Established – 1961 Subject: OSDBMS

SEVA SADAN'S

R. K. TALREJA COLLEGE

OF

ARTS, SCIENCE & COMMERCE ULHASNAGAR – 421 003

CERTIFICATE

This is to certify that Mr./Ms. Jay Santosh Gajakosh of S.Y. Information Technology (SYIT) Roll No. 2542007 has satisfactorily completed the Open Source DataBase Management System Mini Project entitled Scholarship Eligibility & Disbursement Database during the academic year 2025 – 2026, as a part of the practical requirement. The project work is found to be satisfactory and is approved for submission.

PROF. INCHARGE **HEAD OF DEPT**

__________________ ______________________

INDEX

CHAPTER 1: INTRODUCTION

The Scholarship Eligibility & Disbursement Database is designed to manage student scholarship information in a structured and efficient manner using MySQL. In educational institutions and government scholarship departments, large volumes of student data, scholarship schemes, eligibility criteria, and payment records must be maintained accurately. Manual or semi-digital systems such as paper records or spreadsheets often lead to errors, data duplication, and difficulty in tracking scholarship distribution. The developed database system provides a centralized platform where all scholarship-related information is stored, managed, and processed systematically. The system is built using relational database concepts where data is stored in tables and linked using relationships. This ensures better data organization, faster data retrieval, and improved decision-making. The given SQL code creates a database named ScholarshipDB and includes tables such as Students, Scholarships, Eligibility, and Disbursement, which together form the complete scholarship management system.

A database system is an organized collection of data that can be easily accessed, managed, and updated. In the given project, the database system stores student personal details, scholarship scheme details, eligibility verification results, and scholarship payment information. Each table in the database represents a real-world entity. For example, the Students table stores student information such as name, course, and family income. The Scholarships table stores scholarship criteria such as minimum percentage and maximum income limit. The Eligibility table links students with scholarships and records whether the student is eligible or not. The Disbursement table stores payment transaction details. The database uses Primary Keys to uniquely identify each record and Foreign Keys to maintain relationships between tables. This ensures data integrity and prevents invalid or duplicate records. SQL queries are used to insert data, retrieve data, and generate reports using JOIN operations.

The main problem addressed by this database system is the inefficiency of manual scholarship management systems. In traditional systems, scholarship records are stored in files or spreadsheets, which makes searching and updating data difficult. Manual systems also increase the risk of human errors such as wrong eligibility calculation or duplicate payment entries. Lack of proper data linking makes it difficult to track student scholarship history. The database system solves these problems by providing centralized storage, automatic relationship management, and fast data retrieval. It improves transparency because eligibility status and payment status can be tracked easily. It also improves security because database access can be controlled using user privileges.

MySQL is used in this project because it is an open-source relational database management system that is reliable, scalable, and widely used in industry and academics. MySQL supports structured data storage, powerful query processing, and strong security features. It also supports constraints such as PRIMARY KEY, FOREIGN KEY, NOT NULL, and UNIQUE, which help maintain data accuracy. MySQL Workbench provides an easy graphical interface for database development and query execution. Since MySQL is free and supported by a large community, it is suitable for student projects and real-world applications. Therefore, MySQL is an ideal choice for implementing the Scholarship Eligibility & Disbursement Database system.

CHAPTER 2: PROBLEM DEFINITION

The Scholarship Eligibility & Disbursement process in many institutions is still managed using manual records, Excel sheets, or disconnected software systems. This creates difficulty in managing large student datasets, scholarship rules, eligibility verification, and payment tracking. The absence of a centralized database leads to data inconsistency, duplication, and security risks. The Scholarship Eligibility & Disbursement Database is designed to solve these issues by providing a structured, secure, and relational database system using MySQL.

2.1 Existing Manual / Unstructured System

- Student scholarship records are stored in paper files or spreadsheets
- Scholarship schemes are maintained separately from student records
- Eligibility checking is done manually based on marks and income criteria
- Payment records are maintained in separate accounting files
- Difficult to track scholarship history of a student
- High chances of data loss due to physical damage or file corruption
- Time-consuming process to verify and approve scholarships
- No centralized system to connect students, scholarships, and payments

2.2 Issues in Existing System

- **Data Redundancy**
  - Same student data entered multiple times in different files ○ Duplicate scholarship records may occur
- **Data Inconsistency**
  - Student data mismatch across different records ○ Scholarship eligibility may be incorrectly calculated
- **Security Problems**
  - Unauthorized access to student financial data ○ No proper authentication or access

control ○ Risk of data leakage or accidental

deletion

- **Manual Errors**

  ○ Wrong eligibility status marking ○ Incorrect

  scholarship amount disbursement ○ Missing

  payment tracking

2.3 Need for a Database-Driven Solution

- To create a centralized scholarship data management system
- To reduce manual work and improve efficiency
- To maintain accurate student and scholarship records
- To ensure proper eligibility verification using structured data
- To track scholarship payments systematically
- To improve data security using MySQL user privileges and access control
- To support fast data retrieval using SQL queries
- To maintain data integrity using Primary Key and Foreign Key constraints
- To generate reports for administration and auditing purposes

CHAPTER 3: OBJECTIVES OF THE PROJECT

The main objective of the Scholarship Eligibility & Disbursement Database project is to design and implement a structured and efficient database system using MySQL that can manage scholarship-related data accurately. In many educational institutions, scholarship data is handled manually or using spreadsheets, which leads to errors, duplication, and difficulty in tracking records. This project aims to solve these problems by creating a centralized database system that stores student information, scholarship scheme details, eligibility verification data, and scholarship payment records in an organized manner. The project focuses on using relational database concepts to ensure proper data linking between different entities involved in scholarship processing. By implementing this database system, institutions can reduce manual workload, improve data accuracy, and ensure faster processing of scholarship applications and payments.

One of the key objectives of this project is to design a structured database system. This includes creating tables such as Students, Scholarships, Eligibility, and Disbursement based on real-world scholarship management requirements. Each table is designed with appropriate attributes and data types to store relevant information. The database design ensures normalization, which reduces data redundancy and improves data consistency. Relationships between tables are maintained using Primary Keys and Foreign Keys. For example, student_id is used to uniquely identify each student, while scholarship_id identifies each scholarship scheme. These keys help in maintaining proper relationships between students and scholarships. The structured design also helps in improving query performance and data retrieval speed.

Another important objective is to implement SQL queries for data management. The project uses SQL commands such as CREATE DATABASE, CREATE TABLE, INSERT INTO, and SELECT to manage database operations. The system also supports data retrieval using JOIN queries, which helps in generating reports such as student scholarship payment status. SQL queries make the system flexible and allow administrators to easily add, update, delete, and retrieve data. This helps in maintaining real-time scholarship records and ensures smooth system operation.

The project also focuses on ensuring data integrity using database constraints. Constraints such as PRIMARY KEY, FOREIGN KEY, NOT NULL, and UNIQUE are used to maintain data accuracy and consistency. For example, the UNIQUE constraint on student email ensures that duplicate student records are not created. Foreign Keys ensure that eligibility or payment records cannot exist without valid student and scholarship records. This prevents data inconsistency and maintains database reliability.

Finally, the project helps in gaining practical hands-on experience with MySQL. Students and developers working on this project learn how to design relational databases, create tables, define constraints, insert data, and perform complex queries. This knowledge is important for real-world database development and management. The project also helps in understanding how database systems are used in educational institutions and government organizations. Overall, this project improves technical skills while solving real-world scholarship management problems efficiently.

CHAPTER 4: SCOPE OF THE PROJECT

The scope of the Scholarship Eligibility & Disbursement Database project focuses on providing a structured and centralized system for managing scholarship-related information using MySQL. The system is designed to store and manage student details, scholarship scheme information, eligibility verification status, and scholarship payment records in an organized database structure. The project can be used in educational institutions, government scholarship departments, and private organizations that provide financial assistance to students. By implementing this database system, organizations can improve efficiency, reduce manual errors, and maintain accurate scholarship records. The system is scalable, meaning more tables, fields, or modules can be added in the future based on requirements. The project mainly focuses on database design and SQL implementation, making it suitable for academic learning as well as basic real-world implementation.

4.1 Where the System Can Be Used

The Scholarship Eligibility & Disbursement Database system can be used in multiple sectors where scholarship or financial aid management is required. Educational institutions such as colleges and universities can use the system to manage student scholarship data. Government scholarship portals can use similar database structures to manage applications, eligibility checks, and payment disbursements. Training institutes and private organizations that provide educational grants can also implement this system. The system helps in maintaining centralized student records, scholarship rules, and payment tracking. It can also be integrated with web-based portals in the future to provide online scholarship application and tracking features. Since the system is developed using MySQL, it can be easily deployed on local servers or cloud-based database servers.

4.2 Users of the System

The system can be used by multiple types of users based on their roles and responsibilities:

- **Admin**
  - Manages scholarship schemes
  - Updates eligibility rules
  - Monitors scholarship disbursement records
  - Generates reports
- **Staff / Scholarship Department**
  - Enters student data
  - Verifies eligibility
  - Updates payment status
- **Students**

○ Can check scholarship eligibility status (if integrated with application system) ○ Can view payment details

- **Business / Financial Users** ○ Monitor fund allocation ○ Track scholarship payment reports ○ Perform auditing and financial analysis

4.3 Academic Limitations

Since this project is developed for academic purposes, it has certain limitations. The system mainly focuses on database design and SQL implementation and does not include a front-end user interface. Real-world scholarship systems may require web or mobile application integration, which is outside the scope of this project. The system uses sample data and simplified eligibility rules for demonstration. Advanced features such as real-time payment gateway integration, automated eligibility verification using AI, and multi-level approval workflows are not included. Security implementation is limited to database-level constraints and does not include advanced cybersecurity mechanisms. Despite these limitations, the project provides strong foundational knowledge of database design, relational models, and SQL query implementation.

Subdomains of Application

- **Educational Use** – Used by colleges and universities to manage student scholarships
- **Business Use** – Used by organizations to manage funding and financial assistance programs
- **Administrative Use** – Used by government departments to manage large-scale scholarship schemes

CHAPTER 5: REQUIREMENT SPECIFICATION

Hardware Requirements:

• Computer/Laptop

• Minimum 4GB RAM

• 20GB free storage

Software Requirements:

• MySQL Server

• MySQL Workbench

• Windows/Linux OS

• SQL language

CHAPTER 6: SYSTEM DESIGN

The System Design of the Scholarship Eligibility & Disbursement Database focuses on how data flows within the database and how users interact with the MySQL database system. The design is conceptual and explains the working structure of the database rather than hardware implementation. The system is designed using relational database concepts where multiple tables are connected using keys. The database ensures smooth data storage, retrieval, and update operations using SQL queries. The system design is based on the given database code which includes Students, Scholarships, Eligibility, and Disbursement tables. Each table is connected logically to maintain proper scholarship processing workflow. The system allows administrators and staff to insert student data, verify eligibility, and manage scholarship payments efficiently.

6.1 System Architecture (Conceptual)

The conceptual architecture of the Scholarship Eligibility & Disbursement Database explains how users interact with the database and how MySQL processes user requests. The system follows a database-centric architecture where users interact with the database using SQL queries through MySQL Workbench or other database tools. The user first connects to the MySQL server using authentication credentials. After successful login, the user can perform database operations such as inserting student data, updating scholarship schemes, verifying eligibility, and managing payment records.

When a user enters a query, the MySQL server processes the query by checking syntax correctness and user privileges. If the query is valid and the user has permission, the server executes the query and performs the requested operation. For example, when a staff member inserts student data using the INSERT query, the MySQL server stores the data in the Students table. When eligibility is checked, the system uses data from both Students and Scholarships tables and stores results in the Eligibility table. Similarly, payment details are stored in the Disbursement table.

The MySQL server plays an important role in managing data storage, query processing, transaction management, and security enforcement. It ensures that multiple users can access the database simultaneously without data conflict. It also ensures data integrity using constraints such as Primary Key and Foreign Key. The query execution flow starts from user input, then moves to query parsing, optimization, execution, and finally result output. The result is displayed to the user in tabular format.

CHAPTER 7: DATABASE DESIGN

The Database Design of the Scholarship Eligibility & Disbursement Database focuses on structuring data into well-organized tables based on relational database principles. The database is designed using MySQL and follows normalization concepts to reduce data redundancy and maintain data consistency. The design is based on real-world scholarship management requirements where student details, scholarship schemes, eligibility verification, and scholarship payment details must be stored and linked properly. The database contains four main tables: Students, Scholarships, Eligibility, and Disbursement. These tables are connected using Primary Keys and Foreign Keys, which ensure proper relationships and data integrity. The database design helps in efficient data storage, fast retrieval of information, and accurate scholarship record management. By using structured table design and constraints, the system prevents duplicate entries and maintains data accuracy.

7.1 Entity Description

The **Students** table stores basic information about students who are applying for scholarships. It includes details such as student name, gender, date of birth, email, phone number, course, and family income. The family income field is important because many scholarships depend on income eligibility criteria. Each student is uniquely identified using student_id, which acts as the Primary Key.

The **Scholarships** table stores information about different scholarship schemes available. It includes scholarship name, scholarship provider, minimum percentage required, maximum income limit, and scholarship amount. Each scholarship scheme is uniquely identified using scholarship_id. This table helps administrators manage different scholarship programs easily.

The **Eligibility** table is used to check and store whether a student is eligible for a particular scholarship. It connects Students and Scholarships tables using student_id and scholarship_id as Foreign Keys. It also stores eligibility status and verification date. This table helps in tracking which student qualifies for which scholarship.

The **Disbursement** table stores scholarship payment details. It records student ID, scholarship ID, payment amount, payment date, and payment status. This table helps in tracking scholarship payments and ensures transparency in fund distribution.

7.2 Table Structure

Students Table

- Table Name: Students
- Attributes: student_id, full_name, gender, date_of_birth, email, phone, course, family_income

- Data Types: INT, VARCHAR, DATE, DECIMAL

Scholarships Table

- Table Name: Scholarships
- Attributes: scholarship_id, scholarship_name, provider, min_percentage, max_income_limit, scholarship_amount
- Data Types: INT, VARCHAR, DECIMAL

Eligibility Table

- Table Name: Eligibility
- Attributes: eligibility_id, student_id, scholarship_id, eligibility_status, checked_date
- Data Types: INT, VARCHAR, DATE

Disbursement Table

- Table Name: Disbursement
- Attributes: disbursement_id, student_id, scholarship_id, amount_paid, payment_date, payment_status

  Data Types: INT, DECIMAL, DATE, VARCHAR

7.3 Constraints Used

PRIMARY KEY

- Used to uniquely identify each record in a table
- Example: student_id in Students table
- Prevents duplicate records

FOREIGN KEY

- Used to create relationships between tables
- Example: student_id in Eligibility table references Students table
- Ensures valid reference data

NOT NULL

- Ensures mandatory fields are not left empty

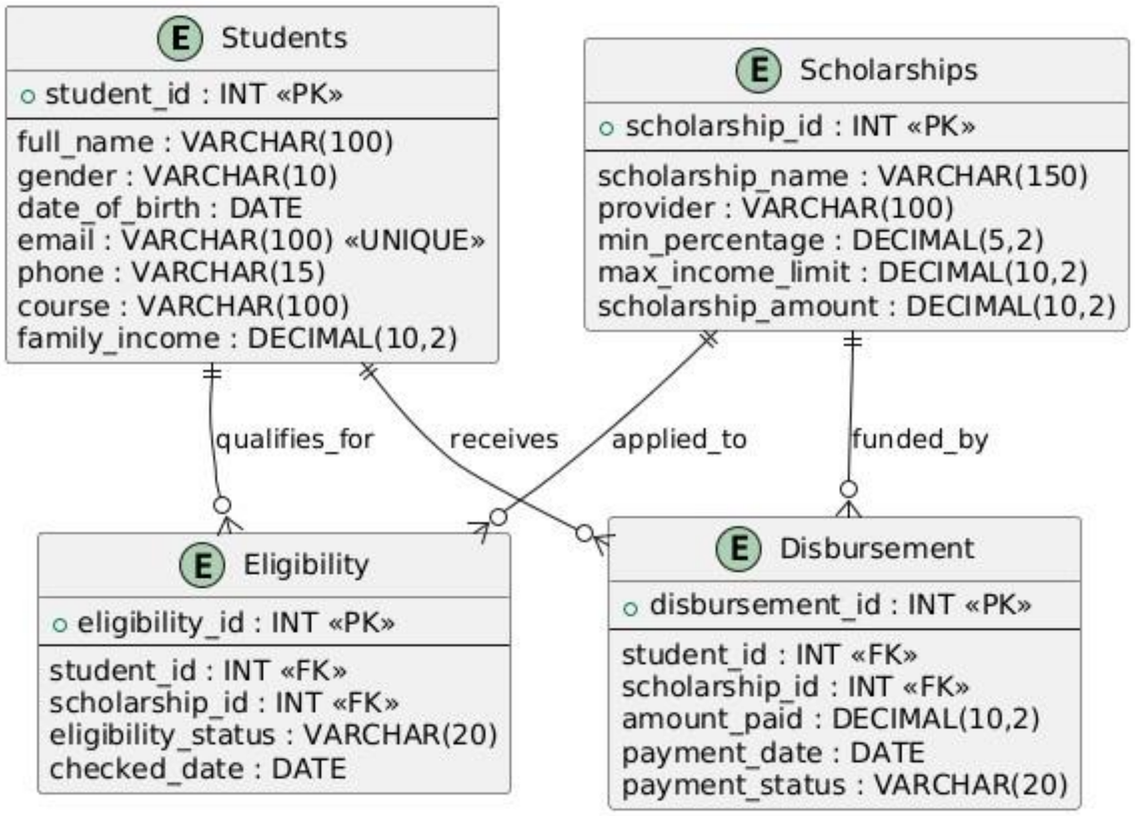
- Example: full_name cannot be NULL

UNIQUE

- Prevents duplicate values in specific columns
- Example: email field in Students table must be unique

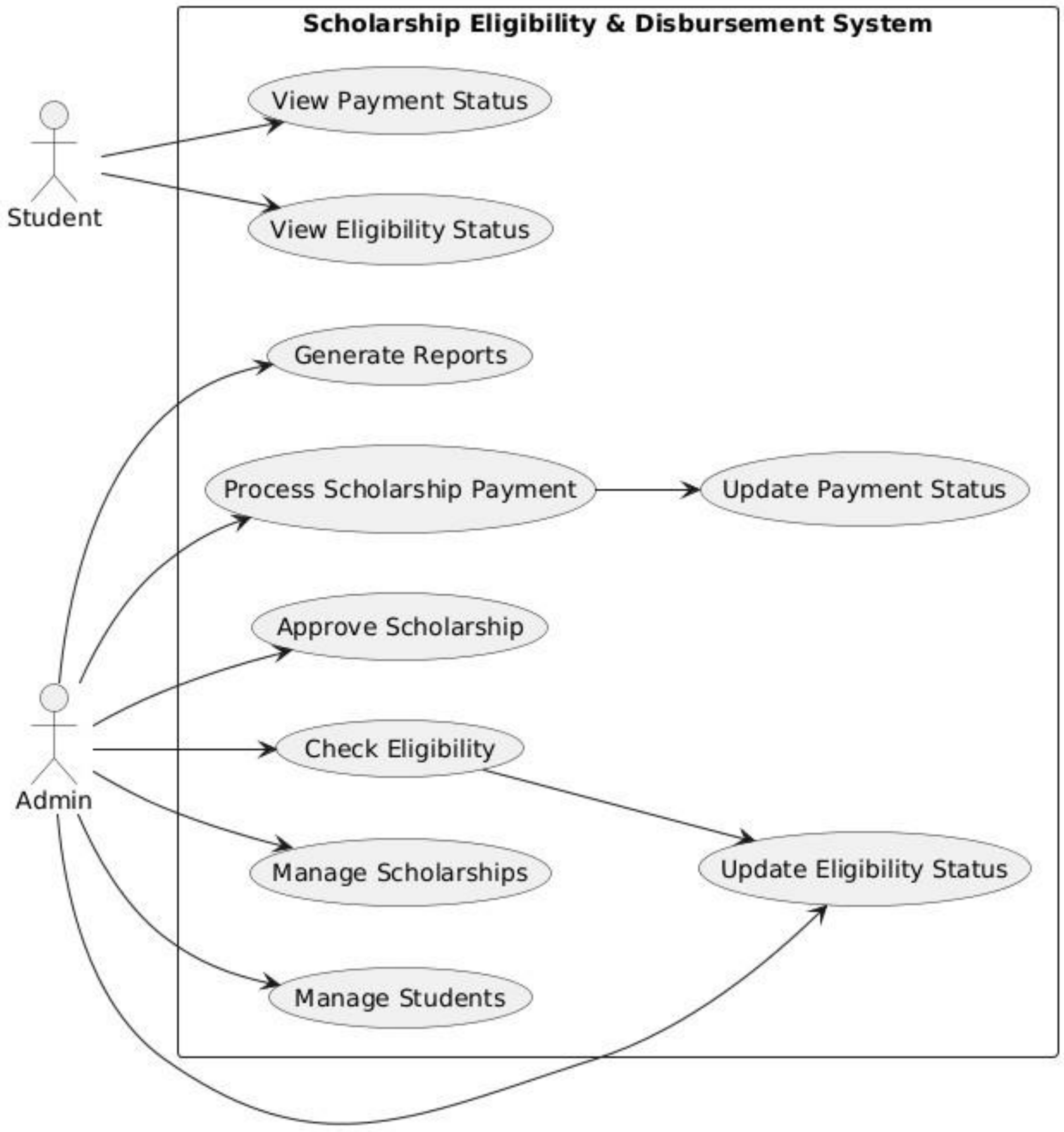
8. UML DIAGRAMS

8.1 ER Diagram

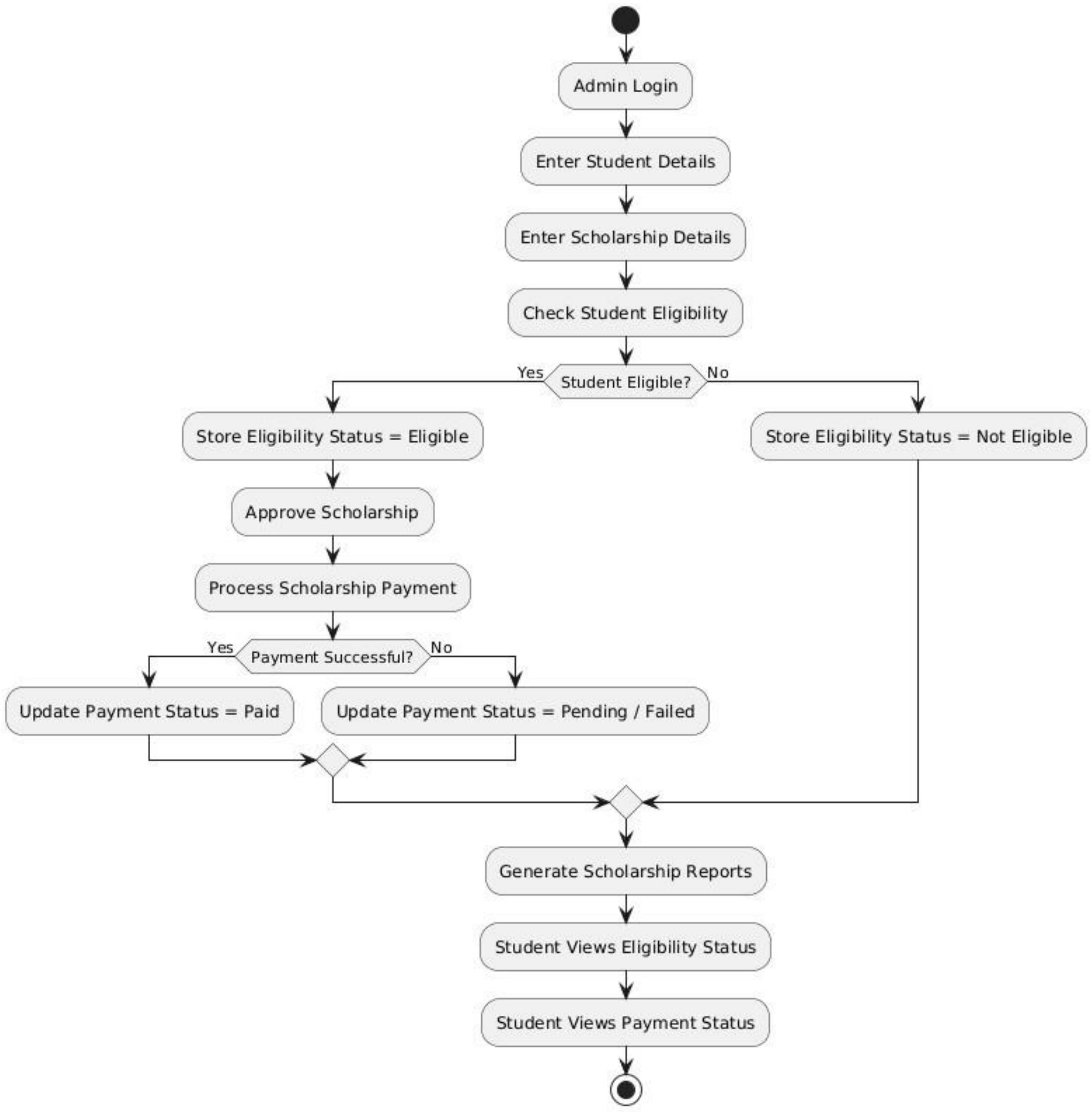


Students
student_id : INT «PK»
full_name : VARCHAR(100)
gender : VARCHAR(10)
date_of_birth : DATE
email : VARCHAR(100) «UNIQUE»
phone : VARCHAR(15)
course : VARCHAR(100)
family_income : DECIMAL(10,2)
Scholarships
scholarship_id : INT «PK»
scholarship_name : VARCHAR(150)
provider : VARCHAR(100)
min_percentage : DECIMAL(5,2)
max_income_limit : DECIMAL(10,2)
scholarship_amount : DECIMAL(10,2)
qualifies_for
receives
applied_to
funded_by
Eligibility
eligibility_id : INT «PK»
student_id : INT «FK»
scholarship_id : INT «FK»
eligibility_status : VARCHAR(20)
checked_date : DATE
Disbursement
disbursement_id : INT «PK»
student_id : INT «FK»
scholarship_id : INT «FK»
amount_paid : DECIMAL(10,2)
payment_date : DATE
payment_status : VARCHAR(20)

8.2 Use Case Diagram

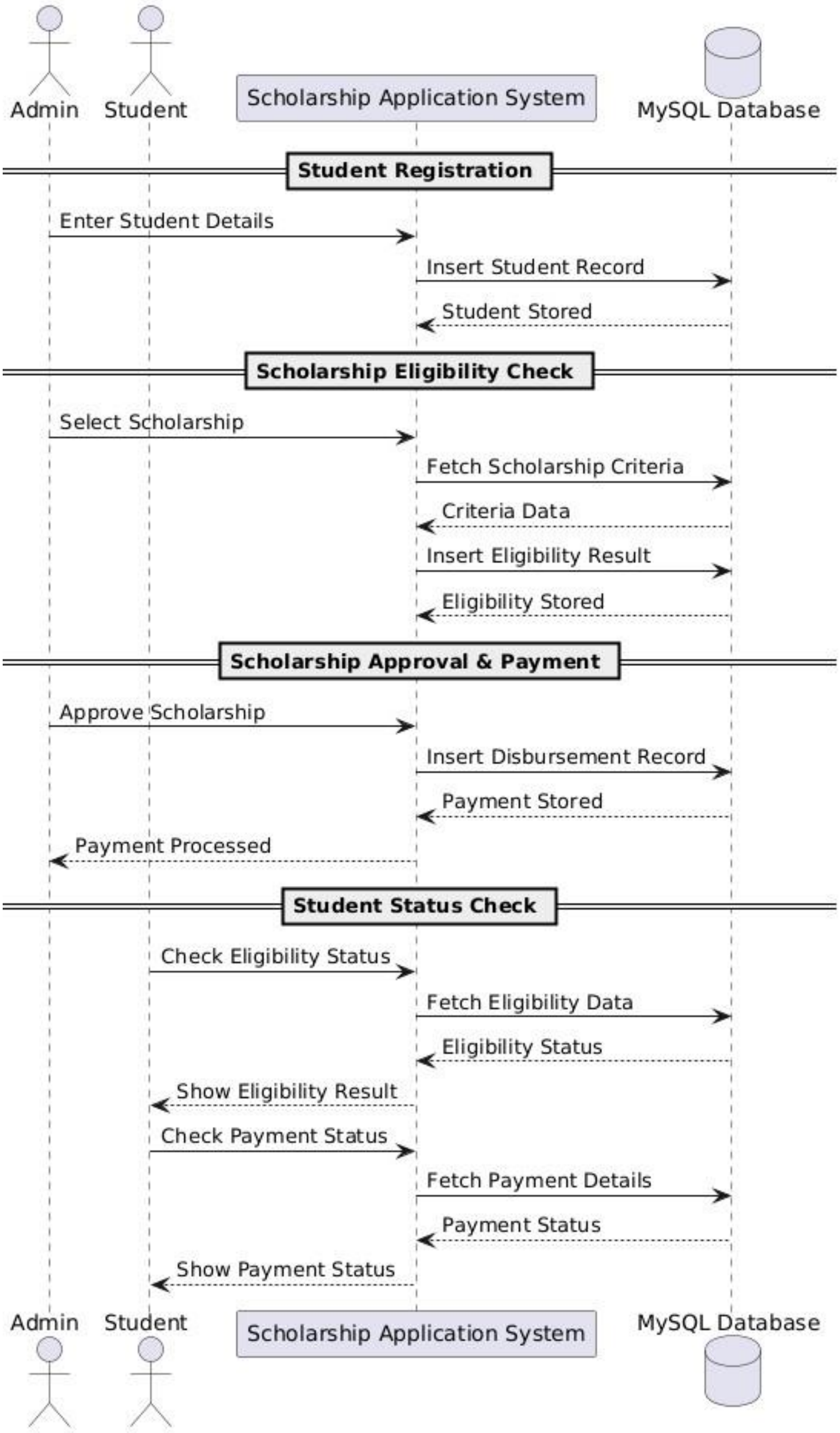


8.3 Activity Diagram



Admin Login
Enter Student Details
Enter Scholarship Details
Check Student Eligibility
Yes
Student Eligible?
No
Store Eligibility Status = Eligible
Store Eligibility Status = Not Eligible
Approve Scholarship
Process Scholarship Payment
Yes
Payment Successful?
No
Update Payment Status = Paid
Update Payment Status = Pending / Failed
Generate Scholarship Reports
Student Views Eligibility Status
Student Views Payment Status

8.4 Sequence Diagram


Admin
Student
Scholarship Application System
MySQL Database
Student Registration
Enter Student Details
Insert Student Record
Student Stored
Scholarship Eligibility Check
Select Scholarship
Fetch Scholarship Criteria
Criteria Data
Insert Eligibility Result
Eligibility Stored
Scholarship Approval & Payment
Approve Scholarship
Insert Disbursement Record
Payment Stored
Payment Processed
Student Status Check
Check Eligibility Status
Fetch Eligibility Data
Eligibility Status
Show Eligibility Result
Check Payment Status
Fetch Payment Details
Payment Status
Show Payment Status
Admin
Student
Scholarship Application System
MySQL Database

CHAPTER 9: SQL IMPLEMENTATION

9.1 Database Creation

```
CREATE DATABASE ScholarshipDB; USE
ScholarshipDB;
```

9.2 Table Creation (CREATE TABLE Queries)

```
CREATE TABLE Students (
  student_id INT PRIMARY KEY AUTO_INCREMENT,
full_name VARCHAR(100) NOT NULL,    gender
VARCHAR(10),    date_of_birth DATE,    email
VARCHAR(100) UNIQUE,    phone VARCHAR(15),
course VARCHAR(100),
  family_income DECIMAL(10,2)
);

CREATE TABLE Scholarships (
  scholarship_id INT PRIMARY KEY AUTO_INCREMENT,
scholarship_name VARCHAR(150) NOT NULL,
  provider VARCHAR(100),
min_percentage DECIMAL(5,2),
max_income_limit DECIMAL(10,2),
  scholarship_amount DECIMAL(10,2)
);

CREATE TABLE Eligibility (
  eligibility_id INT PRIMARY KEY AUTO_INCREMENT,
student_id INT,    scholarship_id INT,
eligibility_status VARCHAR(20),
  checked_date DATE,
  FOREIGN KEY (student_id) REFERENCES Students(student_id),
  FOREIGN KEY (scholarship_id) REFERENCES Scholarships(scholarship_id)
);

CREATE TABLE Disbursement (
  disbursement_id INT PRIMARY KEY AUTO_INCREMENT,
student_id INT,
  scholarship_id INT,
amount_paid DECIMAL(10,2),
payment_date DATE,
payment_status VARCHAR(20),
  FOREIGN KEY (student_id) REFERENCES Students(student_id),
  FOREIGN KEY (scholarship_id) REFERENCES Scholarships(scholarship_id)
);
```

9.3 Data Insertion (INSERT Queries)

```
INSERT INTO Students(full_name, gender, date_of_birth, email, phone, course, family_income) VALUES
('Rahul Sharma', 'Male', '2003-05-10', 'rahul@email.com', '9876543210', 'BSc IT', 180000),
('Sneha Patil', 'Female', '2002-08-15', 'sneha@email.com', '9876501234', 'BCom', 120000);
```

```
INSERT INTO Scholarships(scholarship_name, provider, min_percentage, max_income_limit, scholarship_amount) VALUES
('Merit Scholarship', 'State Govt', 75, 300000, 25000),
('Need Based Scholarship', 'Central Govt', 60, 200000, 30000);

INSERT INTO Eligibility(student_id, scholarship_id, eligibility_status, checked_date) VALUES
(1, 1, 'Eligible', CURDATE()),
(2, 2, 'Eligible', CURDATE());

INSERT INTO Disbursement(student_id, scholarship_id, amount_paid, payment_date, payment_status)
VALUES
(1, 1, 25000, CURDATE(), 'Paid'), (2,
2, 30000, CURDATE(), 'Paid');
```

9.4 Data Retrieval (SELECT, WHERE, JOIN)

- **Simple SELECT**

```
SELECT * FROM Students;
```

- **SELECT with WHERE**

```
SELECT full_name, family_income
FROM Students
WHERE family_income < 200000;
```

- **JOIN Query**

```
SELECT s.full_name, sc.scholarship_name, d.amount_paid, d.payment_status
FROM Students s
JOIN Disbursement d ON s.student_id = d.student_id
JOIN Scholarships sc ON sc.scholarship_id = d.scholarship_id;
```

9.5 Advanced Queries

◈ **GROUP BY**

Total amount paid to each student:

```
SELECT student_id, SUM(amount_paid) AS total_amount
FROM Disbursement
GROUP BY student_id;
```

◈ **HAVING**

Students who received more than 25000:

```
SELECT student_id, SUM(amount_paid) AS total_amount
FROM Disbursement
GROUP BY student_id
HAVING SUM(amount_paid) > 25000;
```

◈ Subquery

Students whose income is less than the maximum income limit of Merit Scholarship:

```
SELECT full_name
FROM Students
WHERE family_income <
(
   SELECT max_income_limit
   FROM Scholarships
   WHERE scholarship_name = 'Merit Scholarship'
);
```

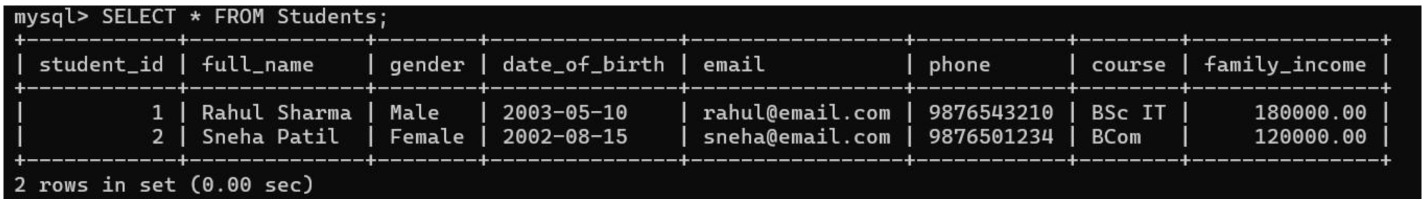
◈ View

Create a view for paid scholarships:

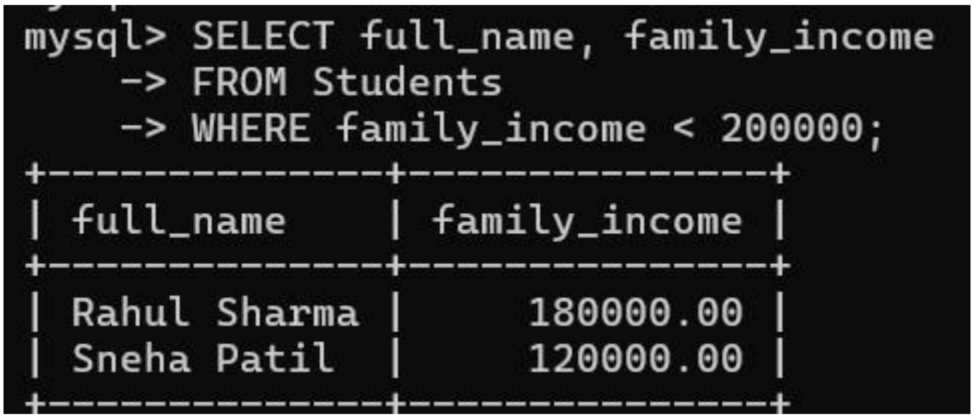
```
CREATE VIEW Paid_Scholarships AS
SELECT s.full_name, sc.scholarship_name, d.amount_paid
FROM Students s
JOIN Disbursement d ON s.student_id = d.student_id
JOIN Scholarships sc ON sc.scholarship_id = d.scholarship_id
WHERE d.payment_status = 'Paid'; To use the view:
```

```
SELECT * FROM Paid_Scholarships;
```

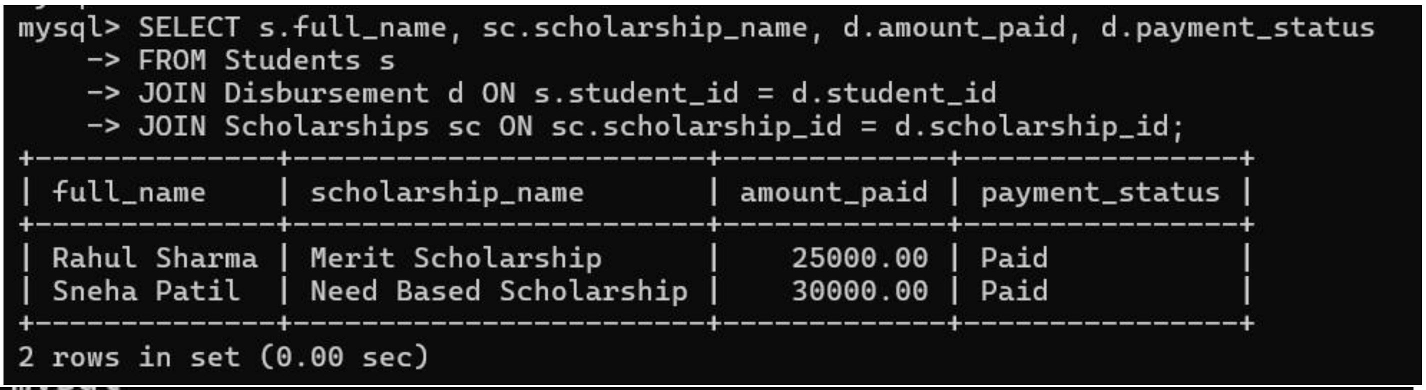
CHAPTER 10: SYSTEM TESTING AND RESULT



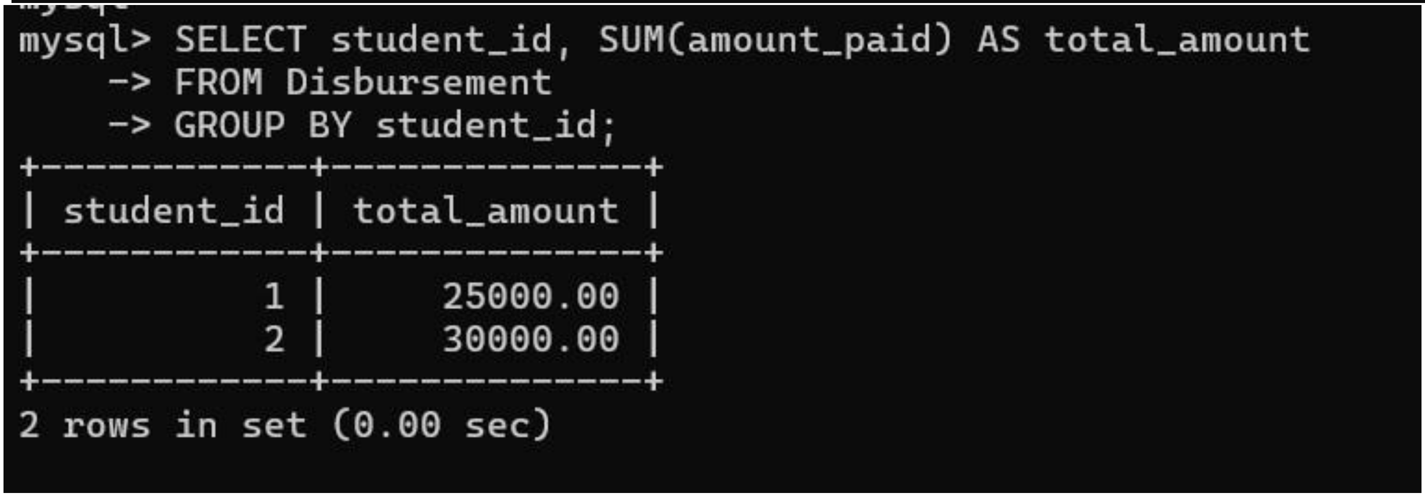
```
mysql> SELECT * FROM Students;
+------------+--------------+--------+---------------+-----------------+------------+--------+---------------+
| student_id | full_name    | gender | date_of_birth | email           | phone      | course | family_income |
+------------+--------------+--------+---------------+-----------------+------------+--------+---------------+
|          1 | Rahul Sharma | Male   | 2003-05-10    | rahul@email.com | 9876543210 | BSc IT |     180000.00 |
|          2 | Sneha Patil  | Female | 2002-08-15    | sneha@email.com | 9876501234 | BCom   |     120000.00 |
+------------+--------------+--------+---------------+-----------------+------------+--------+---------------+
2 rows in set (0.00 sec)
```



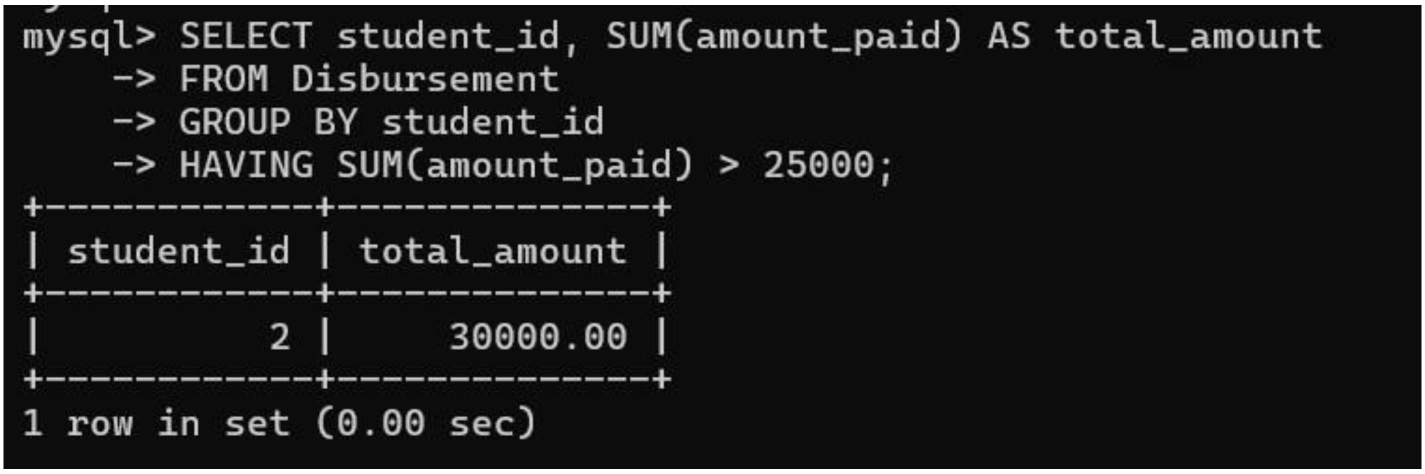
```
mysql> SELECT full_name, family_income
    -> FROM Students
    -> WHERE family_income < 200000;
+--------------+---------------+
| full_name    | family_income |
+--------------+---------------+
| Rahul Sharma |     180000.00 |
| Sneha Patil  |     120000.00 |
+--------------+---------------+
```



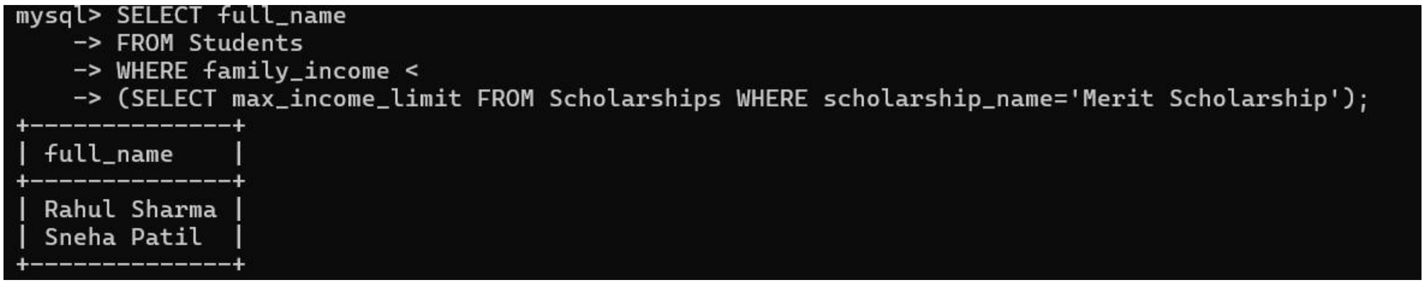
```
mysql> SELECT s.full_name, sc.scholarship_name, d.amount_paid, d.payment_status
    -> FROM Students s
    -> JOIN Disbursement d ON s.student_id = d.student_id
    -> JOIN Scholarships sc ON sc.scholarship_id = d.scholarship_id;
+--------------+-------------------------+-------------+----------------+
| full_name    | scholarship_name        | amount_paid | payment_status |
+--------------+-------------------------+-------------+----------------+
| Rahul Sharma | Merit Scholarship       |    25000.00 | Paid           |
| Sneha Patil  | Need Based Scholarship  |    30000.00 | Paid           |
+--------------+-------------------------+-------------+----------------+
2 rows in set (0.00 sec)
```



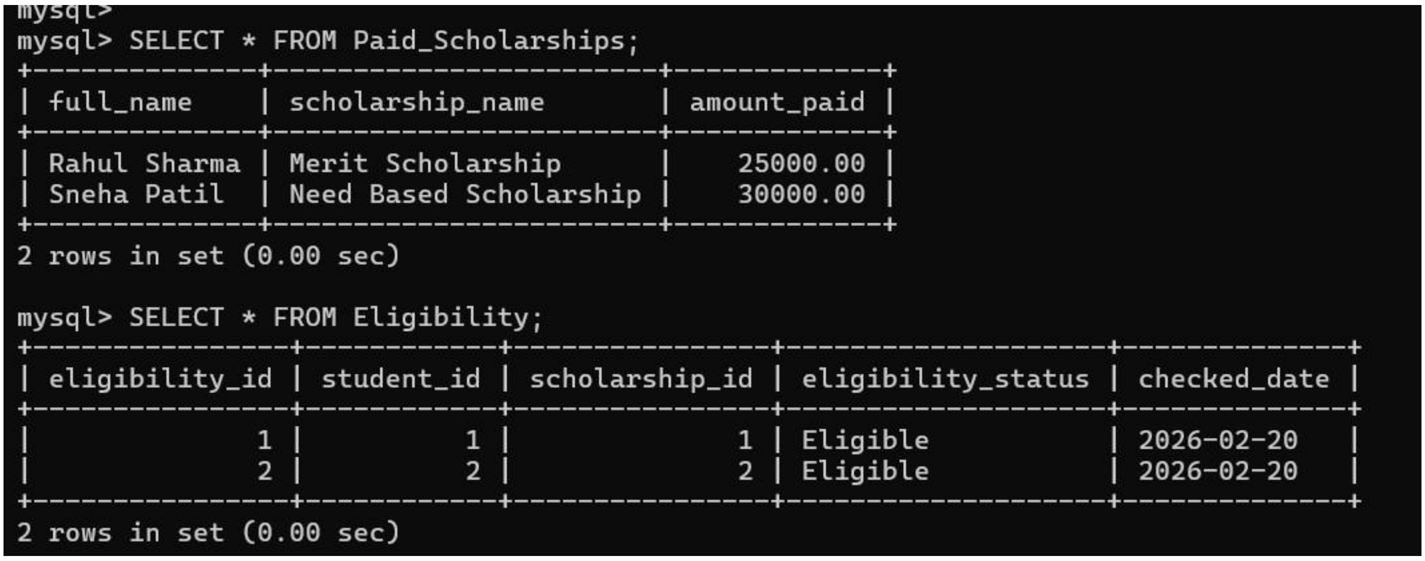
```
mysql> SELECT student_id, SUM(amount_paid) AS total_amount
    -> FROM Disbursement
    -> GROUP BY student_id;
+------------+--------------+
| student_id | total_amount |
+------------+--------------+
|          1 |     25000.00 |
|          2 |     30000.00 |
+------------+--------------+
2 rows in set (0.00 sec)
```



```
mysql> SELECT student_id, SUM(amount_paid) AS total_amount
    -> FROM Disbursement
    -> GROUP BY student_id
    -> HAVING SUM(amount_paid) > 25000;
+------------+--------------+
| student_id | total_amount |
+------------+--------------+
|          2 |     30000.00 |
+------------+--------------+
1 row in set (0.00 sec)
```



```
mysql> SELECT full_name
    -> FROM Students
    -> WHERE family_income <
    -> (SELECT max_income_limit FROM Scholarships WHERE scholarship_name='Merit Scholarship');
+--------------+
| full_name    |
+--------------+
| Rahul Sharma |
| Sneha Patil  |
+--------------+
```



```
mysql>
mysql> SELECT * FROM Paid_Scholarships;
+--------------+------------------------+-------------+
| full_name    | scholarship_name       | amount_paid |
+--------------+------------------------+-------------+
| Rahul Sharma | Merit Scholarship      |    25000.00 |
| Sneha Patil  | Need Based Scholarship |    30000.00 |
+--------------+------------------------+-------------+
2 rows in set (0.00 sec)

mysql> SELECT * FROM Eligibility;
+----------------+------------+----------------+--------------------+--------------+
| eligibility_id | student_id | scholarship_id | eligibility_status | checked_date |
+----------------+------------+----------------+--------------------+--------------+
|              1 |          1 |              1 | Eligible           | 2026-02-20   |
|              2 |          2 |              2 | Eligible           | 2026-02-20   |
+----------------+------------+----------------+--------------------+--------------+
2 rows in set (0.00 sec)
```

CHAPTER 11: SECURITY, BACKUP AND RECOVERY

Security, backup, and recovery are essential components of the Library Book Issue & Fine Management System. Since the system stores important information such as book records, member details, issue transactions, and fine data, it is necessary to protect the database from unauthorized access, accidental deletion, and system failures. MySQL provides strong security features and reliable backup and recovery tools that help maintain data safety and system reliability. Implementing proper security ensures that only authorized users can access or modify data. Backup and recovery ensure that data can be restored in case of hardware failure, system crash, or human error. These features help maintain continuous database operation and protect valuable library information.

Security

- **User Privileges**

User privileges define what actions a user can perform on the database. In MySQL, administrators can assign different permissions such as SELECT, INSERT, UPDATE, DELETE, CREATE, and DROP. For example, an admin can have full access to manage database structure and data, while library staff can be given limited access to perform daily operations like inserting issue records or updating return details. This helps prevent unauthorized data modification.

- **Access Control**

Access control ensures that only authorized users can log in and use the database system. MySQL provides authentication using usernames and passwords. Access control also allows restriction of specific tables or database operations. This helps protect sensitive data and ensures data confidentiality and integrity.

Backup

- Mysql dump Explanation mysql dump is a MySQL utility tool used to create database backup files. It exports database structure and data into a SQL file. This backup file can be stored safely and used later for recovery. Regular backups help protect data from accidental deletion, hardware failure, or system crash.

Recovery

- **Restore Concept**

Recovery is the process of restoring database data using backup files. If the database is damaged or

data is lost, the backup file created using mysqldump can be used to restore the database. Recovery helps maintain system continuity and prevents permanent data loss.

CHAPTER 12: FUTURE SCOPE AND CONCLUSION

The Library Book Issue & Fine Management System provides a strong foundation for managing library records using a relational database. Although the current system focuses mainly on database implementation using MySQL, it can be enhanced further by adding new technologies and advanced features. Future improvements can make the system more user-friendly, automated, and suitable for large-scale organizations. The system currently handles book records, member records, issue transactions, return tracking, and fine calculation efficiently. However, with technological advancements, more intelligent and automated features can be added to improve overall performance and usability. The future scope mainly focuses on expanding the system beyond database-level implementation and integrating it with modern applications and tools.

Future Scope

• Web Integration

The system can be converted into a web-based application using technologies such as PHP, Java, or Python. This will allow users to access the library system through web browsers without directly using MySQL Workbench.

• Mobile Application Integration

A mobile app can be developed so that students and staff can check book availability, due dates, and fine status using smartphones.

• Advanced Reporting System

The system can include graphical reports, charts, and automated report generation for library analysis and management.

• Notification System

Automatic email or SMS notifications can be added to remind members about due dates and fine payments.

• Cloud Database Storage

The database can be hosted on cloud platforms for better accessibility and backup support.

• Data Analytics Features

Advanced analytics can help identify popular books, member reading trends, and library usage statistics.

Conclusion

The Library Book Issue & Fine Management System successfully demonstrates how database technology can be used to manage library operations efficiently. The system reduces manual work, minimizes errors, and improves data accuracy. Using MySQL provides strong data security, fast query processing, and reliable data storage. The project helps in understanding database design, SQL
implementation, table relationships, and constraint usage.

Through this project, practical knowledge of database management is gained. It helps in understanding how real-world database systems are designed and implemented. The system ensures efficient data management, faster data retrieval, and reliable record storage.

13. REFERENCES

• MySQL Official Documentation – Oracle Corporation
Provides complete reference for MySQL commands, database design, and administration.

• Elmasri, R., & Navathe, S. B. – *Fundamentals of Database Systems*
Used for understanding database concepts, relational model, and normalization.

• Silberschatz, A., Korth, H. F., & Sudarshan, S. – *Database System Concepts* Reference for database architecture, SQL queries, and transaction management.

• MySQL Workbench Official Guide
Used for understanding database design, query execution, and schema management.

• W3Schools SQL Tutorial
Used for learning SQL syntax, joins, constraints, and query examples.

• GeeksforGeeks – Database Management System & SQL Tutorials Used for practical SQL query examples and DBMS concept explanations.

• Tutorialspoint – MySQL Tutorial
Used for learning MySQL commands and database operations.

14. GLOSSARY

• DBMS (Database Management System)
A software system used to create, manage, and maintain databases. It allows users to store, retrieve, and manipulate data efficiently.

• SQL (Structured Query Language)
A standard programming language used to interact with databases. It is used for creating tables, inserting data, updating records, and retrieving data.

• Database
An organized collection of related data stored electronically for easy access, management, and updating.

• Table
A structure in a database that stores data in rows and columns. Each table represents an entity such as Students or Scholarships.

• Primary Key
A unique identifier for each record in a table. It ensures that no duplicate records exist.

• Foreign Key
A field that creates a relationship between two tables. It links data from one table to another.

• MySQL
An open-source relational database management system used to store and manage structured data.

• Constraint
Rules applied to table columns to maintain data accuracy and integrity, such as NOT NULL, UNIQUE, PRIMARY KEY, and FOREIGN KEY.

• Normalization
The process of organizing database tables to reduce data redundancy and improve data consistency.

• Query
A command used to interact with the database, such as retrieving or updating data.

• JOIN
An SQL operation used to combine data from two or more tables based on related columns.

• Schema
The logical structure of a database that defines tables, fields, and relationships.

• Data Integrity
Ensures accuracy and consistency of data throughout its lifecycle.

• Data Redundancy
Duplication of data in multiple places, which can lead to inconsistency.

• Relational Database
A database that stores data in tables and uses relationships between tables for data management.