Sinhgad Institutes

A PROJECT ON

"STUDENT MANAGEMENT SYSTEM"

SUBMIT TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN PARTIAL FULFILLMENT FOR THE AWARDS OF DEGREE OF

SECOND YEAR ENGINEERING

(INFORMATION TECHNOLOGY)

SUBMITTED BY

SHREYAS ZOPE	S190578603
AKSHATA GARJE	S190578526
SANSKRUTI PAWAR	S190578569
AKSHAY WALUNJKAR	S190578601
RUSHIKESH TOKLE	S190578596

Under the Guidance of

Prof. B.N.Durgade

2023-24

DEPARTMENT OF INFORMATION TECHNOLOGY
SINHAGAD TECHNICAL EDUCATION SOCIETY
SINHAGAD INSTITUTE OF TECHNOLOGY AND SCIENCE,
NARHE, PUNE - 411041

CERTIFICATE

This is to certify that the project report entitled

"STUDENT MANAGEMENT SYSTEM"

SUBMITTED BY

SHREYAS ZOPE	S190578603
AKSHATA GARJE	S190578526
SANSKRUTI PAWAR	S190578569
AKSHAY WALUNJKAR	S190578601
RUSHIKESH TOKLE	S190578596

is a Bonafide work carried out by him under the Guidance of Prof. B. N. Durgade and it is approved for the partial fulfillment of requirement of the University of Pune for the award of the Second Year Engineering (Information Technology)
This project report has not been earlier submitted to any other institute of university for the award of any degree or diploma.

Prof. B. N. Durgade **Project Guide**	**Mr. T. D. khadtare** **Head, IT**	**Dr. S.D.Markande** **Principal, SITS**

DEPARTMENT OF INFORMATION TECHNOLOGY
SINHAGAD INSTITUTE OF TECHNOLOGY AND SCIENCE,
NARHE, PUNE – 411041

Acknowledgements

With deep sense of Gratitude we would like to thank all the people who have lit our path with their kind guidance. We are very grateful to these intellectuals who did their best to help during our project work.

It is our proud privilege to express a deep sense of gratitude to Prof. Dr. S. D. Markande, Principal of Sinhgad Institute of Technology and Science (SITS), for his comments and kind permission to complete this project. We remain indebted to Prof. T. D. Khadtare, (H.O.D). Information Technology Department for his timely suggestion and valuable guidance.

The special gratitude goes to Mrs. B.N. Durgade excellent and precious guidance in completion of this work. We thanks to all the colleagues for their appreciable help for our working project. With various industry owners or lab technicians to help, it has been our endeavor throughout our work to cover the entire project work.

We are also thankful to our parents who provided their wishful support for our project completion successfully. And Lastly we thank our all friends and the people who are directly or indirectly related to our project work.

Abstract

The student management system is an environment where all the process of the student in the institution is managed . It is done through the automated computerized method. Conventionally this system is done using papers , files and binders. This system saves the time of the student and of the administrator.

The system offers a user friendly interface, allowing and manage various aspects of the event ,including guest management, vendor coordination, budgeting, and task scheduling. The database structure is designed to support the complex relationships and dependencies involved in student management system planning, enabling seamless data organization and retrieval

CONTENTS

Sr. No.	Title	Page No.
1.	INTRODUCTION	1-1
2.	E-R DIAGRAM	2-3
3.	MYSQL INTRODUCTION COMMAND TO CREATE DATABASE LIST OF TABLES STRUCTURE AND DATA OF EACH TABLE	4-8
4.	SOURCE CODE	9-13
5.	NORMALIZATION	14-18
6.	CONCLUSION	19-19

CHAPTER 1

INTRODUCTION

A Student Management System (SMS) in a Database Management System (DBMS) project is designed to efficiently store, manage, and retrieve information related to students, courses, grades, attendance, and other relevant data within an educational institution.

This report aims to explore the concept of a student management system using MYSQL a popular and powerful relational database management system. The system offers a comprehensive and efficient solution to streamline various aspects of wedding planning, including managing teaching staff, students and their information ,data related to students and exam information.The primary objective of this report is to highlight the importance of utilizing a database management system like MYSQL in the context of student management. Throughout this report, we will delve into the fundamental components and functionalities of a student management system powered by MYSQL. We will discuss its key features, such as creating and managing students data related to their exam, class, fees, personal information and generating reports for analysis and documentation purposes.

The primary objective of the Student Management System project is to provide an efficient and centralized platform for managing student-related activities and information within educational institutions, such as schools, colleges, and universities. The project involves designing a relational database schema to represent the entities and relationships involved in student management. This includes defining tables for students, courses, classes, faculty, grades, attendance, and other relevant entities. The database design aims to ensure data integrity, minimize redundancy, and optimize performance.

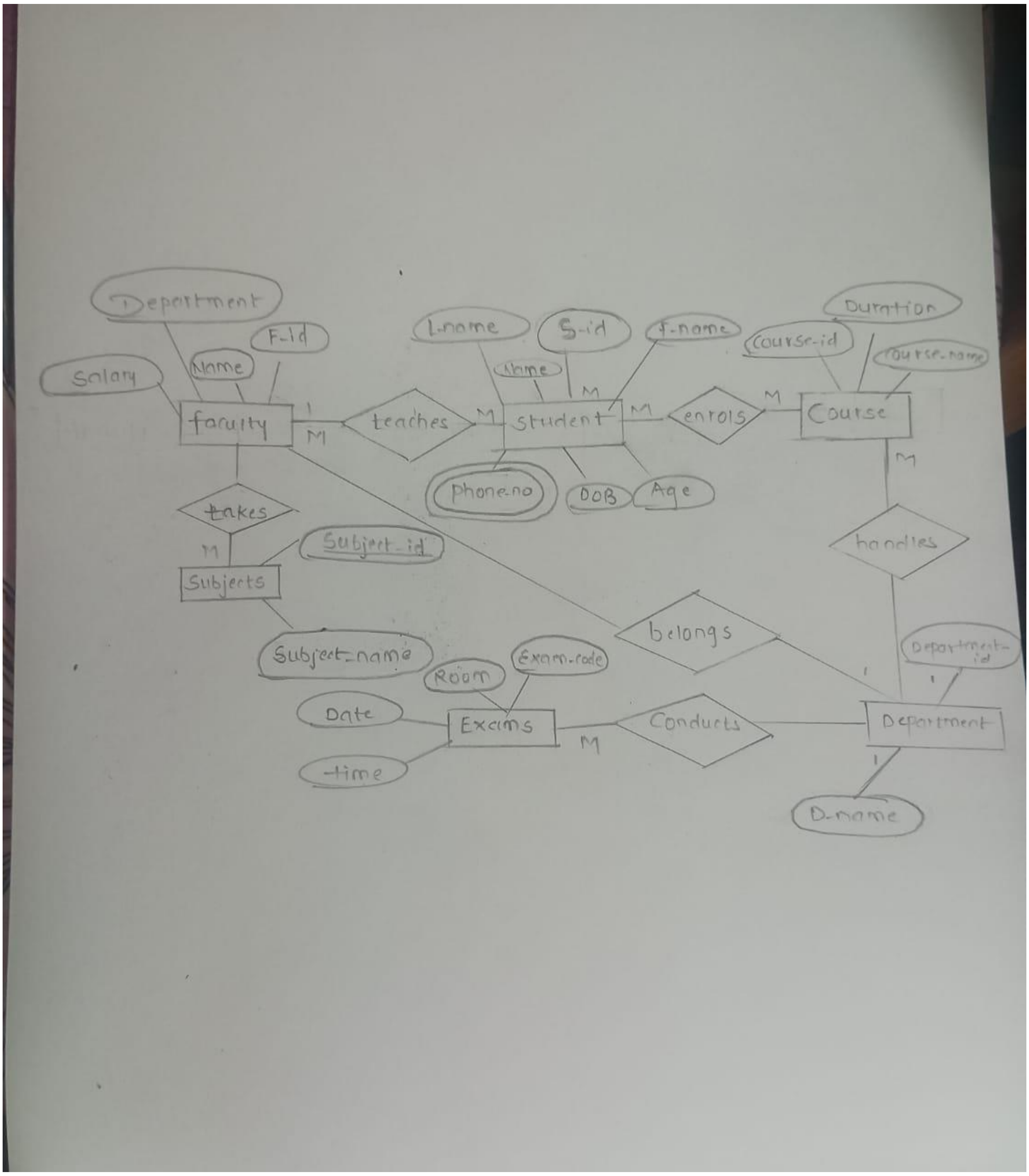
The SMS project involves implementing the database schema using a suitable database management system (DBMS) such as MySQL, PostgreSQL, or Microsoft SQL Server. Additionally, a user-friendly interface (e.g., web-based or desktop application) is developed to interact with the database and provide functionality for administrators, faculty, and students

CHAPTER 2

E-R DIAGRAM

An entity relationship model (ER model) describes inter related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types. In software engineering an ER model is commonly formed to represent things that a business an abstract data model that defines a data or information structure that can be implemented in a database typically a relational database relationship modelling was developed for database design Peter Chen and published in a 1976 paper. However variants of the idea existed previously, some ER modelers show: super and subtype entities connected by generalization specialization relationships, and an ER model can be used also in the specification of domain specific ontology. An ER model is typically implemented as a database. In a simple relational database implementation ,each row of a table represents one instance of an entity type, and each field in a table represents an attribute type. In a relational database a relationship between entities is implemented by storing the primary key of one entity as a pointer or foreign key: in the table of another entity there is a tradition for ER/data models to built at two or three levels of abstraction. Note that the conceptual logical physical hierarchy below is used in other kinds of specification and its different from the three schema approach ton software engineering.

The ER diagram for student management system



CHAPTER 3

MYSQL

- MySQL is a popular open-source relational database management system.

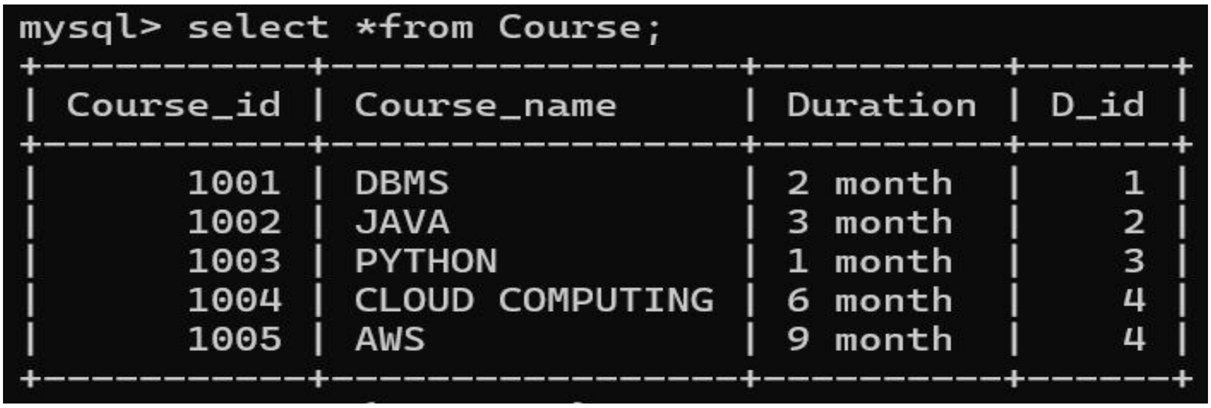
- It allows users to store, manage, and retrieve data efficiently.

- MySQL uses Structured Query Language (SQL) for managing the database.

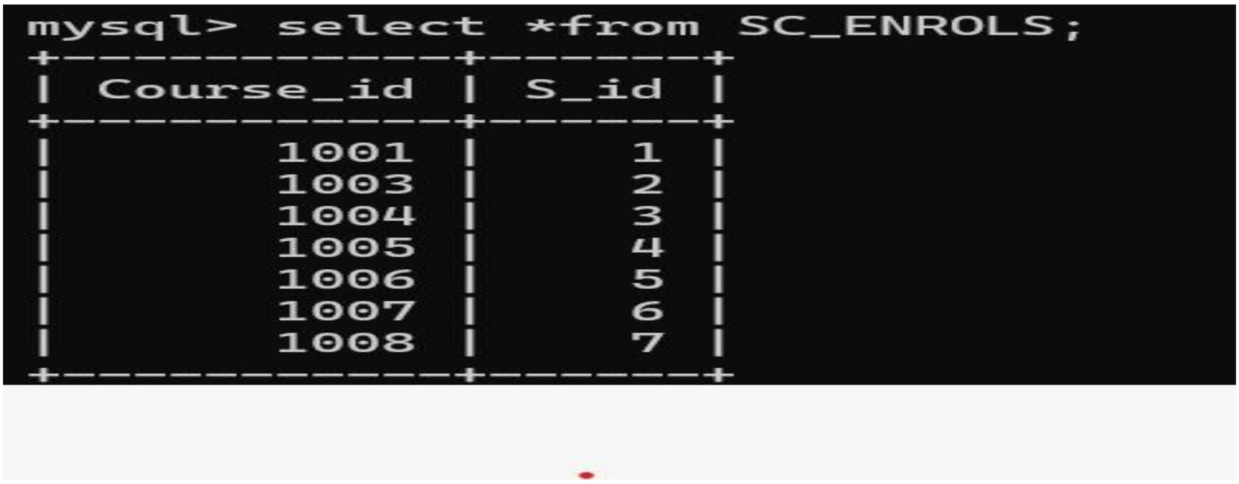
- It supports various operating systems and programming languages.

- MySQL is widely used for web applications, business databases, and more.
- MySQL is known for its scalability, making it suitable for both small-scale and large-scale applications.
- It offers robust security features to protect sensitive data.
- MySQL provides high availability through features like replication and clustering.
- It has a large and active community of developers and users, providing extensive support and resources.
- MySQL offers various storage engines to optimize performance and meet different application requirements.
- It is used by many popular websites and applications, including Facebook, Twitter, and YouTube.
- MySQL is continually updated and improved, with new features and enhancements regularly released by the MySQL development team.
- It can be easily integrated with other technologies and tools, such as PHP, Python, and Java.
- MySQL provides comprehensive documentation and tutorials for users of all skill levels.

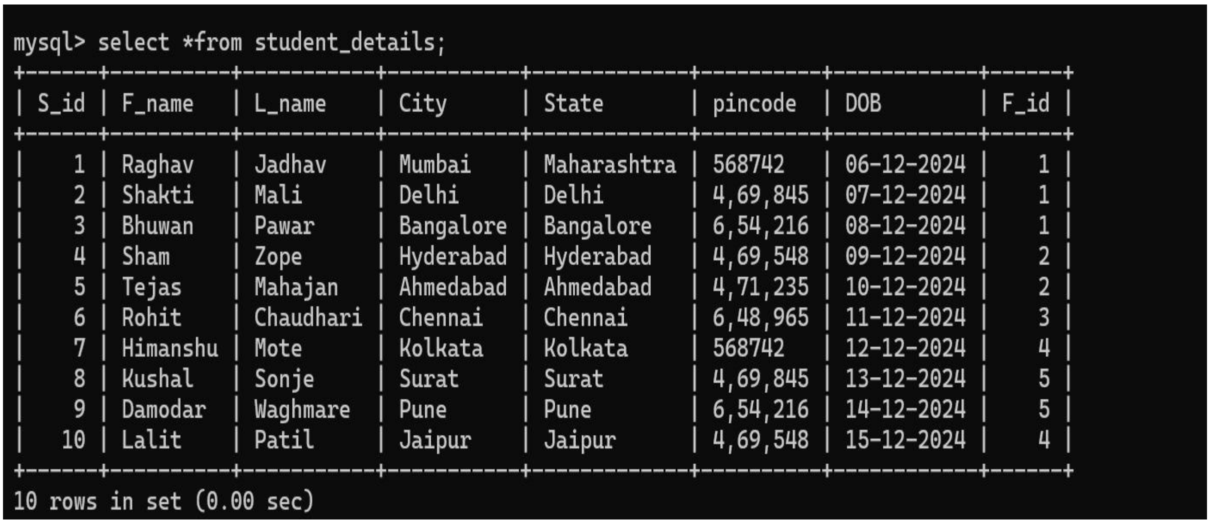
LIST OF TABLES



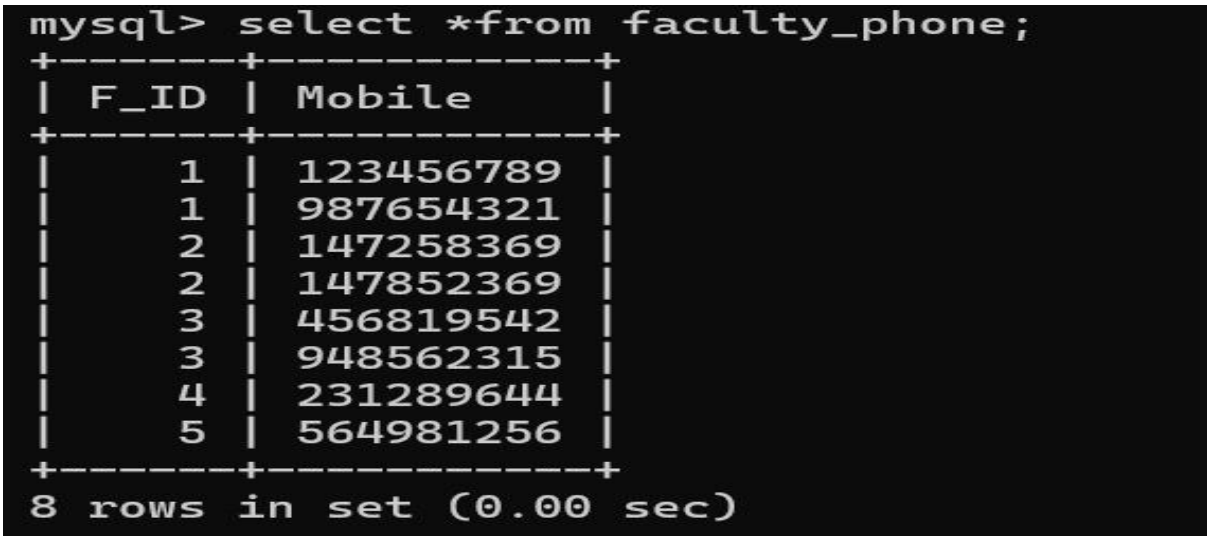
```
mysql> select *from Course;
+-----------+-----------------+----------+------+
| Course_id | Course_name     | Duration | D_id |
+-----------+-----------------+----------+------+
|      1001 | DBMS            | 2 month  |    1 |
|      1002 | JAVA            | 3 month  |    2 |
|      1003 | PYTHON          | 1 month  |    3 |
|      1004 | CLOUD COMPUTING | 6 month  |    4 |
|      1005 | AWS             | 9 month  |    4 |
+-----------+-----------------+----------+------+
```



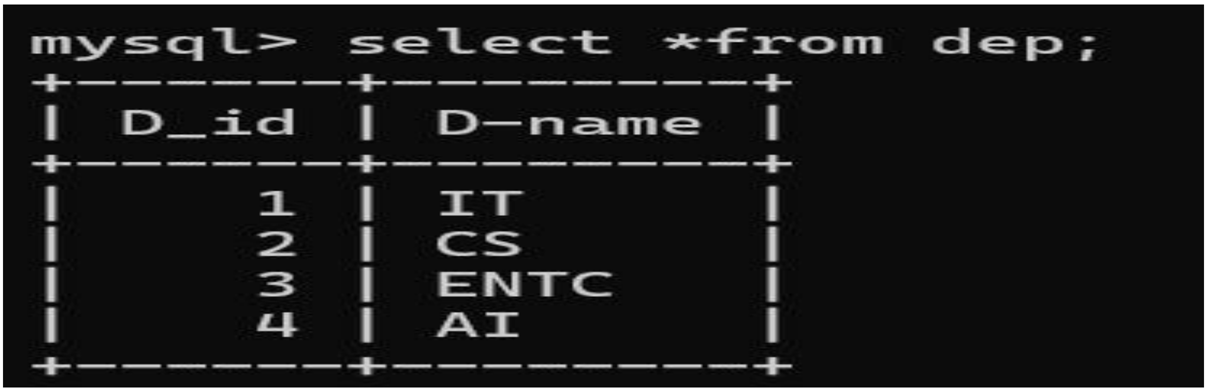
```
mysql> select *from SC_ENROLS;
+-----------+-------+
| Course_id | S_id  |
+-----------+-------+
|      1001 |     1 |
|      1003 |     2 |
|      1004 |     3 |
|      1005 |     4 |
|      1006 |     5 |
|      1007 |     6 |
|      1008 |     7 |
+-----------+-------+
```



```
mysql> select *from student_details;
+------+----------+----------+-----------+-------------+----------+------------+------+
| S_id | F_name   | L_name   | City      | State       | pincode  | DOB        | F_id |
+------+----------+----------+-----------+-------------+----------+------------+------+
|    1 | Raghav   | Jadhav   | Mumbai    | Maharashtra | 568742   | 06-12-2024 |    1 |
|    2 | Shakti   | Mali     | Delhi     | Delhi       | 4,69,845 | 07-12-2024 |    1 |
|    3 | Bhuwan   | Pawar    | Bangalore | Bangalore   | 6,54,216 | 08-12-2024 |    1 |
|    4 | Sham     | Zope     | Hyderabad | Hyderabad   | 4,69,548 | 09-12-2024 |    2 |
|    5 | Tejas    | Mahajan  | Ahmedabad | Ahmedabad   | 4,71,235 | 10-12-2024 |    2 |
|    6 | Rohit    | Chaudhari| Chennai   | Chennai     | 6,48,965 | 11-12-2024 |    3 |
|    7 | Himanshu | Mote     | Kolkata   | Kolkata     | 568742   | 12-12-2024 |    4 |
|    8 | Kushal   | Sonje    | Surat     | Surat       | 4,69,845 | 13-12-2024 |    5 |
|    9 | Damodar  | Waghmare | Pune      | Pune        | 6,54,216 | 14-12-2024 |    5 |
|   10 | Lalit    | Patil    | Jaipur    | Jaipur      | 4,69,548 | 15-12-2024 |    4 |
+------+----------+----------+-----------+-------------+----------+------------+------+
10 rows in set (0.00 sec)
```



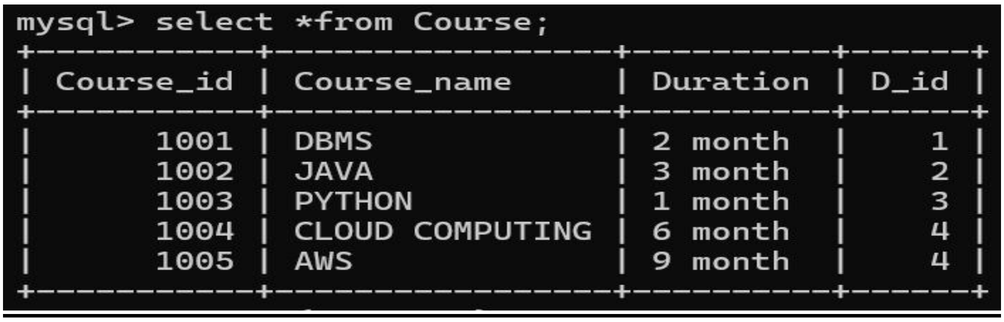
```
mysql> select *from faculty_phone;
+------+-----------+
| F_ID | Mobile    |
+------+-----------+
|    1 | 123456789 |
|    1 | 987654321 |
|    2 | 147258369 |
|    2 | 147852369 |
|    3 | 456819542 |
|    3 | 948562315 |
|    4 | 231289644 |
|    5 | 564981256 |
+------+-----------+
8 rows in set (0.00 sec)
```



```
mysql> select *from dep;
+------+---------+
| D_id | D-name  |
+------+---------+
|    1 | IT      |
|    2 | CS      |
|    3 | ENTC    |
|    4 | AI      |
+------+---------+
```

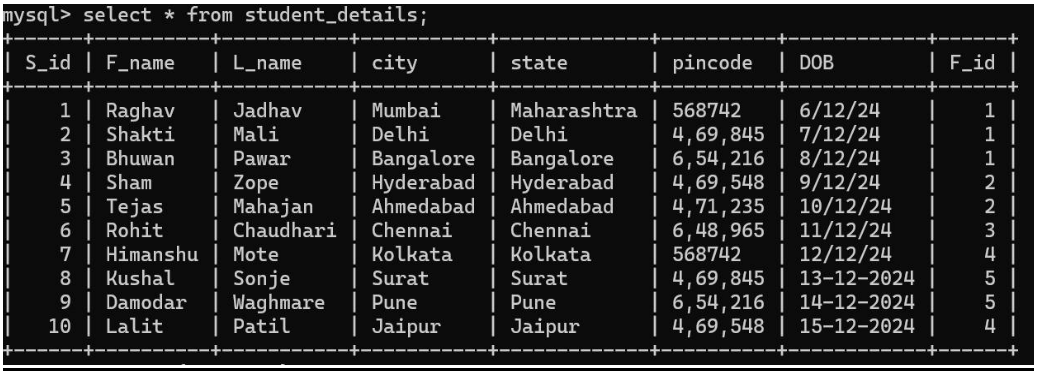
STRUCTURE AND DATA OF EACH TABLE

1) Cousrse details:



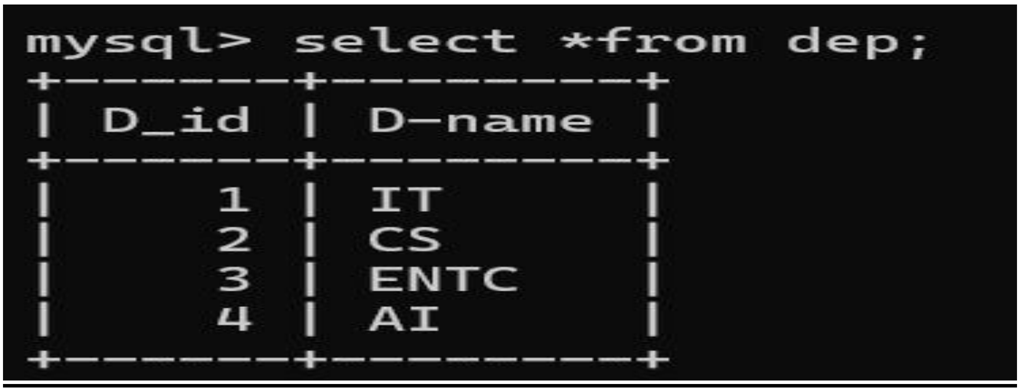
```
mysql> select *from Course;
+-----------+-----------------+----------+------+
| Course_id | Course_name     | Duration | D_id |
+-----------+-----------------+----------+------+
|      1001 | DBMS            | 2 month  |    1 |
|      1002 | JAVA            | 3 month  |    2 |
|      1003 | PYTHON          | 1 month  |    3 |
|      1004 | CLOUD COMPUTING | 6 month  |    4 |
|      1005 | AWS             | 9 month  |    4 |
+-----------+-----------------+----------+------+
```

2) Students details



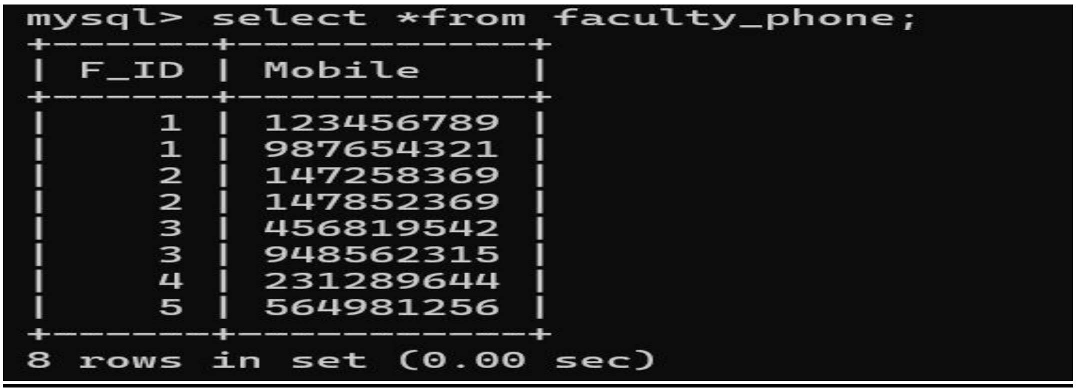
```
mysql> select * from student_details;
+------+----------+-----------+-----------+-------------+----------+------------+------+
| S_id | F_name   | L_name    | city      | state       | pincode  | DOB        | F_id |
+------+----------+-----------+-----------+-------------+----------+------------+------+
|    1 | Raghav   | Jadhav    | Mumbai    | Maharashtra | 568742   | 6/12/24    |    1 |
|    2 | Shakti   | Mali      | Delhi     | Delhi       | 4,69,845 | 7/12/24    |    1 |
|    3 | Bhuwan   | Pawar     | Bangalore | Bangalore   | 6,54,216 | 8/12/24    |    1 |
|    4 | Sham     | Zope      | Hyderabad | Hyderabad   | 4,69,548 | 9/12/24    |    2 |
|    5 | Tejas    | Mahajan   | Ahmedabad | Ahmedabad   | 4,71,235 | 10/12/24   |    2 |
|    6 | Rohit    | Chaudhari | Chennai   | Chennai     | 6,48,965 | 11/12/24   |    3 |
|    7 | Himanshu | Mote      | Kolkata   | Kolkata     | 568742   | 12/12/24   |    4 |
|    8 | Kushal   | Sonje     | Surat     | Surat       | 4,69,845 | 13-12-2024 |    5 |
|    9 | Damodar  | Waghmare  | Pune      | Pune        | 6,54,216 | 14-12-2024 |    5 |
|   10 | Lalit    | Patil     | Jaipur    | Jaipur      | 4,69,548 | 15-12-2024 |    4 |
+------+----------+-----------+-----------+-------------+----------+------------+------+
```

3) Department



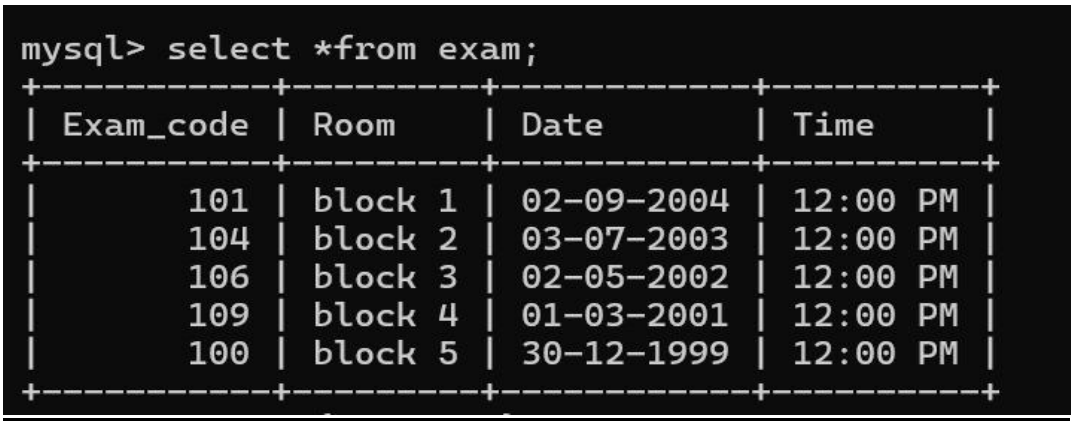
```
mysql> select *from dep;
+------+--------+
| D_id | D-name |
+------+--------+
|    1 | IT     |
|    2 | CS     |
|    3 | ENTC   |
|    4 | AI     |
+------+--------+
```

4) Faculty



```
mysql> select *from faculty_phone;
+------+-----------+
| F_ID | Mobile    |
+------+-----------+
|    1 | 123456789 |
|    1 | 987654321 |
|    2 | 147258369 |
|    2 | 147852369 |
|    3 | 456819542 |
|    3 | 948562315 |
|    4 | 231289644 |
|    5 | 564981256 |
+------+-----------+
8 rows in set (0.00 sec)
```

5) Exam:



```
mysql> select *from exam;
+-----------+---------+------------+----------+
| Exam_code | Room    | Date       | Time     |
+-----------+---------+------------+----------+
|       101 | block 1 | 02-09-2004 | 12:00 PM |
|       104 | block 2 | 03-07-2003 | 12:00 PM |
|       106 | block 3 | 02-05-2002 | 12:00 PM |
|       109 | block 4 | 01-03-2001 | 12:00 PM |
|       100 | block 5 | 30-12-1999 | 12:00 PM |
+-----------+---------+------------+----------+
```

CHAPTER 4

SOURCE CODE

```
CREATE TABLE course (
  Course_id INT,
  Course_name VARCHAR(255),
  Duration VARCHAR(255),
  D_id INT
);


 INSERT INTO course (Course_id, Course_name, Duration, D_id)
VALUES (1001, 'DBMS', '2 month', 1),
       (1002, 'JAVA', '3 month', 1),
       (1003, 'PYTHON', '1 month', 2),
       (1004, 'CLOUD COMPUTING', '6 month', 4),
       (1005, 'AWS', '9 month', 3);

CREATE TABLE student_details (
  S_id INT,
  F_name VARCHAR(255),
  L_name VARCHAR(255),
  City VARCHAR(255),
  State VARCHAR(255),
  pincode INT, DOB DATE, F_id INT);
 INSERT INTO student_details (S_id, F_name, L_name, City, State, pincode, DOB, F_id)
VALUES (1, 'Raghav', 'Jadhav', 'Mumbai', 'Maharashtra', 568742, '2024-12-06', 1),
       (2, 'Shakti', 'Mali', 'Delhi', 'Delhi', 469845, '2024-12-07', 1),
       (3, 'Bhuwan', 'Pawar', 'Bangalore', 'Bangalore', 654216, '2024-12-08', 1),
       (4, 'Sham', 'Zope', 'Hyderabad', 'Hyderabad', 469548, '2024-12-09', 2),
       (5, 'Tejas', 'Mahajan', 'Ahmedabad', 'Ahmedabad', 471235, '2024-12-10', 2),
       (6, 'Rohit', 'Chaudhari', 'Chennai', 'Chennai', 648965, '2024-12-11', 3),
       (7, 'Himanshu', 'Mote', 'Kolkata', 'Kolkata', 568742, '2024-12-12', 41),
       (8, 'Kushal', 'Sonje', 'Surat', 'Surat', 469845, '2024-12-13', 5),
       (9, 'Damodar', 'Waghmare', 'Pune', 'Pune', 654216, '2024-12-14', 5),
       (10, 'Lalit', 'Patil', 'Jaipur', 'Jaipur', 469548, '2024-12-15', 41);

CREATE TABLE faculty_phone (
  F_ID INT,
  Mobile VARCHAR(255)
);
```

```
INSERT INTO faculty_phone (F_ID, Mobile)
VALUES (1, '1234567890'),
       (1, '9876543210'),
       (2, '1472583690'),
       (2, '1478523690'),
       (3, '4568195420'),
       (3, '9485623150'),
       (4, '2312896440'),
       (5, '5649812560');

CREATE TABLE dep (
  D_id INT PRIMARY KEY,
  D_name VARCHAR(255)
);

INSERT INTO dep (D_id, D_name)
VALUES (1, 'IT'),
       (2, 'CS'),
       (3, 'ENTC'),
       (4, ‘AI');
```

```
CREATE TABLE exam (

  Exam_code INT,

  Room VARCHAR(255),

  Date DATE,

  Time VARCHAR(255)

);


 INSERT INTO exam (Exam_code, Room, Date, Time)

VALUES (101, 'block 1', '2004-09-02', '12:00 PM'),

       (104, 'block 2', '2003-07-03', '12:00 PM'),

       (106, 'block 3', '2002-05-02', '12:00 PM'),

       (109, 'block 4', '2001-03-01', '12:00 PM'),

       (100, 'block 5', '1999-12-30', '12:00 PM');


CREATE TABLE sc_enroll (

  Course_id INT,

  Student_id INT,

  Faculty_id INT

);
```

```
INSERT INTO sc_enroll (Course_id, Student_id, Faculty_id)
VALUES (1001, 1, NULL),
       (1001, 10, 1),
       (1002, 2, 1),
       (1003, 3, 2),
       (1004, 4, 4),
       (1005, 5, 3),
       (1003, 1, NULL),
       (1004, 2, NULL),
       (1005, 3, NULL),
       (1001, 5, 1),
       (1002, NULL, 2),
       (1003, 4, NULL),
       (1004, NULL, 4),
       (1005, NULL, 3);
```

CHAPTER 5

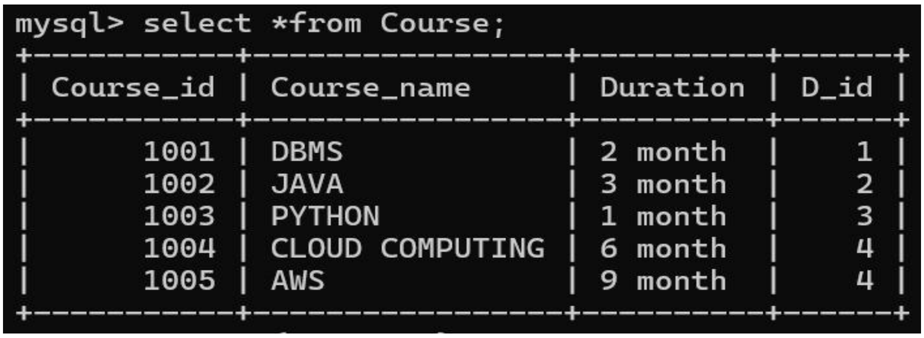
NORMALIZATION

Normalization in DBMS (Database Management Systems) is a systematic process of organizing data in a database to reduce redundancy and dependency. It involves breaking down a larger table into smaller, more manageable tables and establishing relationships between them. The goal of normalization is to design a database schema that minimizes redundancy, prevents data anomalies, and ensures data integrity.

Need of normalization:

1. **Elimination of Redundancy**
2. **Data Integrity**
3. **Simplified Updates**
4. **Improved Query Performance**
5. **Scalability**
6. **Easier Maintenance**

First Normal form:

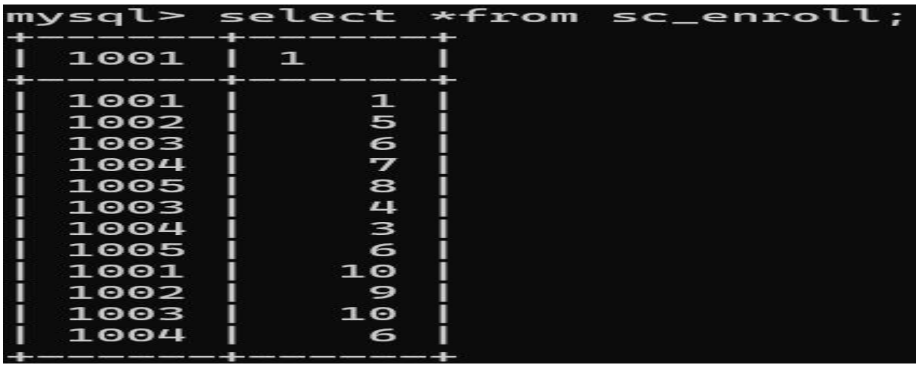


```
mysql> select *from Course;
+-----------+-----------------+----------+------+
| Course_id | Course_name     | Duration | D_id |
+-----------+-----------------+----------+------+
|      1001 | DBMS            | 2 month  |    1 |
|      1002 | JAVA            | 3 month  |    2 |
|      1003 | PYTHON          | 1 month  |    3 |
|      1004 | CLOUD COMPUTING | 6 month  |    4 |
|      1005 | AWS             | 9 month  |    4 |
+-----------+-----------------+----------+------+
```

First Normal Form :

It is a database design principle that ensures each table in a relational database contains only atomic values, meaning each cell contains a single value rather than a set of values. This eliminates repeating groups within a table, ensuring data integrity and simplifying data manipulation. To achieve 1NF, you break down tables into smaller, more manageable structures, avoiding multi-valued attributes and nested structures.

Second Normal Form:



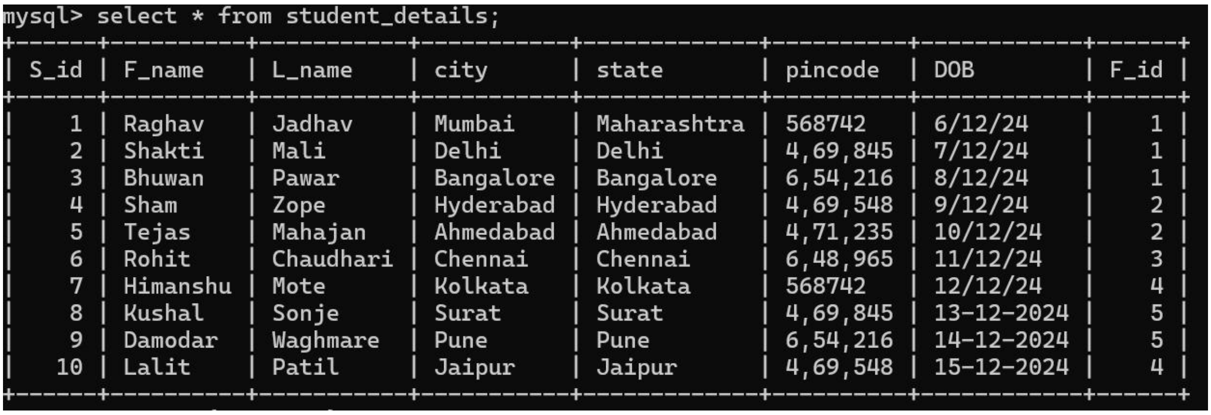
```
mysql> select *from sc_enroll;
+-------+-------+
| 1001  | 1     |
+-------+-------+
| 1001  |     1 |
| 1002  |     5 |
| 1003  |     6 |
| 1004  |     7 |
| 1005  |     8 |
| 1003  |     4 |
| 1004  |     3 |
| 1005  |     6 |
| 1001  |    10 |
| 1002  |     9 |
| 1003  |    10 |
| 1004  |     6 |
+-------+-------+
```

Second Normal Form (2NF) is a database design concept that builds upon First Normal Form (1NF). In 2NF, a table must meet the criteria of 1NF, and additionally, all non-key attributes must be fully functionally dependent on the entire primary key. This means that every non-key attribute must depend on the entire primary key, rather than only a part of it. By ensuring 2NF, you reduce data redundancy and anomalies, improving the efficiency and integrity of the database.

1. Building upon 1NF: Before achieving 2NF, the database must already be in First Normal Form (1NF), which ensures that each table contains only atomic values.
2. Eliminating Partial Dependencies: 2NF aims to eliminate partial dependencies within a table. A partial dependency occurs when a non-prime attribute (an attribute that is not part of the primary key) is dependent on only a portion of the primary key.
3. Prime and Non-Prime Attributes: In the context of 2NF, prime attributes are attributes that form part of the primary key, while non-prime attributes are those that are not part of the primary key.

4. Functional Dependency: An attribute A is functionally dependent on attribute B if, for every value of B, there is exactly one corresponding value of A. In 2NF, every non-prime attribute must be fully functionally dependent on the entire primary key.
5. Decomposition: To achieve 2NF, you may need to decompose (split) tables into multiple tables to ensure that each table meets the requirements of 2NF. This involves creating separate tables for subsets of related attributes.
6. Reducing Redundancy: By eliminating partial dependencies and ensuring full functional dependencies, 2NF helps reduce data redundancy, which can lead to more efficient storage and maintenance of the database.

Third Normal Form:



```
mysql> select * from student_details;
+------+----------+-----------+-----------+-------------+----------+------------+------+
| S_id | F_name   | L_name    | city      | state       | pincode  | DOB        | F_id |
+------+----------+-----------+-----------+-------------+----------+------------+------+
|    1 | Raghav   | Jadhav    | Mumbai    | Maharashtra | 568742   | 6/12/24    |    1 |
|    2 | Shakti   | Mali      | Delhi     | Delhi       | 4,69,845 | 7/12/24    |    1 |
|    3 | Bhuwan   | Pawar     | Bangalore | Bangalore   | 6,54,216 | 8/12/24    |    1 |
|    4 | Sham     | Zope      | Hyderabad | Hyderabad   | 4,69,548 | 9/12/24    |    2 |
|    5 | Tejas    | Mahajan   | Ahmedabad | Ahmedabad   | 4,71,235 | 10/12/24   |    2 |
|    6 | Rohit    | Chaudhari | Chennai   | Chennai     | 6,48,965 | 11/12/24   |    3 |
|    7 | Himanshu | Mote      | Kolkata   | Kolkata     | 568742   | 12/12/24   |    4 |
|    8 | Kushal   | Sonje     | Surat     | Surat       | 4,69,845 | 13-12-2024 |    5 |
|    9 | Damodar  | Waghmare  | Pune      | Pune        | 6,54,216 | 14-12-2024 |    5 |
|   10 | Lalit    | Patil     | Jaipur    | Jaipur      | 4,69,548 | 15-12-2024 |    4 |
+------+----------+-----------+-----------+-------------+----------+------------+------+
```

Third Normal Form (3NF) is a database normalization concept that builds upon the principles of First Normal Form (1NF) and Second Normal Form (2NF). In 3NF, a table must meet the criteria of both 1NF and 2NF, and additionally, it must eliminate transitive dependencies.

Transitive dependencies occur when a non-prime attribute (an attribute that is not part of the primary key) is functionally dependent on another non-prime attribute, rather than directly on the primary key.

To achieve 3NF, you need to:

1. Ensure the table is in 1NF and 2NF.
2. Eliminate transitive dependencies by moving non-key attributes that are dependent on other non-key attributes to their own tables.

By achieving Third Normal Form (3NF), you can further reduce data redundancy, enhance data integrity, and facilitate more efficient data manipulation and management in relational databases.

4NF(Fourth Normal form)Rules:

Fourth Normal Form (4NF) is a level of database normalization that builds upon the principles of Third Normal Form (3NF). In 4NF, a table must satisfy the criteria of 3NF, and it must also address multi-valued dependencies.

A multi-valued dependency occurs when a non-prime attribute (attribute not part of the primary key) is dependent on a combination of attributes that are not the primary key, rather than on the primary key itself.

To achieve 4NF, you need to:

1. Ensure the table is in 3NF.
2. Eliminate or handle multi-valued dependencies by decomposing the table into smaller tables, each representing a single-valued dependency.

By achieving Fourth Normal Form (4NF), you can further reduce data redundancy and anomalies, improving data integrity and database efficiency.

Fifth Normal Form (5NF) :

5NF is a level of database normalization that addresses the issue of join dependencies, specifically in the context of multi-table relations. In 5NF, a table must satisfy the criteria of Fourth Normal Form (4NF), and it must also handle join dependencies.

A join dependency occurs when a table can be reconstructed by joining multiple smaller tables. In 5NF, these join dependencies are addressed by decomposing the tables further to eliminate redundancy and ensure that each table represents an independent entity.

Achieving Fifth Normal Form (5NF) helps improve data integrity, reduce redundancy, and maintain consistency in complex relational database designs. It ensures that the database structure is optimized for efficient data storage and manipulation

Sixth Normal Form (6NF):

It is the highest level of database normalization, designed to handle complex data relationships and eliminate redundancy. In 6NF, a table must satisfy the criteria of Fifth Normal Form (5NF), and it must also address additional types of dependencies known as join dependencies and join irreducibility.

Join dependencies in 6NF refer to situations where information in one table can be derived by joining multiple other tables. Join irreducibility indicates that there are no further decompositions possible without losing information.

Achieving Sixth Normal Form (6NF) ensures the most efficient and non-redundant database structure, ideal for complex data models with intricate relationships and dependencies. It facilitates optimal data storage, retrieval, and maintenance in highly normalized databases.

CHAPTER 5

CONCLUSION

The development and implementation of the Student Management System (SMS) using a Database Management System (DBMS) have been successfully accomplished. Throughout the project lifecycle, several significant achievements have been realized. A comprehensive solution for managing student information, including personal details, academic records, and administrative tasks. The relational database design ensures data integrity, consistency, and efficient retrieval, facilitating streamlined operations for educational institutions. Looking ahead, there are opportunities for further enhancement and expansion of the SMS. Future iterations could focus on implementing advanced functionalities, such as automated reporting, predictive analytics for student performance, and integration with learning management systems (LMS). Additionally, scalability considerations should be prioritized to accommodate growing user bases and evolving educational requirements.

In conclusion, the Student Management System project represents a significant milestone in leveraging DBMS technology to improve educational administration. By embracing innovation and continuous improvement, the SMS has the potential to positively impact student outcomes and institutional effectiveness in the long term.