Library Management System Database Design

Bachelor of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

Candidate Name

AP22110011481
AP22110011492
AP22110011494
AP22110011500

Submitted To

Dr Sriramulu Bojjagani

Assistant Professor, Computer Science and Engineering

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

April , 2024

ABSTRACT

This document presents a comprehensive overview of the design and implementation of a database for a library management system. Beginning with an exploration of the project background, the document progresses through the development stages, including the Entity Relationship Model (ERM), Enhanced Entity Relationship Model (EERM), and the subsequent description and mapping of these models to the relational schema. Furthermore, it delves into the normalization process, ensuring data integrity and minimizing redundancy within the database. The document also provides a collection of SQL queries and views that facilitate efficient data retrieval and management. Additionally, it discusses the front-end interface, offering insights into the user experience and functionality. Finally, concluding remarks summarize the key findings and contributions of the database design, underscoring its significance in enhancing library management operations.

TABLE OF CONTENTS

PROJECT BACKGROUND

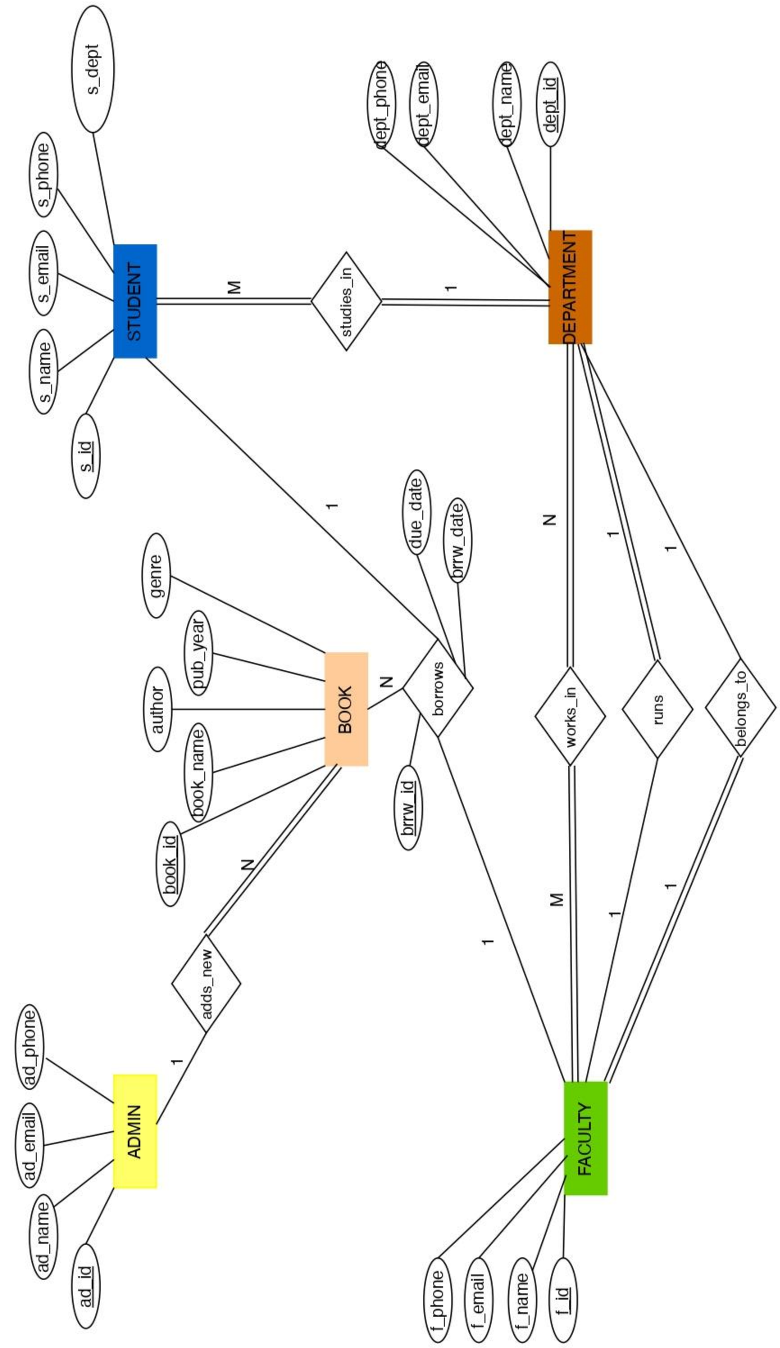
This project is about making a database for a school or college library and academics. It deals with how books, teachers, students, departments, and library workers are all connected. It's designed to handle situations like teachers being part of more than one department and students belonging to just one. The system also has roles for important people like department heads and library staff. The goal is to make things run smoothly and keep track of everything happening in the library and academics.

Statements:

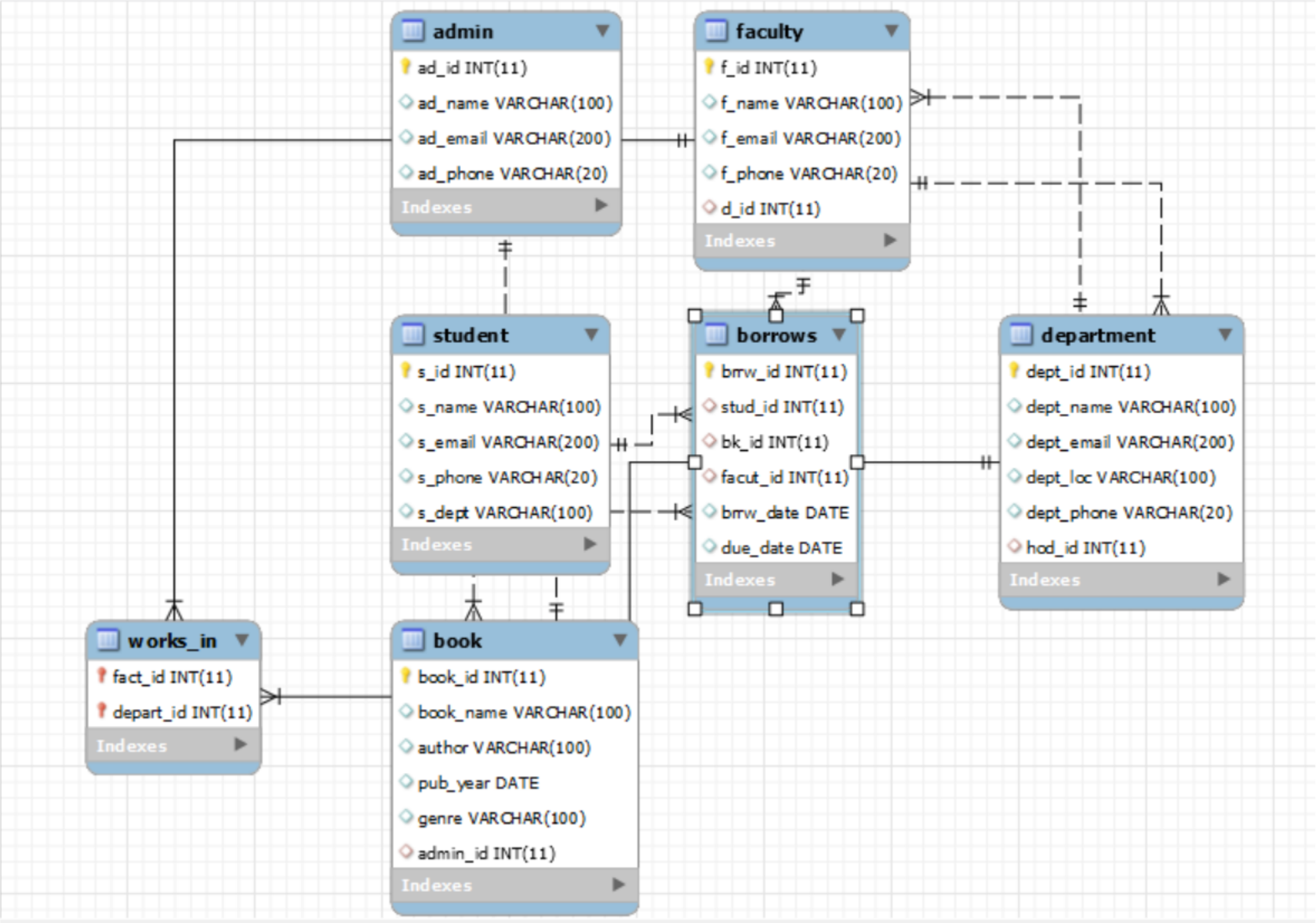
1. Each **BOOK** has an id, name, author, publication year, genre.
2. Each **FACULTY** has an id, name, email, department, contact number.
3. Each **STUDENT** has an id, name, email, department, contact number.
4. Each **LIBRARY_STAFF** has an id, name, email and contact number.
5. Each **DEPARTMENT** has an id, name, email, location and contact number.
6. Each faculty can be associated with more than one department but has a primary department.
7. Each student is associated with a single department, but a department can be associated with multiple students.
8. Each student can borrow multiple books, but one book cannot be borrowed by multiple students.
9. Each faculty can borrow multiple books, but one book cannot be borrowed by multiple faculties.
10. Each library staff is responsible for day-to-day activities (renewal and approval of books) of the library except a special staff named admin who is responsible for including new books in the library database.
11. Each department has a HOD who runs the department.
12. When a book is borrowed, the book_id, borrower_id, borrow_date, due_date is recorded for both students and faculties.

PS: Even if the book is exact the same, it has a different id which as a PK.

ENTITY RELATIONSHIP MODEL



ENHANCED ENTITY RELATIONSHIP MODEL



DESCRPITION OF ENTITY RELATIONSHIP MODEL

STRONG ENTITIES

1. BOOK: ***book_id*** (pk), book_name, author, pub_year, genre
2. ADMIN:***ad_id*** (pk), ad_name, ad_email, ad_phone
3. FACULTY: ***f_id*** (pk), f_name, f_email, f_phone, f_dept
4. STUDENT: ***s_id*** (pk), s_name, s_email, s_phone, s_dept
5. DEPARTMENT: ***dept_id*** (pk), dept_name, dept_loc, dept_email, dept_phone

WEAK ENTITIES

- ➔ No weak entities

RELATIONSHIPS

1. adds_new(admin, book)
2. lends(admin, book)
3. borrows(faculty, book, student): brrw_date, due_date
4. studies_in(student, department)
5. works_in(faculty, department)
6. runs(faculty, department)
7. belongs_to(faculty, department)

CONSTRAINTS

- o [admin] <adds_new> [book] (1:N)
- o [admin] <lends > [book] (1:N)
- o a [faculty] <works_in> many [department] and a [department] can have many [faculty] (M:N)
- o a [faculty] <runs> a [department] (1:1)
- o a [faculty] <belongs_to> a primary [department] (1:1)
- o a [student] <studies_in> in a single [department] and a [department] can have many [students] (M:1)
- o a [faculty] <borrows> many [book] but the same [book] cannot be borrowed by more than one [faculty] (1:M)
- o a [student] <borrows> many [book] but the same [book] cannot be borrowed by more than one [student] (1:M)

ER TO RELATIONAL MODEL MAPPING

Note: Primary Key is represented as bold italicised text e.g. ***PK***

Mapping Strong Entities

BOOK

book_id	book_name	author	pub_year	genre

STUDENT

s_id	s_name	s_email	s_phone	s_dept

FACULTY

f_id	f_name	f_email	f_phone

DEPARTMENT

dept_id	dept_name	dept_email	dept_phone

ADMIN

ad_id	ad _name	ad _email	ad _phone

Mapping Weak Entities

➔ No weak entities

Mapping Binary 1:1 Relationship Type

DEPARTMENT

dept_id	dept_name	dept_email	dept_phone	hod_id (fk)

FACULTY

f_id	f_name	f_email	f_phone	d_id (fk)

Mapping Binary 1: N Relationship Type

ADMIN <adds_new> BOOK (N side)

BOOK

book_id	book_name	author	pub_year	genre	admin_id (fk)

Mapping M:N Relationship Type

FACULTY <works_in> DEPARTMENT

works_in

fact_id	***depart_id***

Mapping N-ary relationship type

borrows

brrw_id	stud_id	bk_id	facut_id	brrw_date	due_date

DESCRIPTION OF TABLE

🔑 **Primary Key**

♦ **Foreign Key**

Final Tables After Relational Mapping

STUDENT	
🔑 ***s_id***	INT
s_name	VARCHAR(100)
s_email	VARCHAR(100)
s_phone	VARCHAR(20)
s_dept	VARCHAR(100)

FACULTY	
🔑 ***f_id***	INT
f_name	VARCHAR(100)
f_email	VARCHAR(100)
f_phone	VARCHAR(20)
♦ d_id	INT

DEPARTMENT	
🔑 ***dept_id***	INT
dept_name	VARCHAR(100)
dept_email	VARCHAR(100)
dept_phone	VARCHAR(20)
♦ hod_id	INT

BOOK	
🔑 ***book_id***	INT
book_name	VARCHAR(100)
author	VARCHAR(100)
pub_year	DATE
genre	VARCHAR(100)
♦ admin_id	INT

ADMIN	
🔑 ***ad_id***	INT
ad_name	VARCHAR(100)
ad_email	VARCHAR(100)
ad_phone	VARCHAR(20)

works_in	
🔑 *fact_id*	INT
🔑 *depart_id*	INT

borrows	
🔑 ***brrw_id***	INT
♦ stud_id	INT
♦ bk_id	INT
♦ facut_id	INT
brrw_date	DATE
due_date	DATE

NORMALIZATION

STUDENT

s_id	s_name	s_email	s_phone	s_dept

1. **First Normal Form**
   All attributes are atomic and there is no any composite or multivalued attributes so the relation is in 1NF**.**

2. **Second Normal Form**
   FD: { *s_id* } -> {s_name, s_email, s_phone, s_dept}

   Since there is no partial dependency so the relation is in 2NF.

3. **Third Normal Form**
   Since there is no transitive dependency so the relation is in 3NF.

FACULTY

f_id	f_name	f_email	f_phone	d_id (fk)

1. **First Normal Form**
   All attributes are atomic and there is no any composite or multivalued attributes so the relation is in 1NF**.**

2. **Second Normal Form**
   FD: { *f_id* } -> {f_name, f_email, f_phone, d_id}

   Since there is no partial dependency so the relation is in 2NF.

3. **Third Normal Form**
   Since there is no transitive dependency so the relation is in 3NF.

DEPARTMENT

dept_id	dept_name	dept_email	dept_phone	hod_id (fk)

1. **First Normal Form**
   All attributes are atomic and there is no any composite or multivalued attributes so the relation is in 1NF**.**

2. **Second Normal Form**
   FD: { *dept_id* } -> {dept_name, dept_email, dept_phone, hod_id}

   Since there is no partial dependency so the relation is in 2NF.

3. **Third Normal Form**
   Since there is no transitive dependency so the relation is in 3NF.

BOOK

book_id	book_name	author	pub_year	genre	admin_id (fk)

1. **First Normal Form**
   All attributes are atomic and there is no any composite or multivalued attributes so the relation is in 1NF**.**

2. **Second Normal Form**
   FD: { *book_id* } -> {book_name, author, pub_year, genre, admin_id}

   Since there is no partial dependency so the relation is in 2NF.

3. **Third Normal Form**
   Since there is no transitive dependency so the relation is in 3NF.

ADMIN

ad_id	ad _name	ad _email	ad _phone

1. **First Normal Form**
   All attributes are atomic and there is no any composite or multivalued attributes so the relation is in 1NF**.**

2. **Second Normal Form**
   FD: { *ad_id* } -> { ad_name, ad_email, ad_phone}

   Since there is no partial dependency so the relation is in 2NF.

3. **Third Normal Form**
   Since there is no transitive dependency so the relation is in 3NF.

works_in

fact_id	***depart_id***

1. **First Normal Form**
   All attributes are atomic and there is no any composite or multivalued attributes so the relation is in 1NF**.**

2. **Second Normal Form**
   Since there is no partial dependency so the relation is in 2NF.

3. **Third Normal Form**
   Since there is no transitive dependency so the relation is in 3NF.

borrows

brrw_id	stud_id	bk_id	facut_id	brrw_date	due_date

1. **First Normal Form**
   All attributes are atomic and there is no any composite or multivalued attributes so the relation is in 1NF**.**

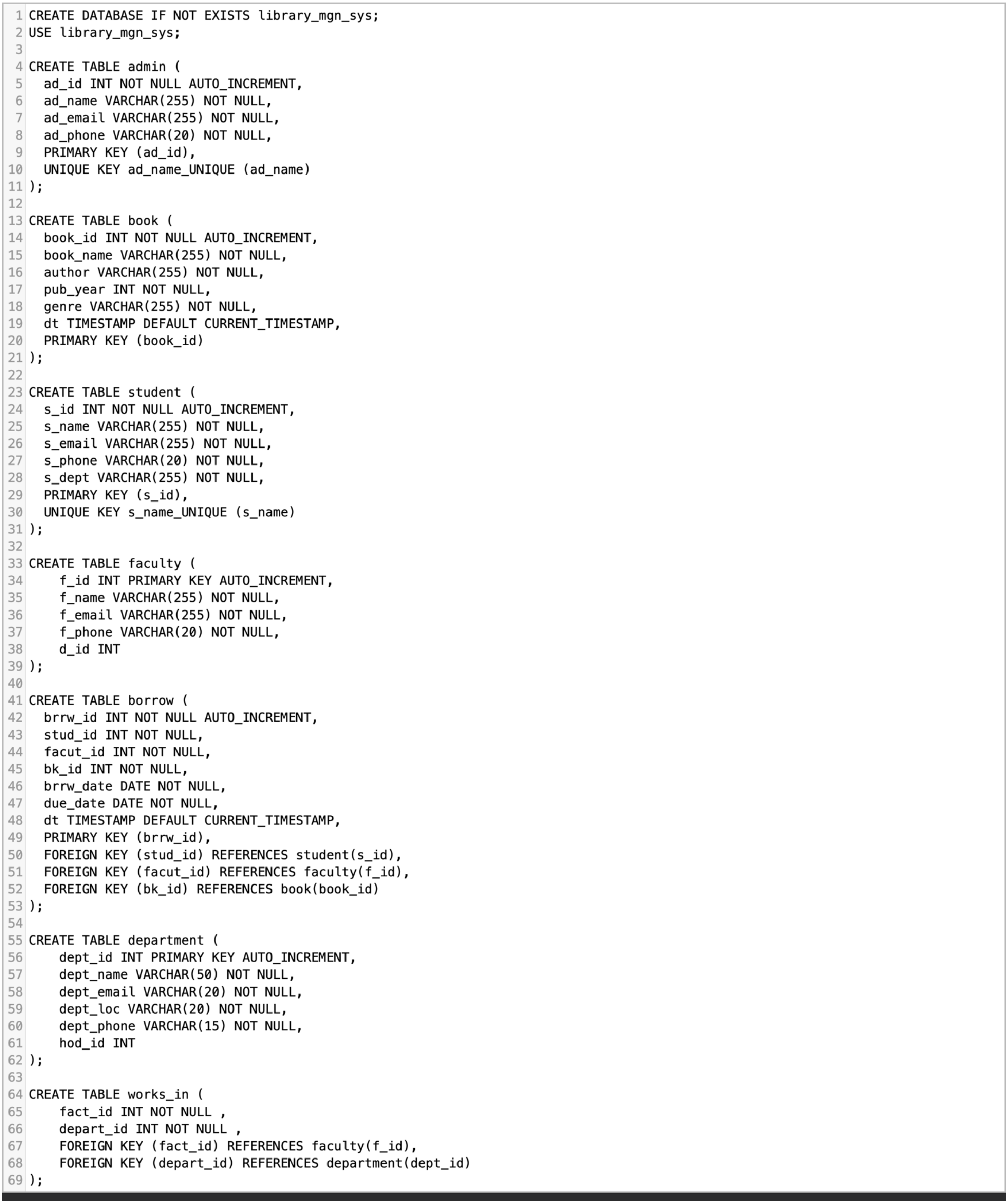
2. **Second Normal Form**

   FD: { *brrw_id* } -> { stud_id, bk_id, facut_id, brrw_date, due_date}

   Since there is no partial dependency so the relation is in 2NF.

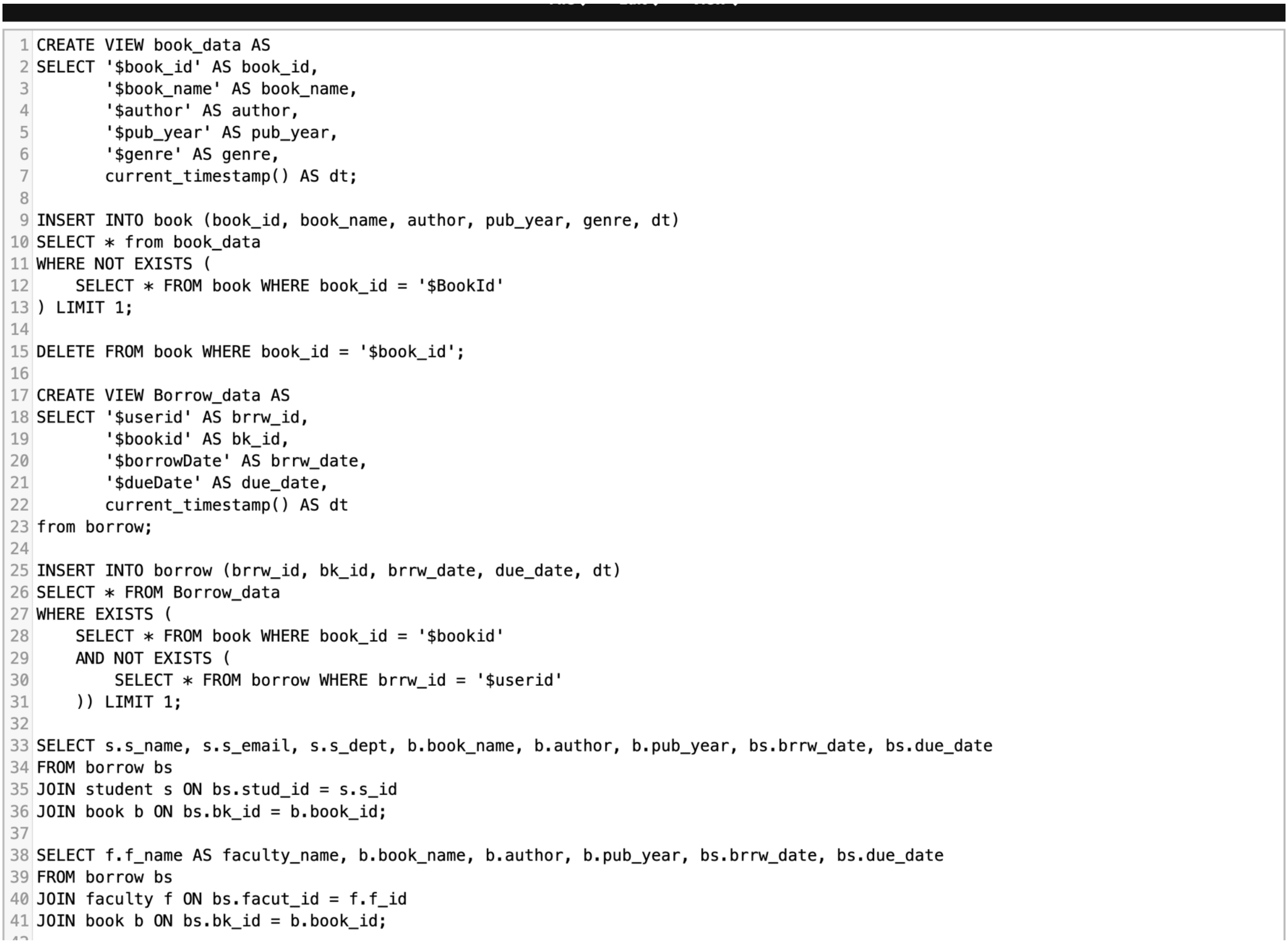
3. **Third Normal Form**
   Since there is no transitive dependency so the relation is in 3NF.

SQL QUERIES

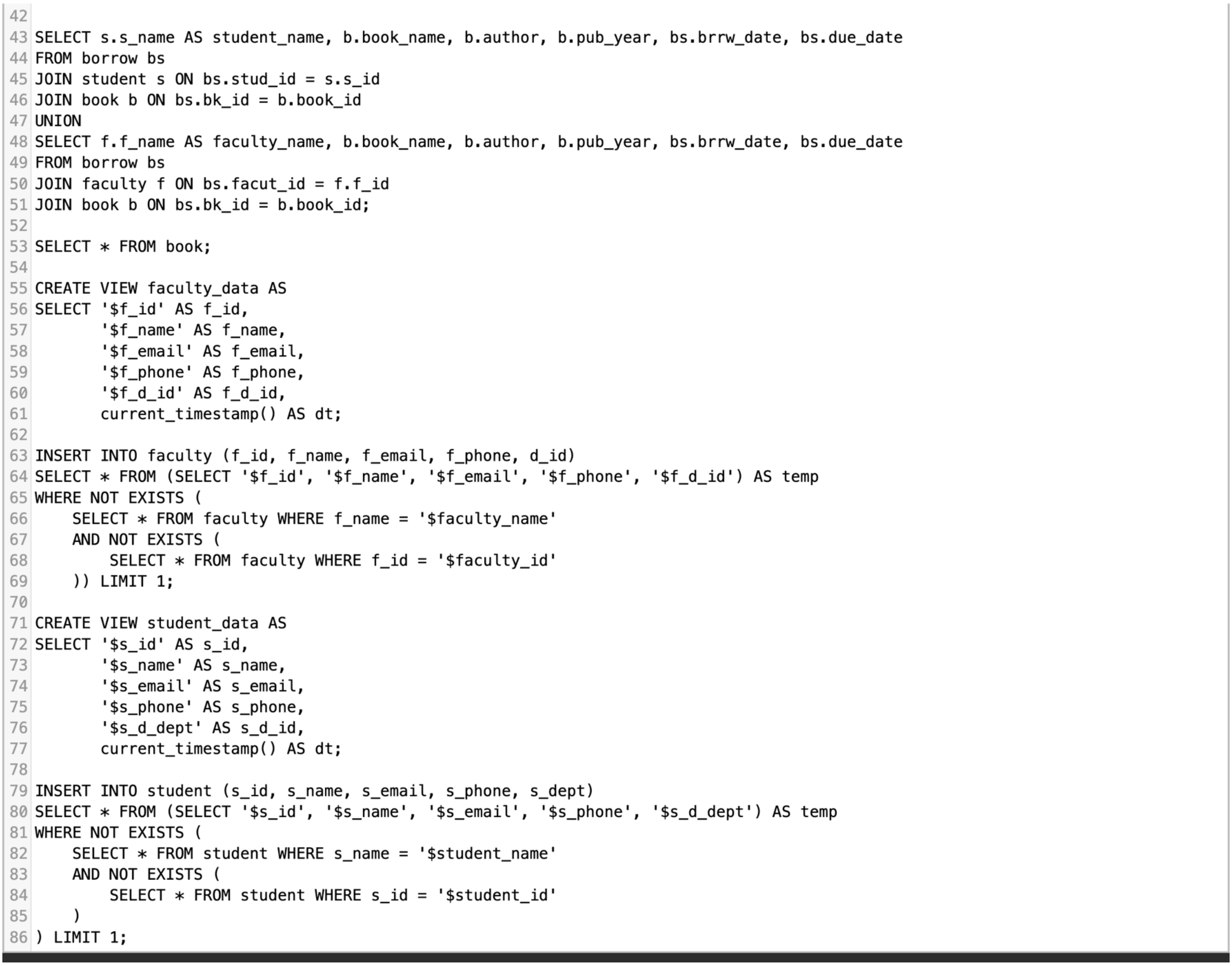


```
CREATE DATABASE IF NOT EXISTS library_mgn_sys;
USE library_mgn_sys;

CREATE TABLE admin (
  ad_id INT NOT NULL AUTO_INCREMENT,
  ad_name VARCHAR(255) NOT NULL,
  ad_email VARCHAR(255) NOT NULL,
  ad_phone VARCHAR(20) NOT NULL,
  PRIMARY KEY (ad_id),
  UNIQUE KEY ad_name_UNIQUE (ad_name)
);

CREATE TABLE book (
  book_id INT NOT NULL AUTO_INCREMENT,
  book_name VARCHAR(255) NOT NULL,
  author VARCHAR(255) NOT NULL,
  pub_year INT NOT NULL,
  genre VARCHAR(255) NOT NULL,
  dt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (book_id)
);

CREATE TABLE student (
  s_id INT NOT NULL AUTO_INCREMENT,
  s_name VARCHAR(255) NOT NULL,
  s_email VARCHAR(255) NOT NULL,
  s_phone VARCHAR(20) NOT NULL,
  s_dept VARCHAR(255) NOT NULL,
  PRIMARY KEY (s_id),
  UNIQUE KEY s_name_UNIQUE (s_name)
);

CREATE TABLE faculty (
    f_id INT PRIMARY KEY AUTO_INCREMENT,
    f_name VARCHAR(255) NOT NULL,
    f_email VARCHAR(255) NOT NULL,
    f_phone VARCHAR(20) NOT NULL,
    d_id INT
);

CREATE TABLE borrow (
  brrw_id INT NOT NULL AUTO_INCREMENT,
  stud_id INT NOT NULL,
  facut_id INT NOT NULL,
  bk_id INT NOT NULL,
  brrw_date DATE NOT NULL,
  due_date DATE NOT NULL,
  dt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (brrw_id),
  FOREIGN KEY (stud_id) REFERENCES student(s_id),
  FOREIGN KEY (facut_id) REFERENCES faculty(f_id),
  FOREIGN KEY (bk_id) REFERENCES book(book_id)
);

CREATE TABLE department (
    dept_id INT PRIMARY KEY AUTO_INCREMENT,
    dept_name VARCHAR(50) NOT NULL,
    dept_email VARCHAR(20) NOT NULL,
    dept_loc VARCHAR(20) NOT NULL,
    dept_phone VARCHAR(15) NOT NULL,
    hod_id INT
);

CREATE TABLE works_in (
    fact_id INT NOT NULL ,
    depart_id INT NOT NULL ,
    FOREIGN KEY (fact_id) REFERENCES faculty(f_id),
    FOREIGN KEY (depart_id) REFERENCES department(dept_id)
);
```

VIEWS



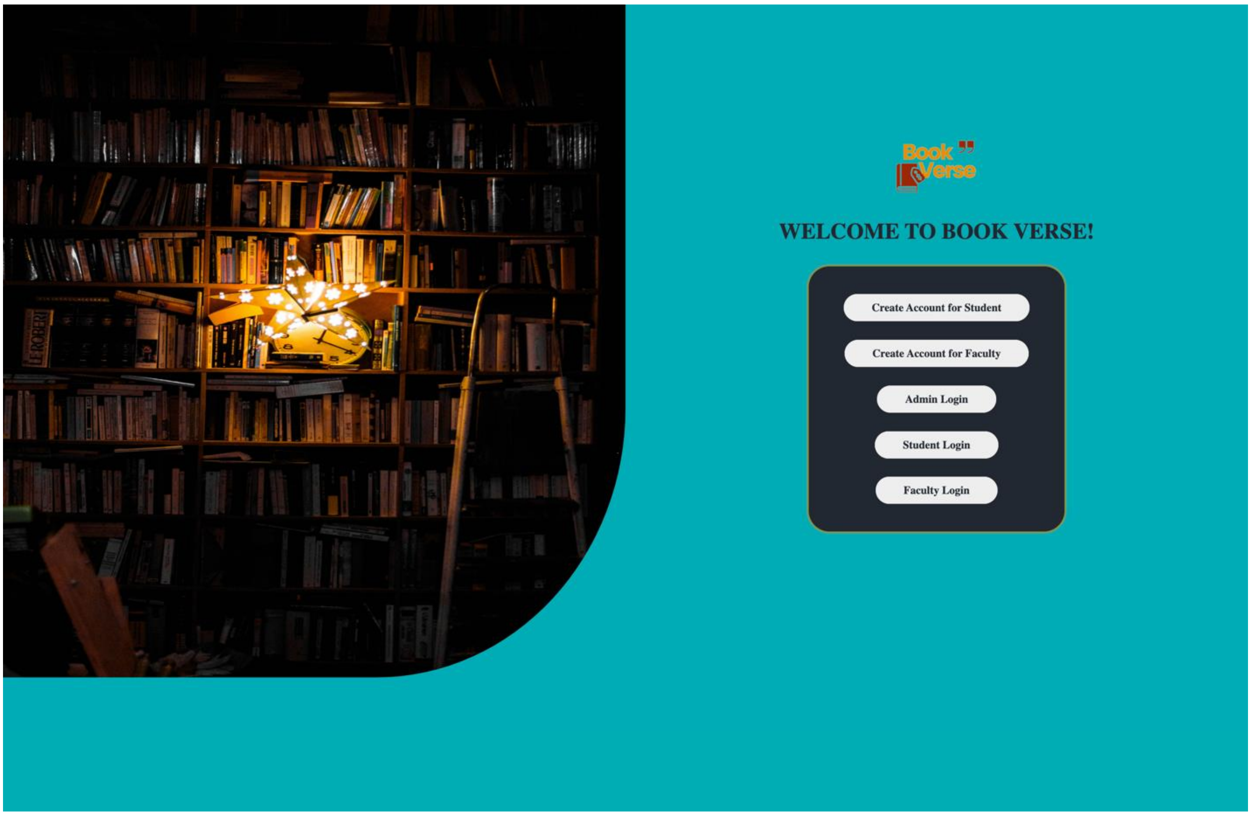
```
CREATE VIEW book_data AS
SELECT '$book_id' AS book_id,
       '$book_name' AS book_name,
       '$author' AS author,
       '$pub_year' AS pub_year,
       '$genre' AS genre,
       current_timestamp() AS dt;

INSERT INTO book (book_id, book_name, author, pub_year, genre, dt)
SELECT * from book_data
WHERE NOT EXISTS (
    SELECT * FROM book WHERE book_id = '$BookId'
) LIMIT 1;

DELETE FROM book WHERE book_id = '$book_id';

CREATE VIEW Borrow_data AS
SELECT '$userid' AS brrw_id,
       '$bookid' AS bk_id,
       '$borrowDate' AS brrw_date,
       '$dueDate' AS due_date,
       current_timestamp() AS dt
from borrow;

INSERT INTO borrow (brrw_id, bk_id, brrw_date, due_date, dt)
SELECT * FROM Borrow_data
WHERE EXISTS (
    SELECT * FROM book WHERE book_id = '$bookid'
    AND NOT EXISTS (
        SELECT * FROM borrow WHERE brrw_id = '$userid'
    )) LIMIT 1;

SELECT s.s_name, s.s_email, s.s_dept, b.book_name, b.author, b.pub_year, bs.brrw_date, bs.due_date
FROM borrow bs
JOIN student s ON bs.stud_id = s.s_id
JOIN book b ON bs.bk_id = b.book_id;

SELECT f.f_name AS faculty_name, b.book_name, b.author, b.pub_year, bs.brrw_date, bs.due_date
FROM borrow bs
JOIN faculty f ON bs.facut_id = f.f_id
JOIN book b ON bs.bk_id = b.book_id;
```



```sql

SELECT s.s_name AS student_name, b.book_name, b.author, b.pub_year, bs.brrw_date, bs.due_date
FROM borrow bs
JOIN student s ON bs.stud_id = s.s_id
JOIN book b ON bs.bk_id = b.book_id
UNION
SELECT f.f_name AS faculty_name, b.book_name, b.author, b.pub_year, bs.brrw_date, bs.due_date
FROM borrow bs
JOIN faculty f ON bs.facut_id = f.f_id
JOIN book b ON bs.bk_id = b.book_id;

SELECT * FROM book;

CREATE VIEW faculty_data AS
SELECT '$f_id' AS f_id,
       '$f_name' AS f_name,
       '$f_email' AS f_email,
       '$f_phone' AS f_phone,
       '$f_d_id' AS f_d_id,
       current_timestamp() AS dt;

INSERT INTO faculty (f_id, f_name, f_email, f_phone, d_id)
SELECT * FROM (SELECT '$f_id', '$f_name', '$f_email', '$f_phone', '$f_d_id') AS temp
WHERE NOT EXISTS (
    SELECT * FROM faculty WHERE f_name = '$faculty_name'
    AND NOT EXISTS (
        SELECT * FROM faculty WHERE f_id = '$faculty_id'
    )) LIMIT 1;

CREATE VIEW student_data AS
SELECT '$s_id' AS s_id,
       '$s_name' AS s_name,
       '$s_email' AS s_email,
       '$s_phone' AS s_phone,
       '$s_d_dept' AS s_d_id,
       current_timestamp() AS dt;

INSERT INTO student (s_id, s_name, s_email, s_phone, s_dept)
SELECT * FROM (SELECT '$s_id', '$s_name', '$s_email', '$s_phone', '$s_d_dept') AS temp
WHERE NOT EXISTS (
    SELECT * FROM student WHERE s_name = '$student_name'
    AND NOT EXISTS (
        SELECT * FROM student WHERE s_id = '$student_id'
    )
) LIMIT 1;
```

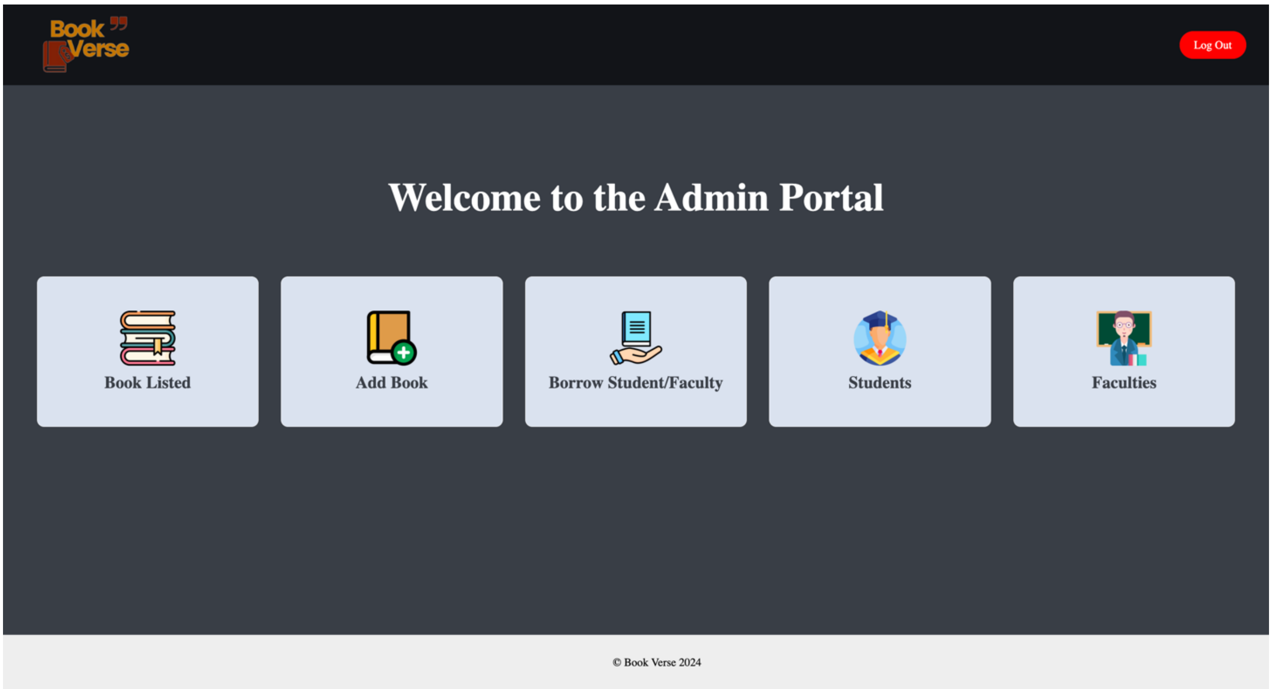
FRONT END

Login Page



WELCOME TO BOOK VERSE!

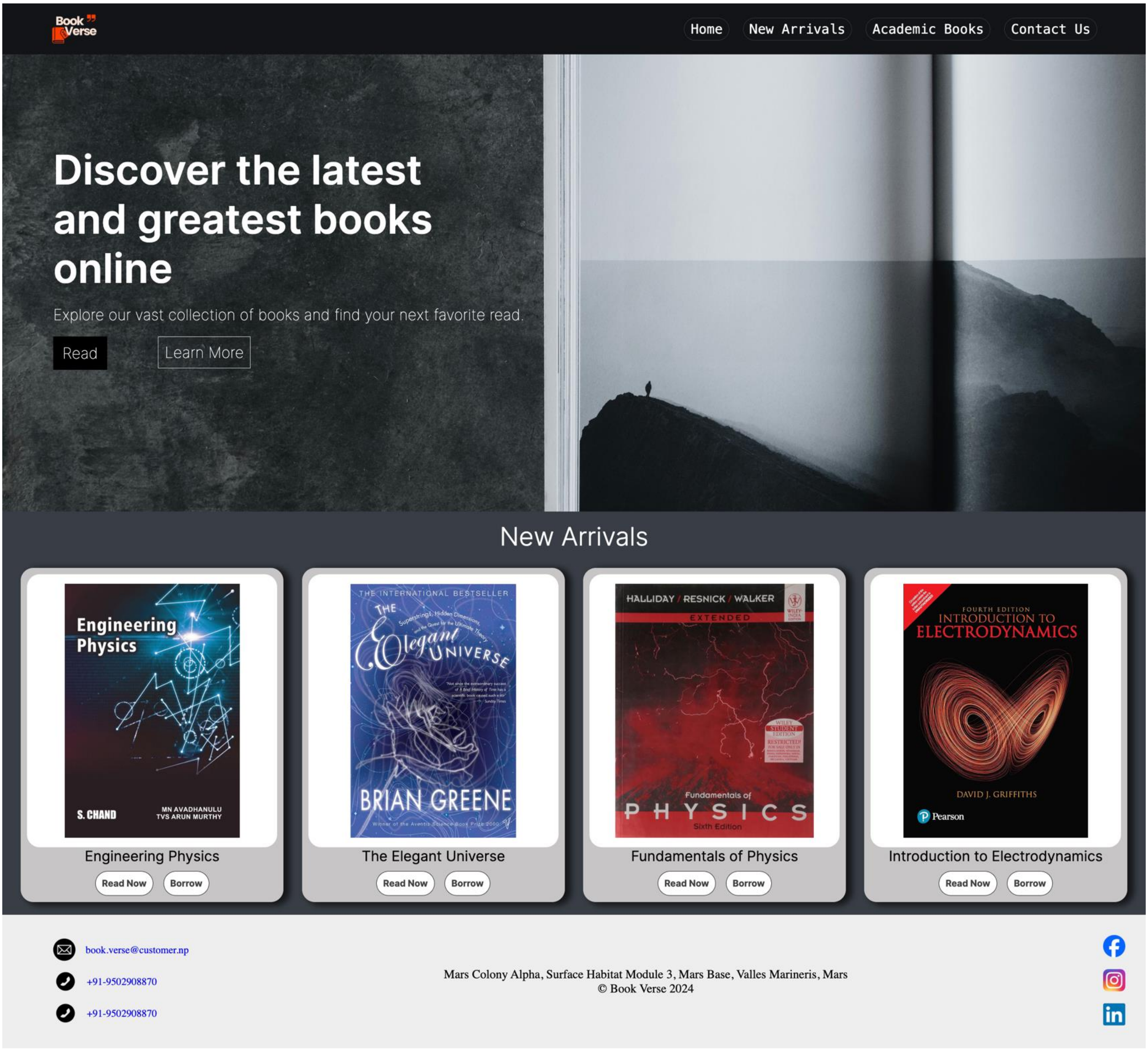
Admin Dashboard



Welcome to the Admin Portal

© Book Verse 2024

Student/Faculty Dashboard



Discover the latest and greatest books online

Explore our vast collection of books and find your next favorite read.

New Arrivals

Engineering Physics

The Elegant Universe

Fundamentals of Physics

Introduction to Electrodynamics

book.verse@customer.np

+91-9502908870

+91-9502908870

Mars Colony Alpha, Surface Habitat Module 3, Mars Base, Valles Marineris, Mars
© Book Verse 2024

CONCLUDING REMARKS

In conclusion, the development and implementation of the library management system's database represent a significant achievement in advancing the operational efficiency of libraries. With careful planning and iterative refinement, we have established a robust database architecture that forms the foundation of our system's functionality. The meticulous creation of Entity Relationship Models (ERM) and Enhanced Entity Relationship Models (EERM) has provided a structured framework for organizing library resources, patrons, and transactions, ensuring seamless management and accessibility.

Furthermore, our commitment to data integrity through the normalization process has resulted in an optimized storage system, minimizing redundancy and facilitating efficient data retrieval and manipulation. The incorporation of SQL queries and views has further enhanced the database's versatility, empowering users to execute a diverse range of operations with ease. As the interface between users and the database, the front-end interface offers intuitive functionality and a user-friendly experience, thanks to its seamless integration with the robust backend.

We owe a debt of gratitude to Dr. Sriramulu Bojjagani, Assistant Professor of Computer Science and Engineering, whose guidance and expertise have been instrumental in navigating the complexities of this project and ensuring its success. His unwavering support and mentorship have been invaluable throughout this journey of development and implementation.