Project Report

## About the Project:

This database manages a library system with multiple branches across different cities. It facilitates borrowing of various resources including books, newspapers, films, and more. The system tracks book details such as author, publisher, issue, and return dates, along with employee salaries.

# Part A: ER Diagram:

A diagram of a computer

Description automatically generated

Writer Table: Stores information about authors including their name, gender, and qualification.

Publisher Table: Contains details about book publishers such as their name, city, and ZIP code.

Books Table: Manages records of books including their name, ISBN, writer ID, and publisher ID.

Branch Table: Stores information about library branches, including their city and street.

Reader Table: Holds data about library readers, including their name, email, phone number, city, and ZIP code.

Loaned Table: Tracks loaned books, linking reader IDs and book IDs to branch IDs, loan dates, and return dates.

The “Publisher-Books” and “Writer-Books” relationships have been combined into the Books relations since they are many-one. A foreign key has been added to the Books entity corresponding to the IDs of Publisher and Writer entities.

# Part B: Relational Schema

**Schemas/Tables in the Library Database:**

- Writer (ID, Name, Gender, Qualification)

- Publisher (ID, Name, City, Zip)

- Books (ID, Name, ISBN, Write\_ID, Publisher\_ID)

- Branch (ID, City, Street)

- Reader (ID, Name, Email, Phone, City, Zip)

- Loaned (Reader\_ID, Book\_ID, Branch\_ID, Loan\_Date, Return\_Date)

**A diagram of a data flow

Description automatically generated**

**Functional Dependencies:**

Here are the list of functional dependencies that exist within the above tables. These are the functional dependencies that are expected to be held in real-world scenario.

#### Books:

ID → Name, Writer\_ID, Publisher\_ID

In this context, the "ID" attribute serves as the super key, ensuring that the database is in Boyce-Codd Normal Form (BCNF), thereby maintaining optimal structural integrity.

Writer:

ID → Name, Gender, Qualification

In this context, the "ID" attribute serves as the super key, ensuring that the database is in Boyce-Codd Normal Form (BCNF), thereby maintaining optimal structural integrity.

Publisher:

ID → Name, City, Zip

In this context, the "ID" attribute serves as the super key, ensuring that the database is in Boyce-Codd Normal Form (BCNF), thereby maintaining optimal structural integrity.

Branch:

ID → City, Street

In this context, the "ID" attribute serves as the super key, ensuring that the database is in Boyce-Codd Normal Form (BCNF), thereby maintaining optimal structural integrity.

Customer:

ID → Name, Email, Phone, City, Zip

In this context, the "ID" attribute serves as the super key, ensuring that the database is in Boyce-Codd Normal Form (BCNF), thereby maintaining optimal structural integrity.

Loaned:

Reader\_ID, Book\_ID → Branch\_ID, Loan\_Date, Return\_Date

In the above table, the Branch\_ID is not part of the composite primary key because each loan entry (Reader\_ID, Book\_ID) is associated with a specific branch through the Branch\_ID attribute.

However, including Branch\_ID as part of the composite primary key would imply that each loan record is uniquely identified by the combination of Reader\_ID, Book\_ID, and Branch\_ID, which may not be accurate.

Including Branch\_ID in the primary key would suggest that the same book borrowed by the same reader from different branches constitutes different loan entries, which is not the desired behavior.

Instead, Branch\_ID serves as a foreign key in the Loaned table, establishing a relationship with the Branch table and indicating the branch from which the book was borrowed. This design allows for flexibility in tracking loans across different branches while maintaining accurate data representation in the Loaned table.

Hence, the Reader\_ID, Book\_ID attribute serves as the super key, ensuring that the database is in Boyce-Codd Normal Form (BCNF), thereby maintaining optimal structural integrity.

Now let us check if the above tables are in third normal form.

The third normal form (3NF) requires that a relation is in 2NF and that no non-prime attribute is transitively dependent on the primary key. In simpler terms, it means that all attributes in a table must depend only on the primary key and nothing else.

Books:

The attributes Name, Writer\_ID, and Publisher\_ID are dependent on the primary key ID.

All non-prime attributes (Name, Writer\_ID, Publisher\_ID) are directly dependent on the primary key.

Therefore, Books is in 3NF.

#### Writer:

The attributes Name, Gender, and Qualification are dependent on the primary key ID.

All non-prime attributes (Name, Gender, Qualification) are directly dependent on the primary key.

Therefore, Writer is in 3NF.

#### Publisher:

The attributes Name, City, and Zip are dependent on the primary key ID.

All non-prime attributes (Name, City, Zip) are directly dependent on the primary key.

Therefore, Publisher is in 3NF.

#### Branch:

The attributes City and Street are dependent on the primary key ID.

All non-prime attributes (City, Street) are directly dependent on the primary key.

Therefore, Branch is in 3NF.

#### Reader:

The attributes Name, Email, Phone, City, and Zip are dependent on the primary key ID.

All non-prime attributes (Name, Email, Phone, City, Zip) are directly dependent on the primary key.

Therefore, Reader is in 3NF.

#### Loaned:

The attributes Branch\_ID, Loan\_Date, and Return\_Date are dependent on the composite primary key (Reader\_ID, Book\_ID).

All non-prime attributes (Branch\_ID, Loan\_Date, Return\_Date) are directly dependent on the composite primary key (Reader\_ID, Book\_ID).

Therefore, Loaned is in 3NF.

In conclusion, all tables in the given schema appear to satisfy the conditions of the third normal form (3NF).

# Part C: Creating the Database using SQL

### Creating the Tables and Inserting the Data**:**

I used the following SQL queries to create the tables in the mysql database.

Publisher Table

create table Publisher

(

ID integer,

NAME varchar(100) not null,

CITY varchar(50),

ZIP varchar(5),

PRIMARY KEY(ID)

-- CONSTRAINT "Publisher\_ID\_PK" PRIMARY KEY ("ID") ENABLE,

-- CONSTRAINT "Publisher\_Zip\_CHK" CHECK ( "ZIP" like '\_\_\_\_\_') ENABLE

);

Writer Table

create table Writer

(

ID integer,

Name varchar(100) not null,

Age numeric(10),

Gender char(1) not null,

Qualification varchar(50),

PRIMARY KEY(ID)

-- constraint "Gender\_che" check (Gender in ('M', 'F')),

-- constraint "Author\_ID\_PK" primary key (ID)

);

Books Table

create table Books

(

ID integer,

Name varchar(100),

ISBN varchar(30) not null,

Writer\_ID integer,

Publisher\_ID integer,

No\_of\_Books numeric(10, 0),

PRIMARY KEY(ID),

FOREIGN KEY(Publisher\_ID) REFERENCES Publisher(ID) ON DELETE CASCADE,

FOREIGN KEY(Writer\_ID) REFERENCES Writer(ID) ON DELETE CASCADE

);

Branch Table

create table Branch

(

ID integer,

Street varchar(50),

City varchar(50) not null,

PRIMARY KEY(ID)

-- constraint "Branch\_ID\_PK" primary key (ID),

-- constraint "Branch\_ID\_CHK" check (ID like '\_\_\_')

);

Reader Table

create table Reader

(

ID integer,

Name varchar(100) not null,

Email varchar(100),

Phone varchar(11),

City varchar(50),

Zip varchar(4),

PRIMARY KEY(ID)

);

Loaned Table

create table Loaned

(

Reader\_ID integer not null,

Book\_ID integer not null,

Branch\_ID integer not null,

Issue\_Date date,

Return\_Date date,

PRIMARY KEY(Reader\_ID, Book\_ID),

FOREIGN KEY(Reader\_ID) REFERENCES Reader(ID) ON DELETE CASCADE,

FOREIGN KEY(Book\_ID) REFERENCES Books(ID) ON DELETE CASCADE,

FOREIGN KEY(Branch\_ID) REFERENCES Branch(ID) ON DELETE CASCADE

);