**LIBRARY MANAGEMENT SYSTEM**

**(GROUP NO 22)**

NAVNEET SINGLA-2021A7PS1450P

DHRUV KAMRA -2021A7PS0713P

**Google drive link for videos: https://drive.google.com/drive/folders/1Cv3A1HhJs6\_1yJQHPjEYgiS8PpYhduy1?usp=sharing**

**Index**

1. Introduction
2. Relations
3. Normalization
4. ER-Diagram

***Introduction***

The Database Library System is intended to Automate the library activities such as creating a new borrower, giving books to the borrowers, maintaining the details of all the item that were available in the books . This also helps the librarians by providing information such as total copies available each book, list of books available in the library and store and retrieve records related to book borrowing.

***Relations***

**Book**

**Candidate key: {Book\_id}**

* Book\_id(PK)
* author\_name
* pub\_name
* book\_title
* price
* ISBN
* No\_of\_copies

**User**

**Candidate key:{card\_no}**

* card\_no(PK)
* user\_name
* user\_address
* user\_contact

**Fine**

**Candidate Key:{fine\_id}**

* fine\_id(PK)
* record\_id
* card\_no
* amount
* reason

**Record**

**Candidate Key:{record\_id}**

* record\_id(PK)
* checkout\_date
* due\_date
* book\_id

**Final\_record**

* record\_id(PK)
* return\_date
* book\_id

***Normalization***

**Normalization for Book table:**

Let

**a=book\_id,b=author\_name,c=pub\_name,d=book\_title,e=price,f=ISBN,g=No\_of\_Copies**

**FD’S are as follows:**

**a->e,f,g**

**f->b,d,c**

**2NF**

Finding all candidate keys. The candidate keys are {**book\_id**}, The set of key attributes are: {**book\_id**}

For each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes

checking FD: a --> e,f,g

checking FD: f --> b,d,c

**It follows 2NF rule.**

**3NF**

find all candidate keys. The candiates keys are { a}, The set of key attributes are: { a }

for each FD, check whether the LHS is superkey or the RHS are all key attributes

checking functional dependency a --> e,f,g

checking functional dependency f --> b,d,c

**The above FD violates definition of 3NF**: it is non-trivial, LHS is not superkey, RHS contains a non-key attribute.

**BCNF**

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

The FD f --> b,d,c is non-trivial and its LHS is not a superkey. **It violates BCNF.**

Normalize to BCNF

**Step 1**. Find merged minimal cover of FDs, which contains:

a --> e,f,g

f --> b,d,c

Initially rel[1] contains the original table, with the FDs above

**Round1:** Checking whether table rel[1] is in BCNF

The FD [f --> b,d,c] violates BCNF as the LHS is not superkey. Table is split into the two below:

rel[2]= (f,b,d,c )

With FDs:

rel[3]= (a,e,f,g )

With FDs:

**Round2**: Checking whether table rel[2] is in BCNF

Table rel[2] is in BCNF already, send it to output

**Round3:** Checking whether table rel[3] is in BCNF

Table rel[3] is in BCNF already.

**Normalization for User table:**

Let

a=card\_no,b=user\_name,c=user\_address,d=user\_contact

**FD’S are as follows:**

a->b,c,d

## Check Normal Form

**2NF**

find all candidate keys. The candidates keys are {a}, The set of key attributes are: { a }

for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes

checking FD: a --> b,c,d. **Its follows 2NF rule.**

**3NF**

find all candidate keys. The candidate keys are {a}, The set of key attributes are: { a }

for each FD, check whether the LHS is superkey or the RHS are all key attributes

checking functional dependency a --> b,c,d. **It follows 3NF rule.**

**BCNF**

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey. **It follows BCNF principle.**

**Normalization for Fine table:**

**Let**

a=fine\_id,b=record\_id,c=card\_no, d=amount, e=reason

**FD’S are as follows:**

a->b,c,d,e

## Check Normal Form

**2NF**

find all candidate keys. The candidates keys are {a}, The set of key attributes are: { a }

for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes

checking FD: a --> b,c,d,e. **Its follows 2NF rule.**

**3NF**

find all candidate keys. The candidate keys are {a}, The set of key attributes are: { a }

for each FD, check whether the LHS is superkey or the RHS are all key attributes

checking functional dependency a --> b,c,d,e. **It follows 3NF rule.**

**BCNF**

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey. **It follows BCNF principle.**

**Normalization for Record table:**

**Let**

a=record\_id,b=checkout\_date,c=due\_date, d=book\_id

**FD’S are as follows:**

a->b,c,d

## Check Normal Form

**2NF**

find all candidate keys. The candidates keys are {a}, The set of key attributes are: { a }

for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes

checking FD: a --> b,c,d. **Its follows 2NF rule.**

**3NF**

find all candidate keys. The candidate keys are {a}, The set of key attributes are: { a }

for each FD, check whether the LHS is superkey or the RHS are all key attributes

checking functional dependency a --> b,c,d. **It follows 3NF rule.**

**BCNF**

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey. **It follows BCNF principle.**

**Normalization for Fine table:**

**Let**

a=record\_id,b=due\_date,c=book\_id

**FD’S are as follows:**

a->b,c

## Check Normal Form

**2NF**

find all candidate keys. The candidates keys are {a}, The set of key attributes are: { a }

for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes

checking FD: a --> b,c. **Its follows 2NF rule.**

**3NF**

find all candidate keys. The candidate keys are {a}, The set of key attributes are: { a }

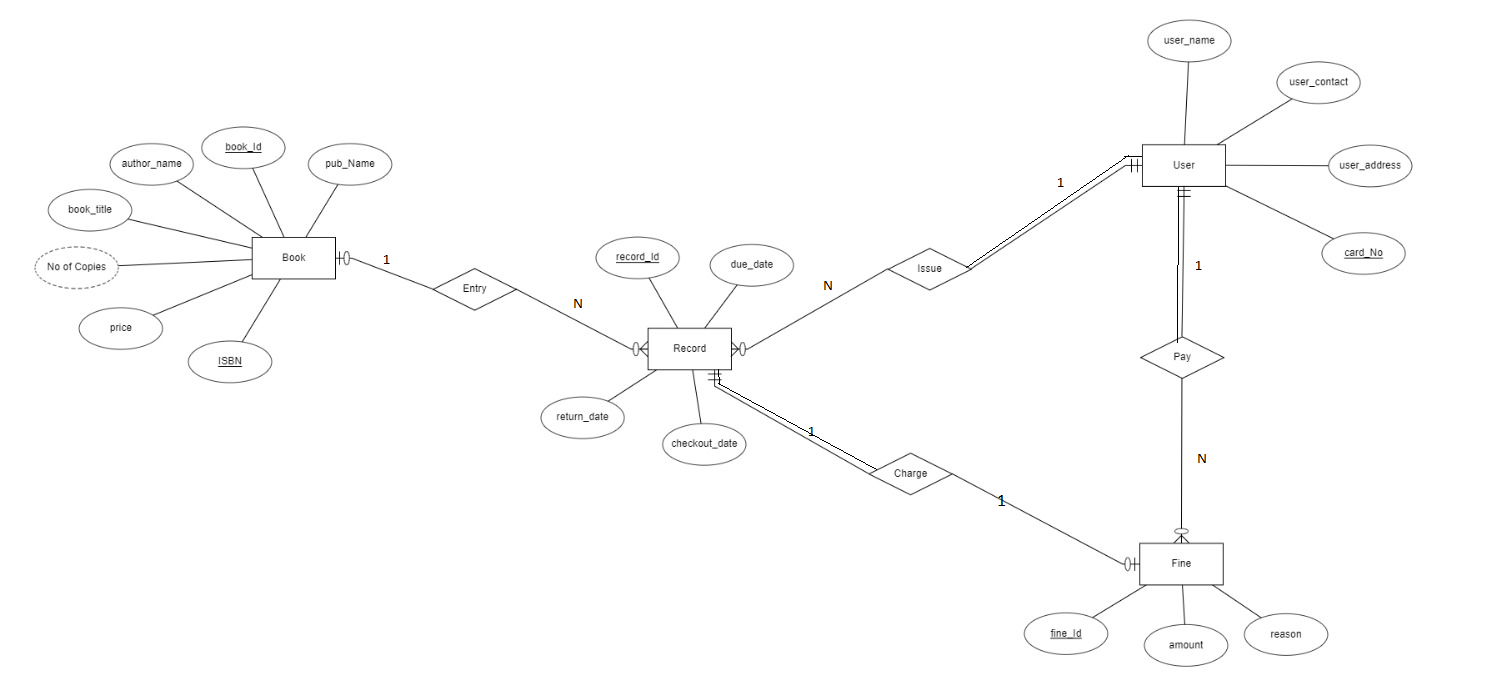
for each FD, check whether the LHS is superkey or the RHS are all key attributes

checking functional dependency a --> b,c. **It follows 3NF rule.**

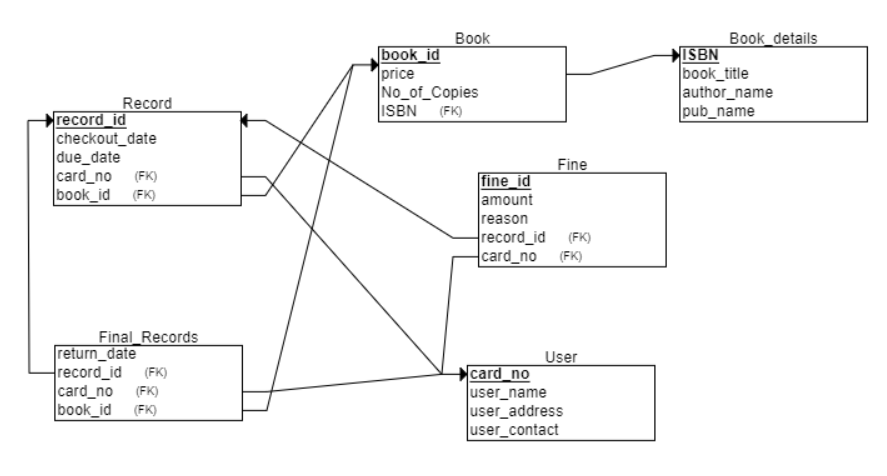
**BCNF**

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey. **It follows BCNF principle.**

**ER DIAGRAM AND RELATIONAL MODEL**

****

**ER DIAGRAM FOR THIS DESIGN**



**RELATIONAL SCHEMA FOR THIS DESIGN**

***SQL Queries (create,select, update, insert, delete)***

1. Create table Book

CREATE TABLE Book (

  book\_id INT PRIMARY KEY AUTO\_INCREMENT,

  ISBN VARCHAR(13) UNIQUE NOT NULL,

  price INT NOT NULL,

  num\_copies INT NOT NULL

);

2.Create table Book\_Details

CREATE TABLE Book\_Details (

  ISBN varchar(13) PRIMARY KEY,

  author\_name varchar(255) NOT NULL,

  book\_title varchar(255) NOT NULL,

  publisher\_name varchar(255) NOT NULL

);

3.Create table Record

CREATE TABLE Record (

    record\_id INT AUTO\_INCREMENT PRIMARY KEY,

    checkout\_date DATE NOT NULL,

    due\_date DATE NOT NULL,

    book\_id INT NOT NULL,

    card\_no INT NOT NULL,

    FOREIGN KEY (book\_id) REFERENCES Book(book\_id),

    FOREIGN KEY (card\_no) REFERENCES User(card\_no)

);

4.Create table Final\_Record

CREATE TABLE Final\_Record (

    record\_id INT NOT NULL,

    return\_date DATE NOT NULL,

    book\_id INT NOT NULL,

    card\_no INT NOT NULL,

    PRIMARY KEY (record\_id),

    FOREIGN KEY (record\_id) REFERENCES Record(record\_id),

    FOREIGN KEY (book\_id) REFERENCES Book(book\_id),

    FOREIGN KEY (card\_no) REFERENCES User(card\_no)

);

5.Create table User

CREATE TABLE User (

    card\_no INT PRIMARY KEY AUTO\_INCREMENT,

    user\_name VARCHAR(50) NOT NULL,

    user\_address VARCHAR(100) NOT NULL,

    user\_contact VARCHAR(10) NOT NULL

);

6.Create table Fine

CREATE TABLE Fine (

record\_id INT NOT NULL,

    fine\_id INT NOT NULL AUTO\_INCREMENT,

    amount INT NOT NULL DEFAULT 0,

    reason VARCHAR(255),

    card\_no INT NOT NULL,

    PRIMARY KEY (fine\_id),

    FOREIGN KEY (record\_id) REFERENCES Record(record\_id),

    FOREIGN KEY (card\_no) REFERENCES User(card\_no)

);

7. Add Primary Keys to all tables to ensure unique identification of records

Alter TABLE Book ADD PRIMARY KEY(book\_id)

Alter TABLE Book\_details ADD PRIMARY KEY(ISBN)

Alter TABLE User ADD PRIMARY KEY(card\_no)

Alter TABLE Fine ADD PRIMARY KEY(fine\_id)

Alter TABLE Record ADD PRIMARY KEY(record\_id)

Alter TABLE Final\_Record ADD PRIMARY KEY(record\_id)

ADDING CHECK CONSTRAINT(NON-NULL) TO REQUIRED ATTRIBUTES

1. Alter table Records add constraint check book\_id CHECK(book\_id is not NULL)
2. Alter table Records add constraint check card\_no CHECK(card\_no is not NULL)
3. Alter table Fine add constraint check book\_id CHECK(book\_id is not NULL)
4. Alter table Fine add constraint check card\_no CHECK(card\_no is not NULL)
5. Alter table Fine add constraint check amount CHECK(amount>0)
6. Alter table Book\_details add constraint check book\_title CHECK(book\_title is not NULL)
7. Alter table Book\_details add constraint check author\_name CHECK(author\_name is not NULL)
8. Alter table User add constraint check user\_name CHECK(user\_name is not NULL)
9. Alter table User add constraint check user\_address CHECK(user\_address is not NULL)
10. Alter table Records add constraint check checkout\_date CHECK(checkout\_date is not NULL)
11. Alter table Final\_Record add constraint check return\_date CHECK(return\_date is not NULL)
12. Alter table records add constraint check book\_id CHECK(book\_id is not NULL)
13. Alter table records add constraint check book\_id CHECK(book\_id is not NULL)

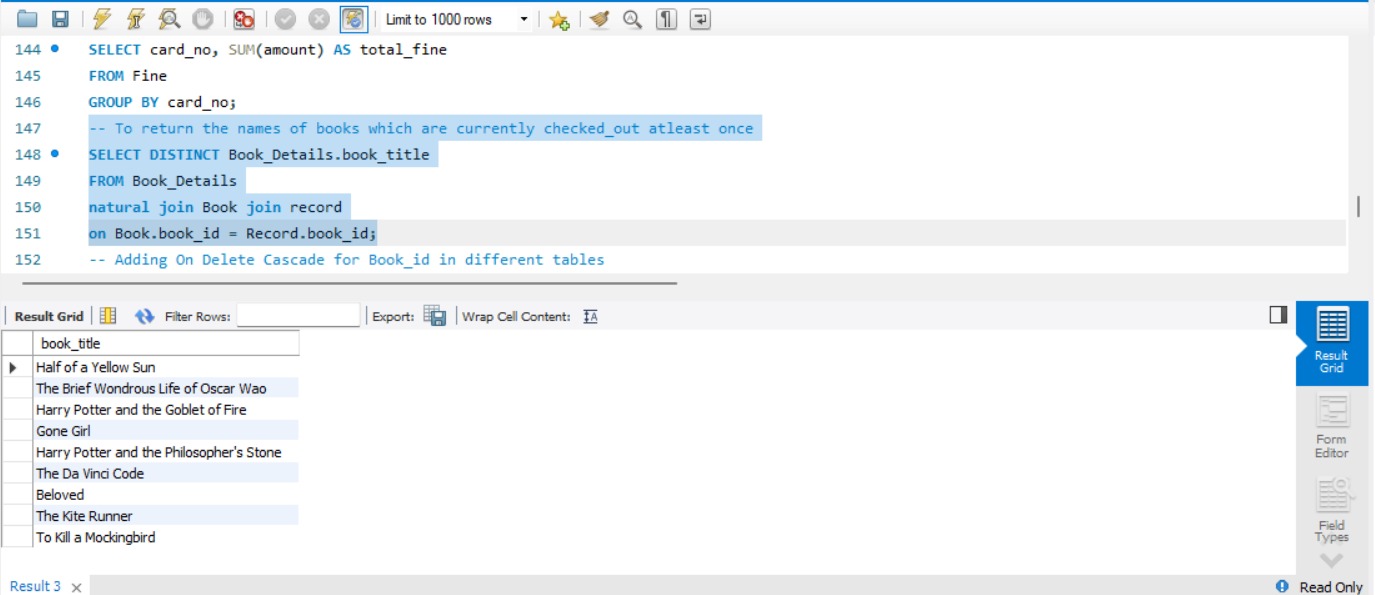
21.WRITE QUERY TO SELECT ALL BOOKS THAT ARE CURRENTLY CHECKEDOUT

SELECT DISTINCT Book\_Details.book\_title

FROM Book\_Details

natural join Book join record

on Book.book\_id = Record.book\_id;

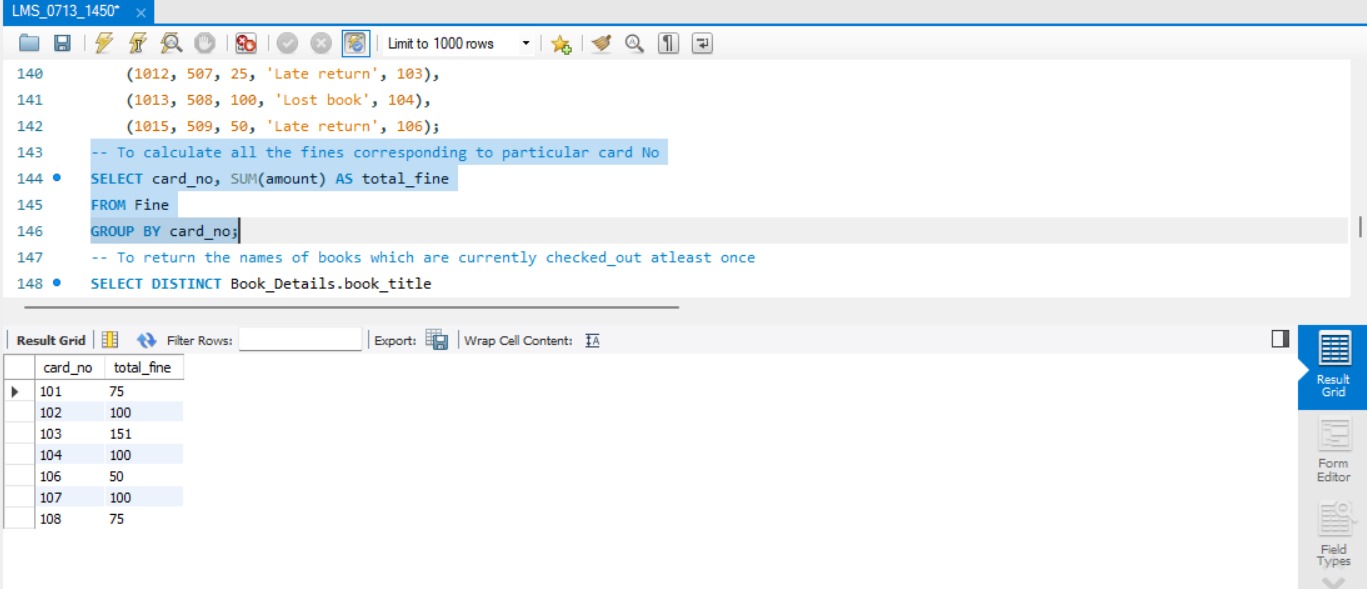


22.WRITE A QUERY TO CALCULATE THE TOTAL FINES FOR A USER

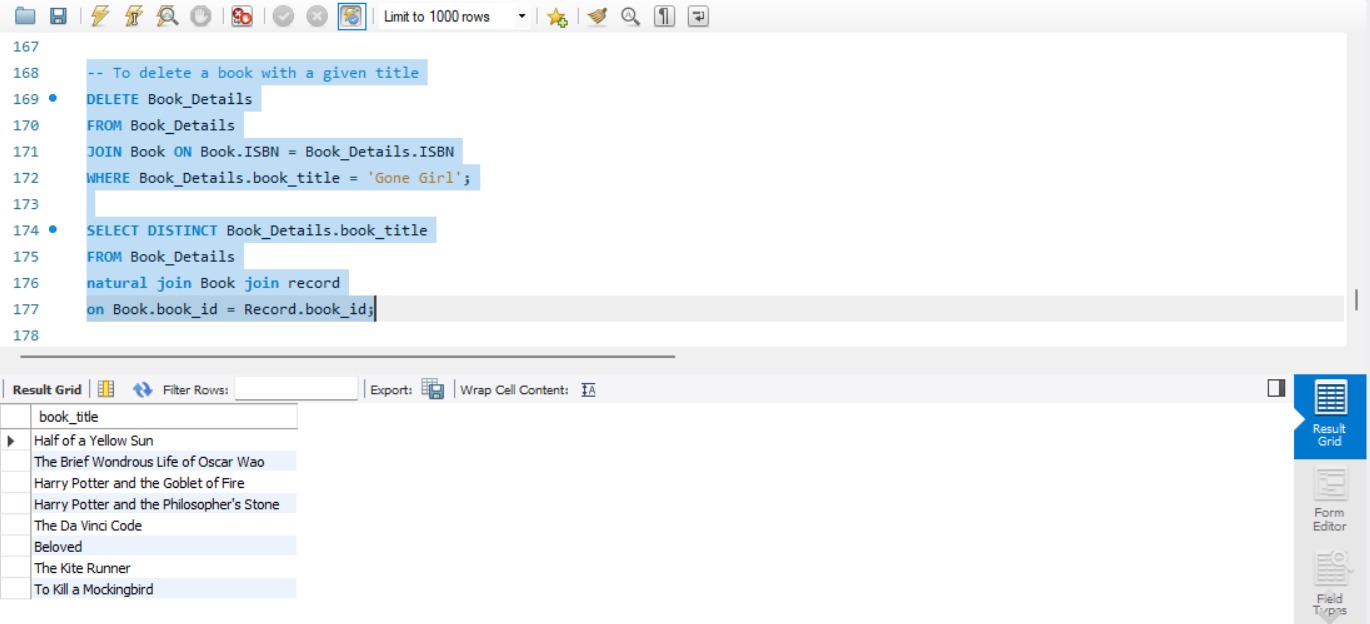
SELECT card\_no, SUM(amount) AS total\_fine

FROM Fine

GROUP BY card\_no;



23. Write a query to delete a book with given title



24. Write a query to display book name issued by a user

