DESCRIPTION:-

An institute has a library where books are issued by an Admin to Students who in return have to pay nominal charges for borrowing and penalty charges for returning late. The library also has staff who maintain the books and refer books to students and faculties . The library receives books from different publishers all around the country. There is a limit on the number of book issues for all which can be borrowed. Before borrowing the book, Students and Faculties have to verify themselves.

ENTITY:- Student, Admin, Book, Publisher, Staff,Payments, Login

ATTRIBUTES:-

- Student: student_id, student_name, student_mobile, student_address, student_branch, student_rollno, student_books.
- Admin: admin_id, admin_name, admin_mobile, admin_email.
- Book: book_id, book_name, book_author, book_status, book_issue_date, book_renew_date, book_issue_expiry_date, book_description.
- Publisher: pub_id, pub_genre,pub_fees.
- Staff: staff_id, staff_username, staff_name, staff_address, staff_mobile, staff_post,books_referred.
- Login: login_id, password.
- Payments: amount, receipt_no, receipt_date,borrower_id.

RELATION:-

- Publisher publishes Books -1:N (Weak Entity Relation)
- Students borrow Books -1:N (Regular Entity Relation)
- Books Issued By Admin- 1:N (Regular Entity Relation)
- Staff Refers Books -M:N (Regular Entity Relation)
- Student Pays Payment-1:1 (Weak Entity Relation)
- Admin Receives Payments - 1:1(Weak Entity Relation)
- Books referred by staff (Multivalued Relation)

<u>**ER DIAGRAM:-**</u>

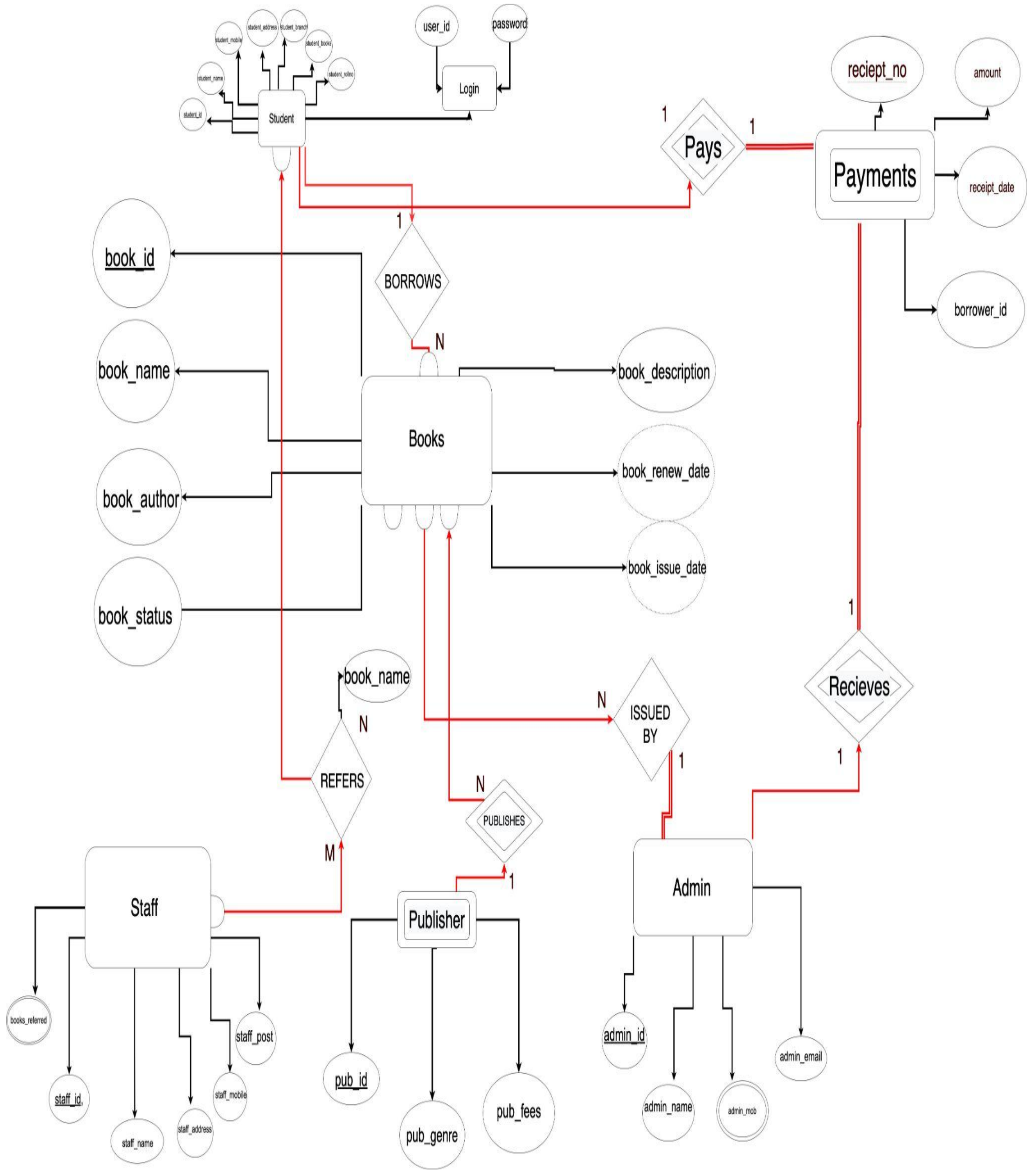
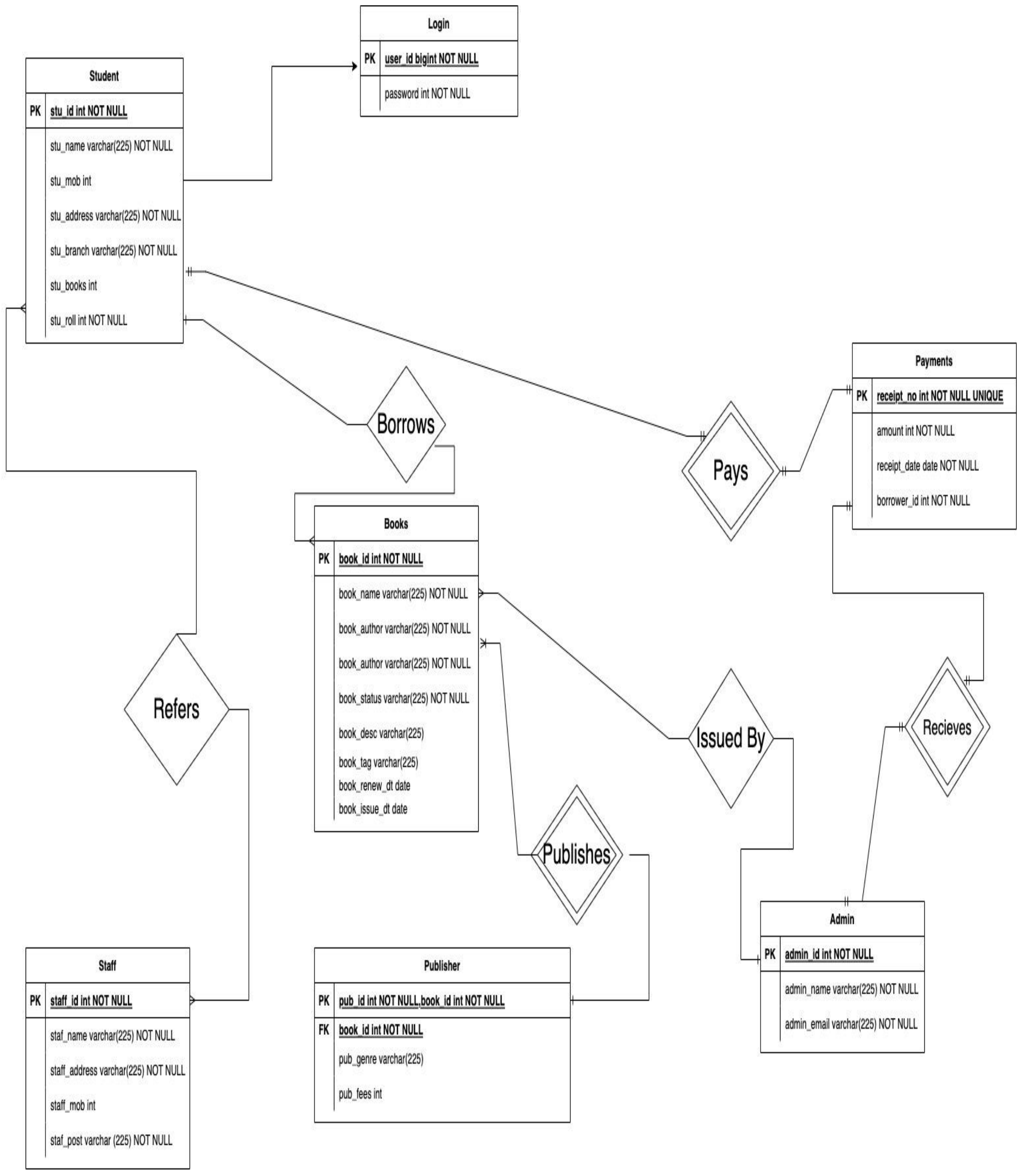

student_mobile
student_address
student_branch
student_books
student_name
student_rollno
student_id
Student
user_id
password
Login
reciept_no
amount
1
Pays
1
Payments
receipt_date
borrower_id
1
BORROWS
N
book_id
book_name
book_author
book_status
Books
book_description
book_renew_date
book_issue_date
book_name
N
REFERS
M
N
ISSUED BY
1
N
PUBLISHES
1
1
Recieves
1
Staff
books_referred
staff_id,
staff_name
staff_address
staff_mobile
staff_post
Publisher
pub_id
pub_genre
pub_fees
Admin
admin_id
admin_name
admin_mob
admin_email

TABLE:


Login
PK user_id bigint NOT NULL
password int NOT NULL
Student
PK stu_id int NOT NULL
stu_name varchar(225) NOT NULL
stu_mob int
stu_address varchar(225) NOT NULL
stu_branch varchar(225) NOT NULL
stu_books int
stu_roll int NOT NULL
Payments
PK receipt_no int NOT NULL UNIQUE
amount int NOT NULL
receipt_date date NOT NULL
borrower_id int NOT NULL
Borrows
Pays
Books
PK book_id int NOT NULL
book_name varchar(225) NOT NULL
book_author varchar(225) NOT NULL
book_author varchar(225) NOT NULL
book_status varchar(225) NOT NULL
book_desc varchar(225)
book_tag varchar(225)
book_renew_dt date
book_issue_dt date
Refers
Issued By
Recieves
Publishes
Admin
PK admin_id int NOT NULL
admin_name varchar(225) NOT NULL
admin_email varchar(225) NOT NULL
Staff
PK staff_id int NOT NULL
staf_name varchar(225) NOT NULL
staff_address varchar(225) NOT NULL
staff_mob int
staf_post varchar (225) NOT NULL
Publisher
PK pub_id int NOT NULL,book_id int NOT NULL
FK book_id int NOT NULL
pub_genre varchar(225)
pub_fees int

<u>By Performing 1NF:</u>

1NF eliminates multivalued attributes and their combinations.

In ER diagram relation **Staff** has an attribute called **book_ref_id** which is multivalued and **Admin** has an attribute called **admin_mob** which is also multivalued.It is necessary because it removes the unnecessary repeating values by creating a separate table i.e Referred and AdminMob and hence does not create issue while inserting/deleting/updating the values and hence these anomalies are removed. **Non atomic values** are those in which each set is considered as a single attribute domain as **book_ref_id** in Staff and **admin_mob** in AdminMob.

<u>Functional Dependency and candidate keys and super keys:</u>

- Candidate key for Admin: admin_id
  Candidate key for Staff: staff_id

- Super key for Admin:
  ({admin_id,admin_name},{admin_id,admin_email},{admin_id,admin_mobile},{admin_id, admin_email,admin_name},{admin_id,admin_name,admin_email,admin_mobile})

  Super key for Staff:
  ({staff_id,staff_name},{staff_id,staff_email},{staff_id,staff_mobile},{staff_address,staff_id}, {staff_id, staff_post},{staff_id,staff_name,staff_email},{staff_id,staff_email,staff_address}, {staff_id,staff_email,staff_post,staff_mobile})

- Dependency for Admin: (admin_id ->admin_email, admin_id ->admin_mobile, admin_id ->admin_name, admin_id ->(admin_email,admin_mobile).

  Dependency for Staff: (staff_id ->staff_email, staff_id ->staff_mobile, staff_id ->staff_name, staff_id -> staff_address, staff_id ->(staff_email,staff_mobile), staff_id ->(staff_mobile,staff_name), staff_id ->(staff_email,staff_mobile,staff_address), staff_id -> (staff_mobile,staff_name,staff_address, staff_email).

<u>Decomposing the tables:</u>

The tables Admin and Staff are decomposed to 2 tables each namely **Admin and AdminMob** for Admin and Staff to **Staff and Referred.**

- CREATE TABLE Referred(staff_id int NOT NULL, book_ref_id int, PRIMARY KEY(staff_id,book_ref_id));
- CREATE TABLE AdminMob(admin_id int NOT NULL, admin_mob int NOT NULL, PRIMARY KEY(admin_id,admin_mob));

<u>Performing 2NF:</u>

Second Normal Form applies to relations with composite keys, that is, relations with a primary key composed of two or more attributes. A relation with a single-attribute primary key is automatically in at least 2NF. A relation that is not in 2NF may suffer from the update anomalies.

In **Publisher,**
genre_fee cannot alone decide the value of publisher_genre or pub_id;
genre_fee together with pub_id cannot decide the value of publisher_genre;
genre_fee together with publisher_genre cannot decide the value of pub_id;
Hence,
genre_fee would be a non-prime attribute, as it does not belong to the one only candidate key {pub_id, publisher_genre} ;
But, publisher_genre -> genre_fee, i.e., genre_fee is dependent on publisher_genre, which is a proper subset of the candidate key. Non-prime attribute genre_fee is dependent on a proper subset of the candidate key, which is a partial dependency and so this relation is not in 2NF.
To convert the above relation to 2NF,
we need to split the table into two tables such as :
Table 1: pub_id, publisher_genre
Table 2: publisher_genre, genre_fee

CREATE TABLE Pub1(pub_id int NOT NULL, pub_genre varchar(225) NOT NULL, PRIMARY KEY(pub_id));

CREATE TABLE Pub2(pub_genre varchar(225) NOT NULL,genre_fees int NOT NULL PRIMARY KEY(pub_genre));

<u>Performing 3NF:</u>

A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.

In **Staff,**
FD set:
{staff_id -> staff_name, staff_id ->staff_mobile, staff_name ->staff_address, staff_id -> staff_post}
Candidate Key:
{staff_id}

For this relation in the table , staff_id -> staff_name and staff_name -> staff_address are true. So staff_address is transitively dependent on staff_id. It violates the third normal form. To convert it in third normal form, we will decompose the relation Staff (staff_id, staff_name, staff_mobile, staff_address,staff_post) as:
Staff (staff_id, staff_name, staff_mobile, staff_post)
Staff_Add (name, address)

Using BCNF:

The tables are already in bcnf and there are no discrepancies. All the prime attributes are dependent on attributes which are primes themselves and thus it occurs that there is no BCNF error.

The Staff was converted in 3NF and thus it is not in BCNF.