DATABASE MANAGEMENT SYSTEM PROJECT(UCS310)

Submitted by

Kushagr Sharma 102203714

Shivansh Yadav 102203791

Jiya 102203801

Shreeya Kesarwani 102203815

LIBRARY MANAGEMENT SYSTEM

Submitted to

Dr. Ranjit Kumar Ranjan

Department of computer science & engineering

Thapar Institute of Engineering & Technology, Patiala

Jan-May, 2024

Table of Contents

Contents

1 Project Description

1.1 Overview

Our Library Management System is designed to streamline and automate the processes involved in managing a library. Built using PL/SQL, the system provides robust functionality for issuing and returning books, managing fines for overdue returns, and facilitating the booking of Group Discussion (GD) rooms. This system aims to enhance the efficiency of library operations while ensuring an optimal experience for both librarians and library users.

1.2 Key Features

The library management system offers a comprehensive suite of features designed to streamline book issuance and returns, fine management, GD (Group Discussion) room booking, reporting, and analytics, all within a user-friendly interface. Users can issue books with ease by providing their credentials and book details. The system manages due dates and sends automated reminders for overdue books, helping users avoid fines. Upon return, books are checked in, availability is updated, and any associated fines are cleared. The fine management system calculates fines for overdue books according to predefined rules, ensuring consistency and transparency. Users are informed about fines through their accounts and can view fine details.

The system also facilitates group collaboration by allowing users to book GD rooms. Room availability is displayed in real-time, enabling users to choose appropriate time slots. Reporting and analytics functionalities provide insights into library operations, with customizable reports on book issuance, returns, fines collected, and room bookings. These analytics features help identify trends, track popular books, and optimize library resources. The interface is designed to be intuitive and accessible across various devices, including desktops, tablets, and smartphones. Multilingual support is also provided, catering to a diverse user base.

1.3 Benefits

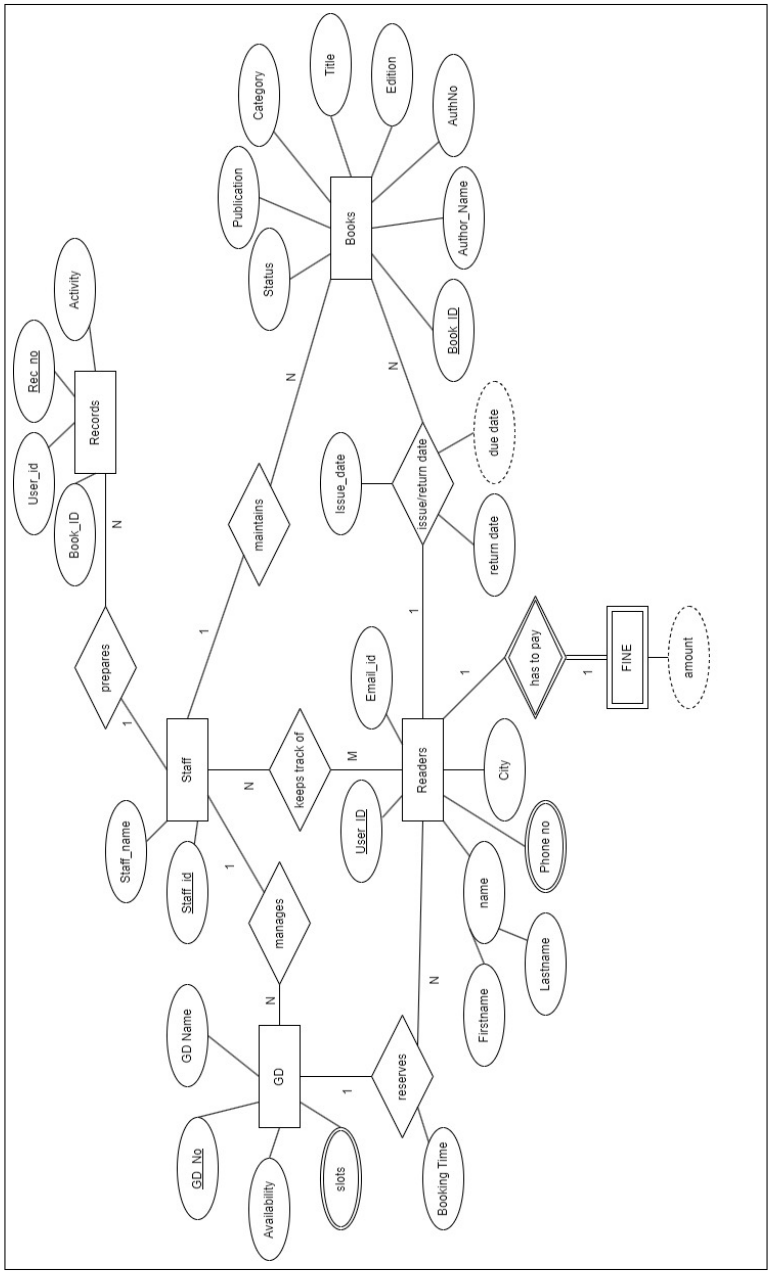
The library management system offers significant benefits, notably in streamlining the management of library resources. It reduces the need for manual processes, thus lowering administrative overhead and improving operational efficiency. By automating book issuance, returns, and fine calculations, library staff can focus on more strategic tasks while users benefit from a smoother experience. This system also centralizes room bookings, allowing for better management of library facilities and optimized utilization of collaborative spaces.

Additionally, the system enhances accountability and transparency through automated tracking of transactions and fines. This feature ensures users are informed of due dates, reducing overdue books, and allows staff to monitor the system for consistency and fairness. The system's scalability and flexibility make it adaptable for future expansion and evolving library requirements. This flexibility ensures that the system can grow with the library, accommodating more users, books, and services as needed, while still maintaining an efficient and user-friendly experience.

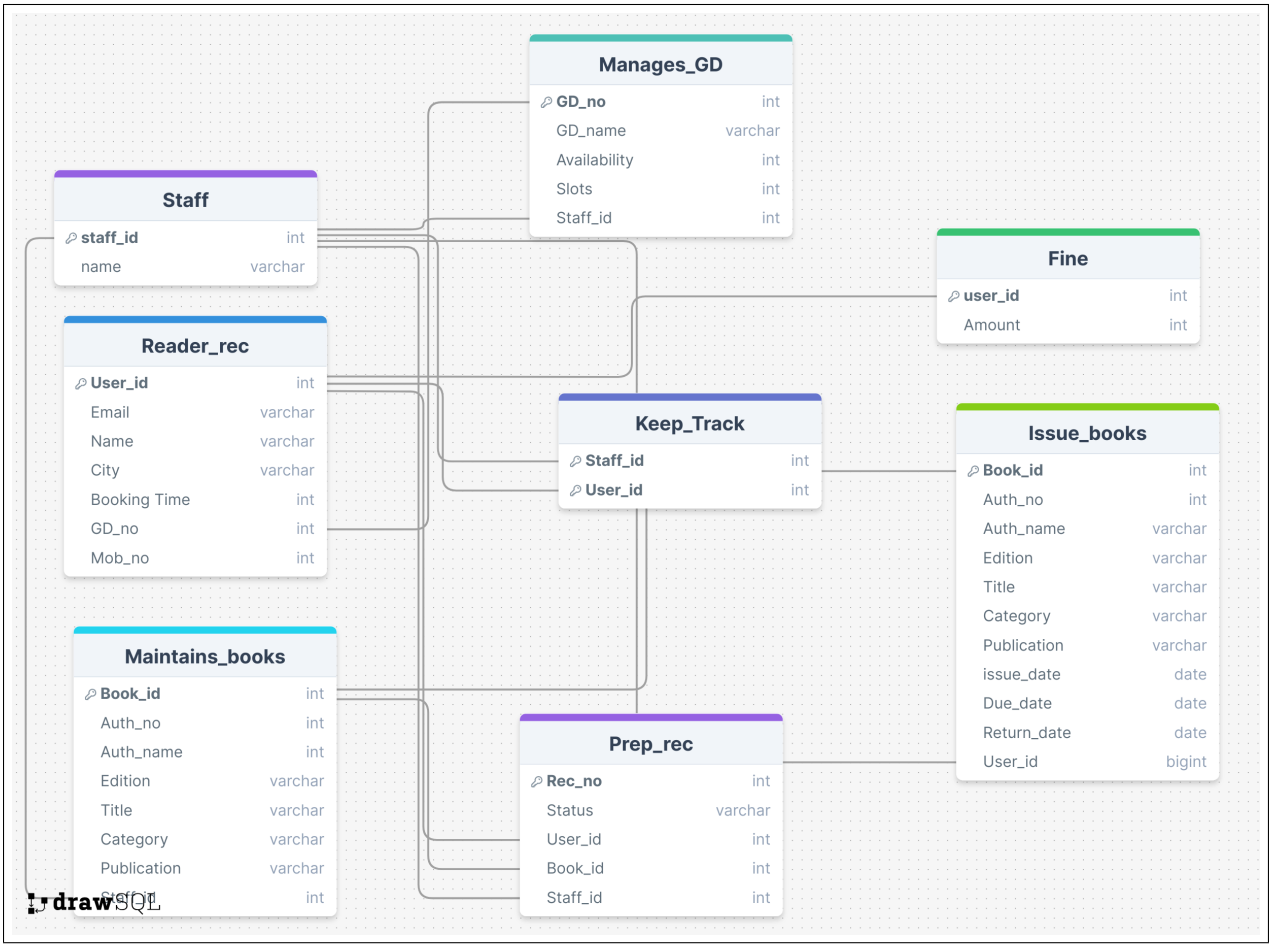
1.4 Software Requirements

- SQL
- PL/SQL
- Oracle Live SQL
- Draw.io
- DrawSQL.com

2 ER Diagram


Books
Title
Edition
AuthNo
Author_Name
Book_ID
Category
Publication
Status
Records
Activity
Rec_no
User_id
Book_ID
maintains
issue/return date
Issue_date
due date
return date
prepares
Staff
Staff_name
Staff_id
keeps track of
Readers
Email_id
User_ID
City
Phone no
name
Firstname
Lastname
has to pay
FINE
amount
manages
GD
GD Name
GD_No
Availability
slots
reserves
Booking Time
N
1
M

3 ER to table


Manages_GD
GD_no int
GD_name varchar
Availability int
Slots int
Staff_id int
Staff
staff_id int
name varchar
Fine
user_id int
Amount int
Reader_rec
User_id int
Email varchar
Name varchar
City varchar
Booking Time int
GD_no int
Mob_no int
Keep_Track
Staff_id int
User_id int
Issue_books
Book_id int
Auth_no int
Auth_name varchar
Edition varchar
Title varchar
Category varchar
Publication varchar
issue_date date
Due_date date
Return_date date
User_id bigint
Maintains_books
Book_id int
Auth_no int
Auth_name int
Edition varchar
Title varchar
Category varchar
Publication varchar
Prep_rec
Rec_no int
Status varchar
User_id int
Book_id int
Staff_id int
drawSQL

4 Normalization

Functional Dependency of ER diagram :

FD1: {Staff_id} → {name}

FD2: {Reg_no} → {Issue/Return, user_id, Book_id, Staff_id}

FD3: {user_id, booking_time} → {email_id, name, city, GD_no, phone}
{user_id} → {email_id, firstname, lastname, city, phone}

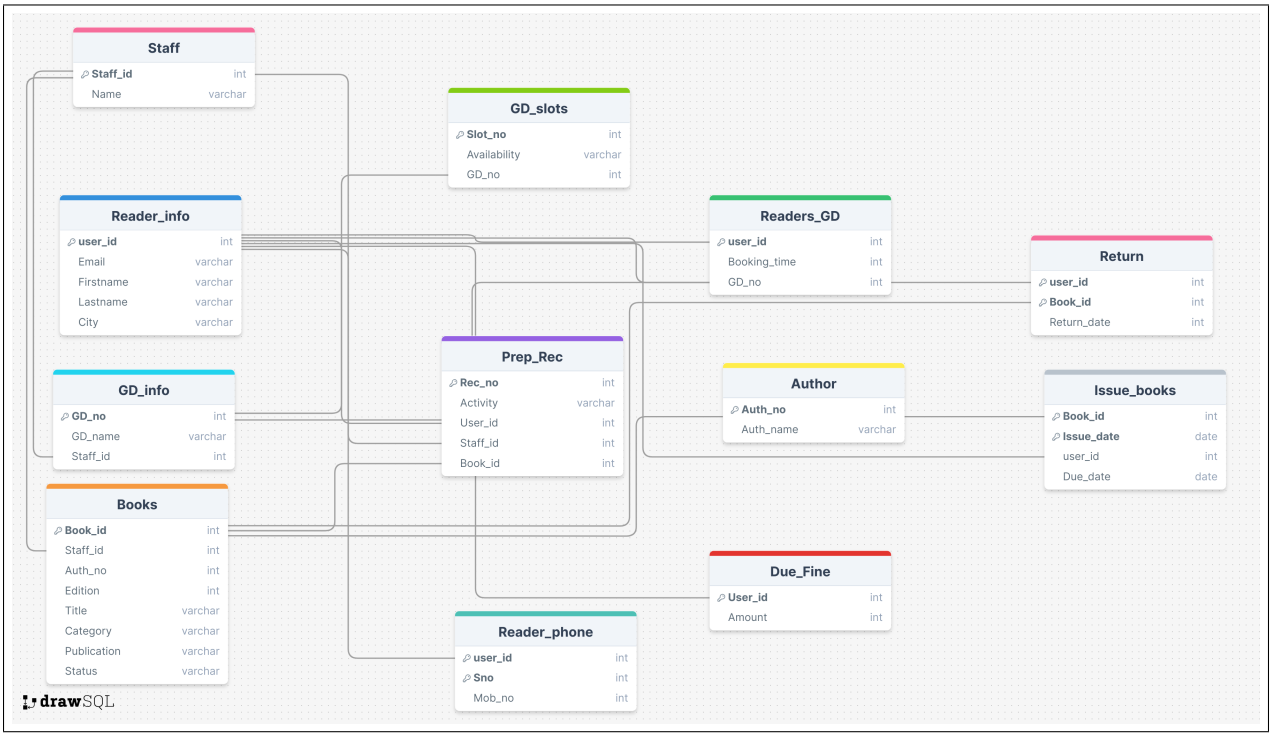
FD4: {Gd_no} → {GD_name, Av/booked, staff_id}

FD5: {book_id} → {authno, auth_name, edition, title, category, publication, staff_id}

FD6: {book_id, issue_date} → {user_id, authno, auth_name, edition, title, category, publication, return_date, due_date}
{User_id, book_id} → {return_date}
{book_id} → {authno, auth_name, edition, title, category, publication}
{issue_date} → {due_date}

Since these functional dependencies violate the normalization forms, we will have to first normalize these relational schemas into 3NF.

Normalized Relational Schema



5 SQL code and output

5.1 Creation of tables

1. Staff table

```
create table Staff(
Staff_id number(3) primary key,
Name varchar(10)
);
```

2. Author table

```
create table Author(
Auth_no number(3) primary key,
Auth_name varchar(15)
);
```

3. Books table

```
create table Books(
Book_id number(6) primary key,
Staff_id number(3) references Staff(Staff_id),
Auth_no number(3) references Author(Auth_no),
Edition number(4) ,
Title varchar(40),
Category varchar(20),
Publication varchar(30)
Status varchar(10) check (Status in ('Available','Issued'))
);
```

4. Prep_Rec table

```
create table Prep_rec(
Rec_no number(5) primary key,
Book_id number(6) references Books(Book_id),
User_id number(4) references Reader_info(User_id),
Activity varchar(1) check( Activity in ('R','I')),
Staff_id number (3) references Staff(Staff_id)
);
```

5. Reader_info table

```
create table Reader_info(
User_id number(3) primary key,
Email varchar(30),
Firstname varchar(10),
Lastname varchar(15),
city varchar(15)
);
```

6. Reader_phone table

```
create table Reader_phone(
User_id number(3) references Reader_info(User_id),
SNo number(2),
Mob_no number(10),
primary key(User_id,SNo)
);
```

7. GD_Info table

```
create table GD_Info(
Gd_no number(3) primary key,
Gd_name varchar(15),
Staff_id number(3) references Staff(Staff_id)
);
```

8. GD_Slots table

```
create table Gd_slots(
Gd_no number(3) references GD_info(Gd_no),
Slot number(2) check (Slot in (8,10,12,14,16,18)),
Availability varchar(10) check (Availability in ('Available','Reserved'))
);
```

9. Readers_GD table

```
create table Readers_gd(
User_id number(3) references Reader_info(User_id),
Booking_time number(2) check (Booking_time in (8,10,12,14,16,18)),
Gd_no number(3) references GD(Gd_no)
);
```

10. Due_fine table

```
create table Due_fine(
User_id number(3) primary key references Reader_info(User_id),
Amount number(3)
);
```

11. Issue_book table

```
create table Issue_book(
Book_id number(6) references Books(Book_id),
Issue_date date,
User_id number(3) references Reader_info(User_id),
Due_date date,
primary key(Book_id,Issue_date)
);
```

12. Return table

```
create table Return(
User_id number(3) references Reader_info(User_id),
Book_id number(6) references Books(Book_id),
Return_date date,
primary key(User_id,Book_id)
);
```

5.2 Inserting values in tables

1. Staff table

```
insert into Staff values(1,'Narmada');
insert into Staff values(2,'Sandeep');
insert into Staff values(3,'Ranjeet');
insert into Staff values(4,'Amrita');
insert into Staff values(5,'Ram');
```

```
SQL> select * from Staff order by Staff_id
  2  /

  STAFF_ID NAME
---------- ----------
         1 Narmada
         2 Sandeep
         3 Ranjeet
         4 Amrita
         5 Ram
```

2. Author table

```
insert into Author values(101,'Shivansh');
insert into Author values(102,'Soumya');
insert into Author values(103,'Bhavya');
insert into Author values(104,'Shaurya');
```

```
SQL> select * from Author order by Auth_no
  2  /

   AUTH_NO AUTH_NAME
---------- ---------------
       101 Shivansh
       102 Soumya
       103 Bhavya
       104 Shaurya
```

3. Books table

```
insert into Books values(201,5,101,2,'Uncharted','Fiction','Naughty_Dog','Issued');
insert into Books values(202,5,103,5,'Playing_with_numbers','Mathematics','APC','Available');
insert into Books values(203,5,102,4,'Fourzan','Networking','Mcgraw','Available');
insert into Books values(204,5,104,3,'Harry_Potter','Fiction','Evergreen','Available');
insert into Books values(205,5,102,9,'learn_Oracle','DBMS','Evergreen','Available');
```

BOOK_ID	STAFF_ID	AUTH_NO	EDITION	TITLE	CATEGORY	PUBLICATION	STATUS
201	5	101	2	Uncharted	Fiction	Naughty_Dog	Issued
202	5	103	5	Playing_with_numbers	Mathematics	APC	Available
203	5	102	4	Fourzan	Networking	Mcgraw	Available
204	5	104	3	Harry_Potter	Fiction	Evergreen	Available
205	5	102	9	learn_Oracle	DBMS	Evergreen	Available

4. Prep_Rec table

```
insert into Prep_rec values(1,203,2,'I',3);
insert into Prep_rec values(2,201,3,'I',3);
insert into Prep_rec values(3,203,2,'R',4);
```

```
SQL> select * from Prep_rec
  2  /

    REC_NO    BOOK_ID    USER_ID A   STAFF_ID
---------- ---------- ---------- - ----------
         1        203          2 I          3
         2        201          3 I          3
         3        203          2 R          4
```

5. Reader_info table

```
insert into Reader_info values(5,'Anushka123@gmail.com','Anushka','Agarwal','Lucknow');
insert into Reader_info values(4,'Tanisha123@gmail.in','Tanisha','Jain','Malerkotla');
insert into Reader_info values(2,'kushagr123@yahoo.in','Kushagr','Sharma','Panchkula');
insert into Reader_info values(3,'Jiya123@yahoo.in','Jiya','Gupta','Patiala');
insert into Reader_info values(1,'Shreeya123@gmail.com','Shreeya','Kesarwani','Patiala');
```

USER_ID	EMAIL	FIRSTNAME	LASTNAME	CITY
5	Anushka123@gmail.com	Anushka	Agarwal	Lucknow
4	Tanisha123@gmail.in	Tanisha	Jain	Malerkotla
2	kushagr123@yahoo.in	Kushagr	Sharma	Panchkula
3	Jiya123@yahoo.in	Jiya	Gupta	Patiala
1	Shreeya123@gmail.com	Shreeya	Kesarwani	Patiala

6. Reader_phone table

```
insert into Reader_phone values(1,1,985670883);
insert into Reader_phone values(2,3,950638013);
insert into Reader_phone values(3,4,754635013);
insert into Reader_phone values(4,5,854585215);
insert into Reader_phone values(5,2,944687719);
```

USER_ID	SNO	MOB_NO
1	1	985670883
2	3	950638013
3	4	754635013
4	5	854585215
5	2	944687719

7. GD_info table

```
insert into GD_info values(1,'SUN',1);
insert into GD_info values(2,'MOON',2);
```

```
SQL> select * from Gd_info order by Gd_no
  2  /

     GD_NO GD_NAME          STAFF_ID
---------- -------------- ----------
         1 ABC                     1
         2 DEF                     2
```

8. GD_Slots table

```
insert into GD_slots values(1,8,'Reserved');
insert into GD_slots values(1,10,'Available');
insert into GD_slots values(1,12,'Available');
insert into GD_slots values(1,14,'Available');
insert into GD_slots values(1,16,'Available');
insert into GD_slots values(1,18,'Available');

insert into GD_slots values(2,8,'Available');
insert into GD_slots values(2,10,'Available');
insert into GD_slots values(2,12,'Available');
insert into GD_slots values(2,14,'Available');
insert into GD_slots values(2,16,'Available');
insert into GD_slots values(2,18,'Available');
```

```
SQL> select * from Gd_slots
  2  /

     GD_NO       SLOT AVAILABILI
---------- ---------- ----------
         1         10 Available
         1         12 Available
         1         14 Available
         1         16 Available
         1         18 Available
         2          8 Available
         2         10 Available
         2         12 Available
         2         14 Available
         2         16 Available
         2         18 Available

     GD_NO       SLOT AVAILABILI
---------- ---------- ----------
         1          8 Reserved

12 rows selected.
```

9. Readers_GD table

```
insert into readers_gd values(1,8,1);
```

```
SQL> select * from Readers_gd
  2  /

   USER_ID BOOKING_TIME      GD_NO
---------- ------------ ----------
         1            8          1
         1            8          1
```

10. Issue_book table

```
insert into issue_book values(201,'1-may-2024',3,'31-may-2024');
```

```
SQL> select * from Issue_book
  2  /

   BOOK_ID ISSUE_DAT    USER_ID DUE_DATE
---------- --------- ---------- ---------
       201 01-MAY-24          3 31-MAY-24
```

11. Return table

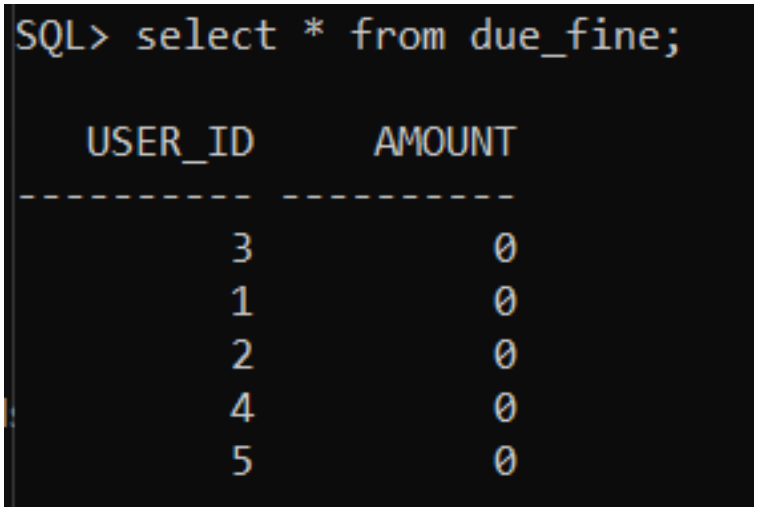
```
insert into return values(2,203,'15-march-2024);
```

```
SQL> select * from Return
  2  /

   USER_ID    BOOK_ID RETURN_DA
---------- ---------- ---------
         2        203 15-MAR-24
```

12. Due_Fine table

```
insert into due_fine values(1,0);
insert into due_fine values(2,0);
insert into due_fine values(3,0);
insert into due_fine values(4,0);
insert into due_fine values(5,0);
```



6 PL/SQL code and output

1. Search book by author name

```
Create or replace procedure Search_auth
(a_name IN Author.auth_name%type)
is
cursor c1 is
    select books.book_id, books.edition,books.title,books.category,
    books.publication,books.status from books,author
    where books.Auth_no = Author.Auth_no and author.auth_name = a_name;
BEGIN
    for rec in  c1 loop
        dbms_output.put_line('book_id : '||rec.book_id||' edition : '||rec.edition||
        ' title : '||rec.title||' category : '||
        rec.category||' publication : '||rec.publication
        ||' status : '||rec.status);
    END loop;
END;
```

```
SQL> begin
  2  search_auth('Shivansh');
  3  end;
  4  /
book_id : 201 edition : 2 title : Uncharted category : Fiction publication :
Naughty_Dog status : Issued

PL/SQL procedure successfully completed.

SQL>
```

2. Search book by book title

```
Create or replace procedure search_title
(book_title IN books.title%type)
IS
cursor c is
select book_id,edition,title,category,publication,status
from books where title = book_title;
BEGIN
    for rec in c loop
        dbms_output.put_line('book_id : '||rec.book_id||'
        edition : '|| rec.edition||' title : '||rec.title||' category : '||rec.category||
        ' publication : '||rec.publication||
        ' status : '||rec.status);
```

```
    END loop;
END;
```

```
SQL> begin
  2  search_title('Uncharted');
  3  end;
  4  /
book_id : 201 edition : 2 title : Uncharted category : Fiction publication :
Naughty_Dog status : Issued

PL/SQL procedure successfully completed.
```

3. Procedure for reserving a GD Room

```
Create or replace procedure Book_gd
(User IN Reader_info.User_id%type ,
Gd_id IN GD_info.Gd_no%type ,
Time IN GD_slots.Slot%type)
AS
temp GD_slots.Availability%type;
BEGIN
    IF Gd_id not in (1,2) THEN
        dbms_output.put_line('Enter Valid GD_id');
    ELSE
        IF Time not in (8,10,12,14,16,18) THEN
            dbms_output.put_line('Enter Valid Time');
        ELSE
            select Availability INTO temp
            from GD_slots where GD_no = Gd_id and Slot = Time;
            IF temp ='Reserved' THEN
                dbms_output.put_line('GD not available at
                the given time, try again');
            END IF;
            IF temp = 'Available' THEN
                Update GD_slots set Availability = 'Reserved'
                where GD_no = Gd_id and Slot = Time;
                insert into Readers_gd values(User,Time,Gd_id);
            END IF;
        END IF;
    END IF;
END;
/
```

4. Checking if reader can issue books

```
Create or replace function is_eligible
(id in due_fine.user_id%type)
RETURN boolean is
fine due_fine.amount%type;
BEGIN
    select amount into fine from due_fine where user_id = id;
    IF fine>0 then
        return false;
    ELSE
        return true;
    END IF;
END;
```

5. Procedure for making records of books being issued

```
Create or replace procedure book_issue
(u_id IN reader_info.user_id%type,
b_id IN books.book_id%type)
IS
S_no number;
i_date date;
d_date date;
flag boolean := is_eligible(u_id);
BEGIN
    select count(*)+1 into S_no from prep_rec;
    select sysdate into i_date from dual;
    select sysdate+30 into d_date from dual;
    UPDATE books SET status='issued' where book_id=b_id;
    insert into prep_rec values(S_no,b_id,u_id,'I',3);
    insert into issue_book values(b_id,i_date,u_id,d_date);
END book_issue;
```

6. Function for calculating fine

```
CREATE OR REPLACE FUNCTION calculate_fine
(this_due_date IN DATE,
this_return_date IN DATE)
RETURN NUMBER
IS
    this_fine NUMBER;
BEGIN
```

```
    IF this_due_date > this_return_date THEN
        this_fine := (this_due_date - this_return_date) * 5;
    ELSE
        this_fine := 0;
    END IF;

    RETURN this_fine;
END;
```

7. Procedure for returning books

```
Create or replace procedure book_return
(this_user_id IN Reader_info.User_id%type,
this_book_id IN Books.Book_id%type,
return_date IN Return.Return_date%type)
AS
this_due_date Issue_book.Due_date%type;
fine number;
BEGIN
    Select max(Due_date) into this_due_date
    from Issue_book
    where User_id = this_user_id
    AND Book_id = this_book_id;
    fine:=calculate_fine(this_due_date,return_date);
    IF(fine=0) THEN
        UPDATE Books SET Status = 'Available'
        where Book_id = this_book_id;
        DBMS_OUTPUT.PUT_LINE('Book returned before due date. No fine');
    ELSIF ( fine > 0) THEN
        UPDATE Books SET Status = 'Available'
        where Book_id = this_book_id;
        DBMS_OUTPUT.PUT_LINE('Book returned after due date. Fine applied');
        DBMS_OUTPUT.PUT_LINE('Fine: Rs' || fine);
    END IF;
    update due_fine set amount = amount + fine where user_id = this_user_id;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Book issue record not found.');
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('Error occurred: ' || SQLERRM);
END;
```

```
SQL> edit
Wrote file afiedt.buf

  1  BEGIN
  2  book_return(4,204,'06-JUN-2024');
  3* END;
  4  /

PL/SQL procedure successfully completed.

SQL> set serveroutput on
SQL> /
Book returned before due date. No fine

PL/SQL procedure successfully completed.

SQL>
```

8. Trigger for booking GD

```
CREATE OR REPLACE TRIGGER Book_gd_msg
AFTER INSERT on Readers_gd
for each row
BEGIN
    DBMS_OUTPUT.PUT_LINE('GD HAS BEEN BOOKED SUCCESSFULLY');
END;
```

```
SQL> edit
Wrote file afiedt.buf

  1  CREATE OR REPLACE TRIGGER Book_gd_msg
  2  AFTER INSERT on Readers_gd
  3  for each row
  4  BEGIN
  5  DBMS_OUTPUT.PUT_LINE('GD HAS BEEN BOOKED SUCCESSFULLY');
  6* END;
SQL> /

Trigger created.

SQL> BEGIN
  2  Book_gd(2,1,10);
  3  END;
  4  /
GD HAS BEEN BOOKED SUCCESSFULLY

PL/SQL procedure successfully completed.
```

9. Trigger for issuing Book

```
CREATE TRIGGER ISSUE_MSG
AFTER INSERT on prep_rec
for each row
BEGIN
    DBMS_OUTPUT.PUT_LINE('BOOK HAS BEEN ISSUED');
```

```
END;
```

```
SQL> edit
Wrote file afiedt.buf

  1  CREATE OR REPLACE TRIGGER ISSUE_MSG
  2  AFTER INSERT on Issue_book
  3  for each row
  4  BEGIN
  5  DBMS_OUTPUT.PUT_LINE('BOOK HAS BEEN ISSUED');
  6* END;
SQL> /

Trigger created.

SQL> BEGIN
  2  book_issue(3,201);
  3  END;
  4  /
BOOK HAS BEEN ISSUED

PL/SQL procedure successfully completed.
```

10. Procedure for resetting availability of GDs

```
Create or replace procedure reset
IS
BEGIN
    Update GD_slots set Availability = 'Available';
END;
```

7 Conclusion

Our Library Management System is designed to streamline and modernize the way libraries manage their operations, making it easier for librarians to track book inventories, manage user accounts, and oversee book issuance and returns. The system is built on PL/SQL and robust database technology, ensuring high performance and scalability to handle the complex demands of today's libraries.

With this system, libraries can efficiently manage book issuance and return processes, reducing manual effort and administrative overhead. Users benefit from a seamless experience with clear tracking of borrowed books, due dates, and fines for overdue returns. Automated notifications and reminders further enhance user satisfaction by keeping patrons informed about their account status and library-related obligations.

Our Library Management System also includes GD room booking capabilities, allowing users to reserve rooms for collaborative study or group discussions. This feature improves the utilization of library resources and facilitates a more dynamic environment for learning and interaction. The system's reporting and analytics tools offer valuable insights into library usage, enabling librarians to make informed decisions about book acquisitions and other operational aspects.

7.1 Future Scope

In the future scope of our project, these are some of the areas to focus on :

- Allowing group of users to book GD
- Notification alert to users whenever a new book is added
- Use automated systems to calculate fines based on predefined rules, with notifications sent to users via email or SMS.
- Implement RFID or barcode technology for automated book issuance and return.

8 References

1. Draw.io : *https://app.diagrams.net*
2. Oracle Live SQL : *https://livesql.oracle.com*
3. Oracle SQL : *https://docs.oracle.com/database/121/SQLRF/toc.htm*
4. Oracle PL/SQL : *https://docs.oracle.com/en/database/oracle/oracle-database/19/lnpls/index.html*