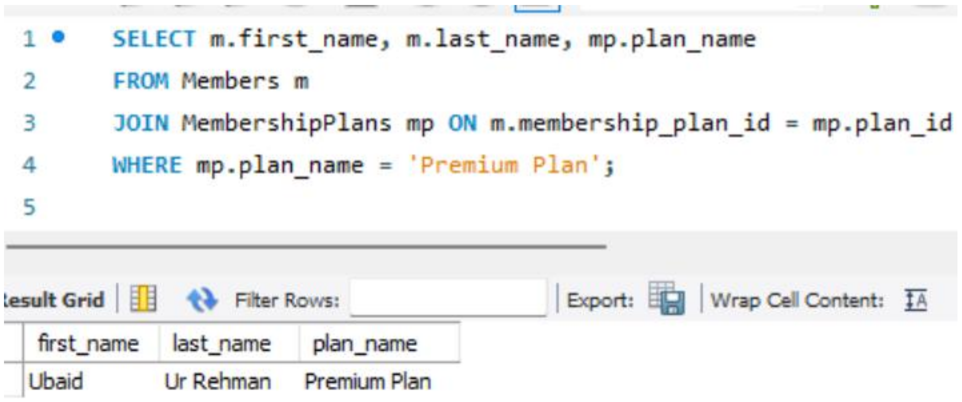


```
SELECT m.first_name, m.last_name, SUM(p.amount) AS total_paid
FROM Payments p
JOIN Members m ON p.member_id = m.member_id
GROUP BY m.member_id;
```

first_name	last_name	total_paid
Jazib	Imran	5000.00
Muhammad	Ahmed	8000.00
Ubaid	Ur Rehman	12000.00
Ali	Khan	5000.00

This query shows the total amount paid by each member by summing their payments. It joins Payments with Members and groups the results by member_id.

8:

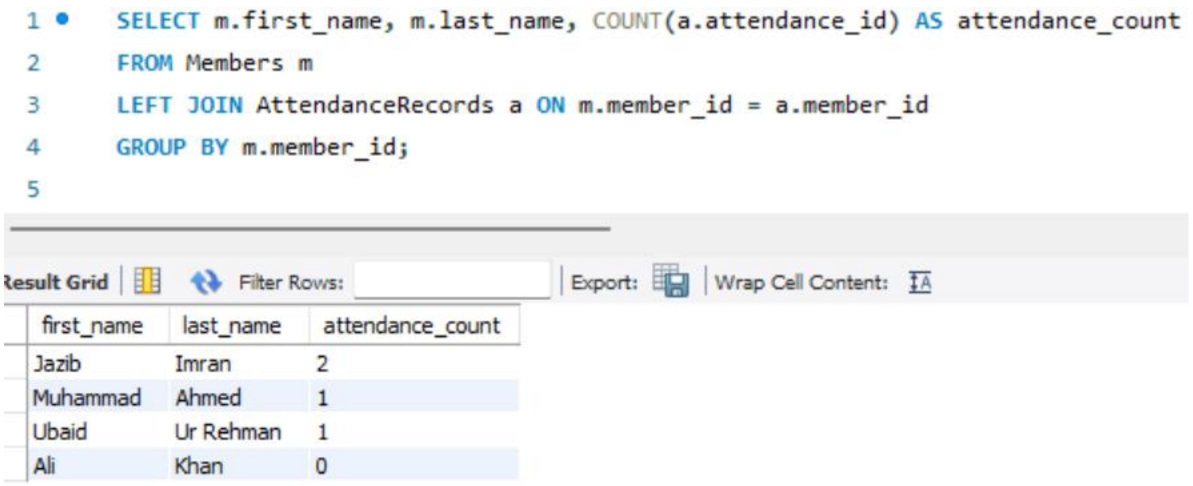


```
SELECT m.first_name, m.last_name, mp.plan_name
FROM Members m
JOIN MembershipPlans mp ON m.membership_plan_id = mp.plan_id
WHERE mp.plan_name = 'Premium Plan';
```

first_name	last_name	plan_name
Ubaid	Ur Rehman	Premium Plan

This query retrieves the names of members who are subscribed to the Premium Plan. It joins Members with MembershipPlans using the membership_plan_id.

9:

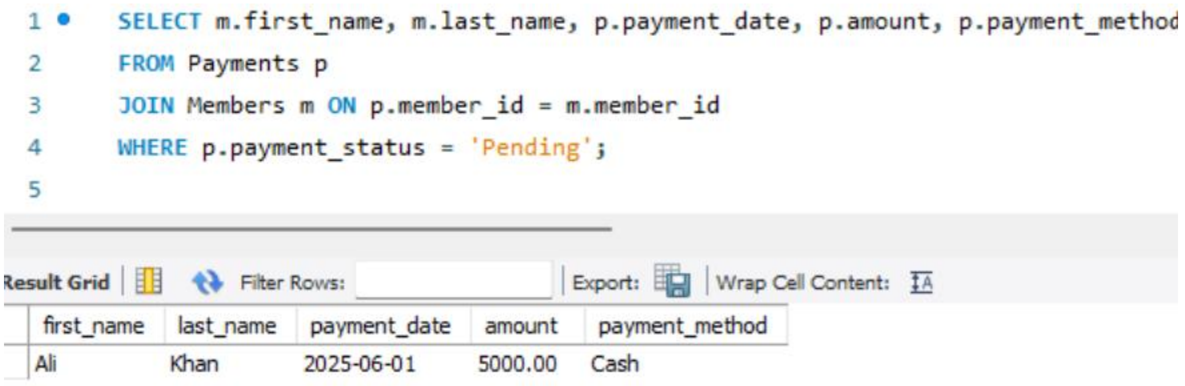


```sql
SELECT m.first_name, m.last_name, COUNT(a.attendance_id) AS attendance_count
FROM Members m
LEFT JOIN AttendanceRecords a ON m.member_id = a.member_id
GROUP BY m.member_id;
```

first_name	last_name	attendance_count
Jazib	Imran	2
Muhammad	Ahmed	1
Ubaid	Ur Rehman	1
Ali	Khan	0

This query displays each member’s name along with the total number of their attendance records. It uses a LEFT JOIN to include members even if they have zero attendance.

10:

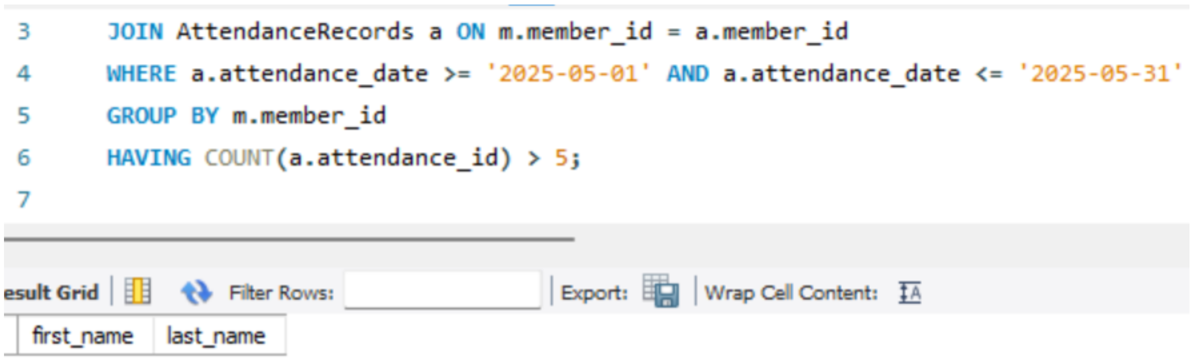


```sql
SELECT m.first_name, m.last_name, p.payment_date, p.amount, p.payment_method
FROM Payments p
JOIN Members m ON p.member_id = m.member_id
WHERE p.payment_status = 'Pending';
```

first_name	last_name	payment_date	amount	payment_method
Ali	Khan	2025-06-01	5000.00	Cash

This query shows the names of members with **pending payments**, along with payment date, amount, and method. It joins Payments with Members using member_id.

11:



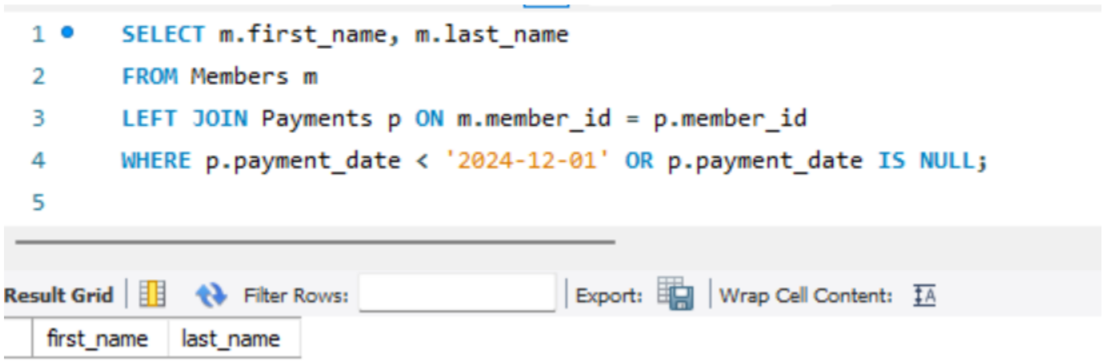
```
JOIN AttendanceRecords a ON m.member_id = a.member_id
WHERE a.attendance_date >= '2025-05-01' AND a.attendance_date <= '2025-05-31'
GROUP BY m.member_id
HAVING COUNT(a.attendance_id) > 5;
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

```
first_name  last_name
```

This query retrieves the names of members who attended the gym **more than 5 times** during **May 2025**. It filters attendance records by date, groups by member, and uses HAVING to count attendances.

12:



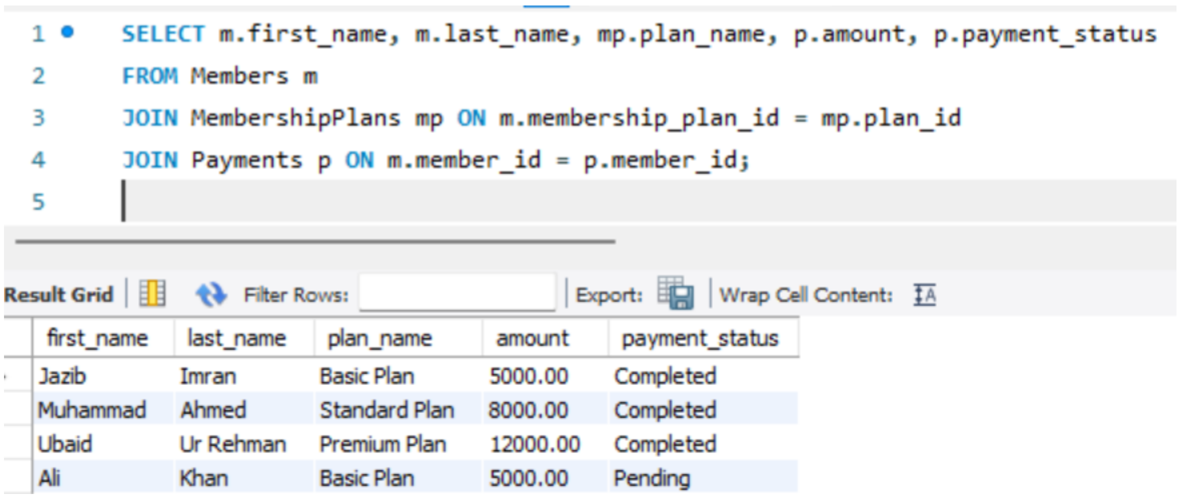
```
SELECT m.first_name, m.last_name
FROM Members m
LEFT JOIN Payments p ON m.member_id = p.member_id
WHERE p.payment_date < '2024-12-01' OR p.payment_date IS NULL;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

```
first_name  last_name
```

This query returns the names of members who have **not made a payment in the last 6 months** before **December 1, 2024**, or have **never made a payment**. It uses a LEFT JOIN and filters accordingly.

13:

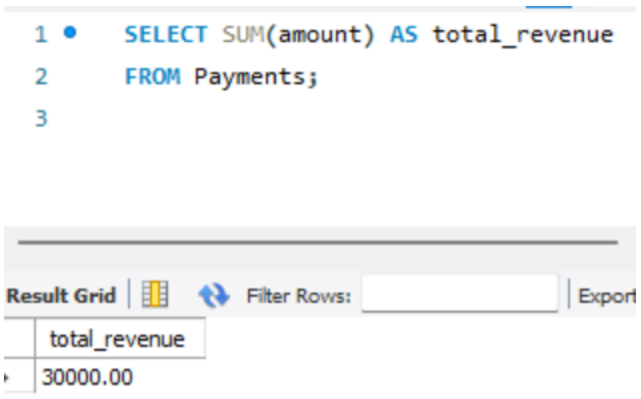


```sql
SELECT m.first_name, m.last_name, mp.plan_name, p.amount, p.payment_status
FROM Members m
JOIN MembershipPlans mp ON m.membership_plan_id = mp.plan_id
JOIN Payments p ON m.member_id = p.member_id;
```

first_name	last_name	plan_name	amount	payment_status
Jazib	Imran	Basic Plan	5000.00	Completed
Muhammad	Ahmed	Standard Plan	8000.00	Completed
Ubaid	Ur Rehman	Premium Plan	12000.00	Completed
Ali	Khan	Basic Plan	5000.00	Pending

This query displays members' names, their membership plan name, payment amount, and payment status. It joins Members with both MembershipPlans and Payments using membership_plan_id and member_id.

14:

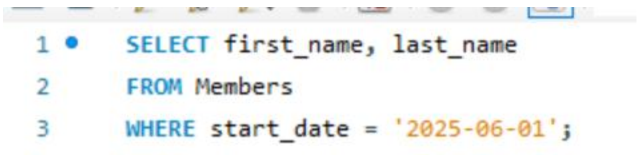


```sql
SELECT SUM(amount) AS total_revenue
FROM Payments;
```

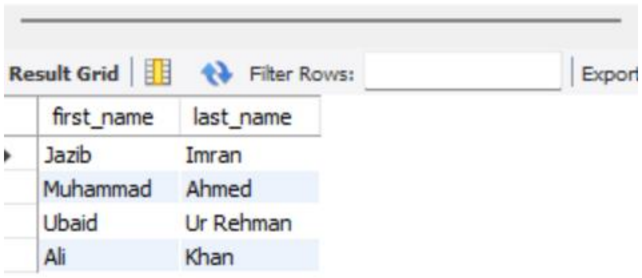
total_revenue
30000.00

This query calculates the **total revenue** generated from all payments by summing the amount column in the Payments table.

15:



```sql
SELECT first_name, last_name
FROM Members
WHERE start_date = '2025-06-01';
```



first_name	last_name
Jazib	Imran
Muhammad	Ahmed
Ubaid	Ur Rehman
Ali	Khan

This query retrieves the names of members who **joined the gym on June 1, 2025**, by filtering the Members table using the start_date.

Challenges Faced:

- **Foreign Key Relationships:**
  Ensuring referential integrity across attendance, payments, and membership plans required careful ordering of inserts.
- **Query Design:**
  Analytical queries such as identifying absent members or filtering premium users demanded precision with joins and date conditions.
- **Constraint Validation:**
  Ensuring data like email uniqueness and proper ENUM values were validated effectively.
- **Sample Data Consistency:**
  Sample data required coordination across related tables to avoid FK violations during insertion.

Conclusion:

The development of the Gym Management System database represents a robust, functional, and scalable solution for efficiently managing the day-to-day operations of a modern fitness facility. Through meticulous planning, normalization, and implementation of best practices in database design, this system has successfully digitized critical gym activities such as membership tracking, attendance logging, and payment processing.

- By replacing manual and error-prone methods with an automated relational database, the project has significantly improved data accuracy, response time, and operational reliability. The system enforces referential integrity through primary and foreign key relationships, minimizes redundancy via normalization, and uses constraints to maintain consistent and meaningful data across all entities.

- In addition, the Gym Management System offers powerful analytical capabilities that support better decision-making. For example, administrators can quickly identify members with expired or soon-to-expire plans, monitor attendance rates over time, and detect revenue trends. These insights help in designing promotional offers, retention strategies, and resource planning.

- Furthermore, the modular and well-documented design ensures that the system can be extended or integrated with other technologies in the future. This could include integration with a mobile application for member check-ins, an online payment gateway, or a web-based dashboard for trainers and admins.

- In essence, the project showcases the importance of using structured databases to transform routine business operations into streamlined, intelligent workflows that are both maintainable and scalable.