[Title]

[Author]

[Institution]

Author Note

[Grant/funding info and complete correspondence address.]

**A LIBRARY DATABASE SYSTEM**

**Task(2): DESCRIPTION OF SYSTEM**

**data independence**

Data Independence is defined as a property of DBMS that helps you to change the library Database schema at one level of a database system without requiring to change the schema at the next higher level. Data independence helps you to keep data separated from all programs that make use of it You can use this stored data for computing and presentation. In many systems, data independence is an essential function for components of the library system.

Problem: library system may have data redundancy because of data independence

**data consistency**

Data consistency is the process of keeping information uniform in a library system as it moves across a network and between various applications on a computer. There are typically three types of data consistency: point in time consistency to save the data loss in power in Library and application consistency database application works well in the Library Ensuring that a computer network has all elements of data consistency covered is the best way to ensure that data of Library is not lost or corrupted as it travels throughout the system. In the absence of data consistency, there are no guarantees that any piece of information on the system is uniform across the breadth of the computer network in a library.

Problem: data may be hard to add or make a backup and run the database of the library or add new data in the database

**data integrity**

## Data integrity is the overall accuracy, completeness, and consistency of data. Data integrity also refers to the safety of data in regard to regulatory compliance and security. It is maintained by a collection of processes, rules, and standards implemented during the design of the Library system phase.

## Problem: bad process to maintain or bad rules or bad design

## Types of data integrity

Maintaining data integrity requires an understanding of the two types of data integrity: physical integrity and logical integrity. Both are collections of processes and methods that enforce data integrity in both hierarchical and relational databases.

### **Physical integrity**

Physical integrity is the protection of the wholeness and accuracy of that data in the library system as it’s stored and retrieved. When natural disasters strike, power goes out, or hackers disrupt database functions, physical integrity is compromised. Human error, storage erosion, and a host of other issues can also make it impossible for data processing managers.

**data security**

Database security refers to the various measures organizations take to ensure their databases are protected from internal and external threats. Database security includes protecting the database itself, the data it contains, its database management system, and the various applications that access it

Eg. Students cant use the library database to take books

Problem: hacker can get in and erase the data of 100 or 1000 books

**the scale of data sharing.**

Sharing the full data sets underlying the results in your article brings many benefits. It enables reuse, reduces research waste, and promotes collaboration. Greater transparency increases trust in research results by allowing results to be independently verified. These benefits lead to a more reliable evidence base and a healthier world.

Example

One campus can use other campus data to see or take books that are used and will be useful to them

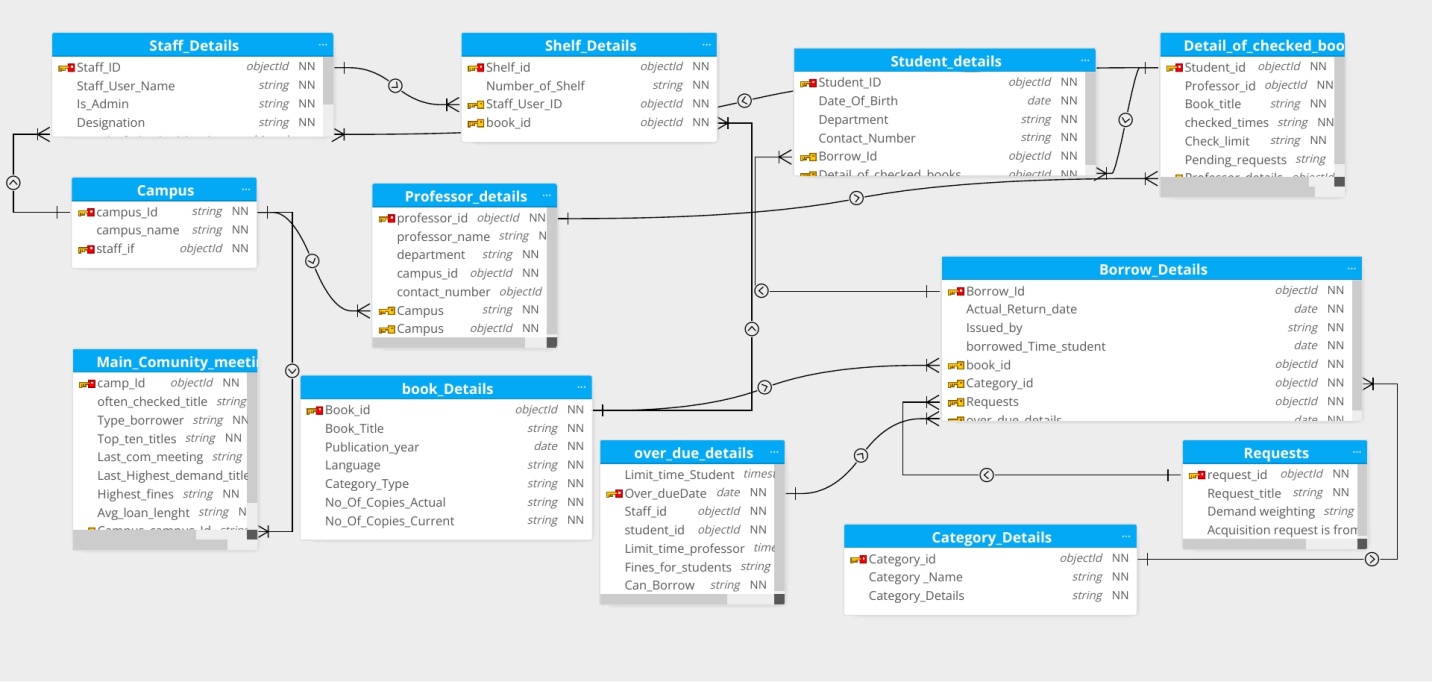
Problem: must be shared on a small scale otherwise it will make an issue

**data redundancy**

Data redundancy is a condition created within a database or data storage technology in which the same piece of data is held in two separate places This can mean two different fields within a single database or two different spots in multiple software environments or platforms. Whenever data is repeated, it basically constitutes data redundancy Data redundancy can occur by accident but is also done deliberately for backup and recovery purposes.

# **If we have a new database and we have to make the backup on the library system we will make it using data redundancy**

**Task(3): ERD OF LIBRARY MANAGEMENT SYSTEM**

****

**Book\_Details:**

This is the master table for all the books that are available in the Library. This table contains the complete list of books that are available in the library. Each Book id provided with a unique

**Binding:**

This column contains the Binding ID whose details can be fetched from the Binding\_Detailstable. The Binding ID is a unique number given to each type of Binding.

**Category\_Details:**

This includes the Category ID and Category Name. The Category ID serves as a primary key.

Columns:

**Borrower\_Details:**

This table contains the details of all the persons who lent a book from the library. Each Student will be given a Unique borrower ID. All the library-related activity for a particular person will be captured based on the Borrower ID. This table will be used to track the borrowing records. The borrower ID will serve as a primary key here.

**staff\_Details:**

This table contains the details of the staff in the Library. Each Staff member will be given a unique User ID which serves as a Primary Key.

**Student\_Details:**

This table contains the details of all the students who are eligible for availing of Library facilities. Each student will be provided with a unique Student ID and Borrower ID. The student ID will be Primary Key, whereas Borrower\_ID and Phone\_no will be Unique.

**Shelf\_Details:**

This table contains the position of the book…That means which floor and shelf the book is situated on.

**Professor details:**

It keeps the records of professors of different departments and campuses.

**Campus:**

It has 6 campus details and their location and id to relate to the library and the connection between them

**Main community meeting:**

Which keep the record of community meetings and books used and fines due and on which book it dues

**Requests:**

Which keeps the request detail of submitted requests on demand of people students and staff

**Detail of checked books:**

It keeps the detail of the last checked books and keeps the record of data of books to due

**Task(4) Database Tables**

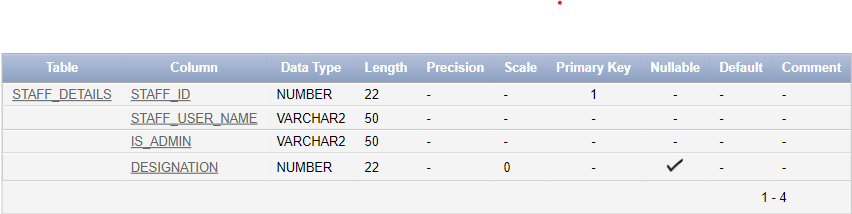
**Table1: STAFF DETAILS**

create Table Staff\_details (staff\_id number PRIMARY KEY,

staff\_user\_name varchar2(50) Not null,

Is\_admin varchar2(50) Not null,

Designation int);

****

**:: their staff\_id is the primary key to connecting with other tables**

**This table also contains staff names, adminship, and designation of the staff members' columns.**

**Table2: OVER DUE DETAILS**

create Table over\_due\_details (limit\_time\_students timestamp,

over\_duedate date primary key,

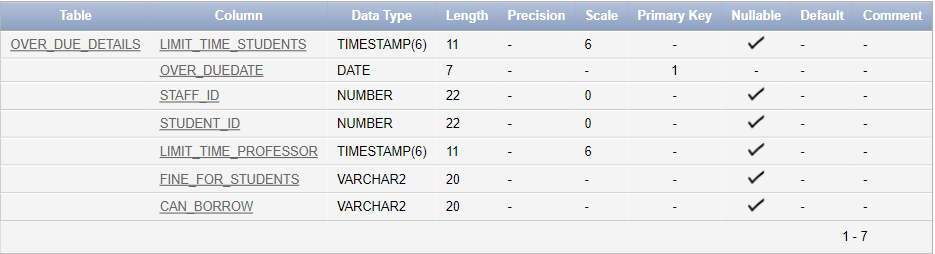
staff\_id int,

student\_id int,

limit\_time\_professor timestamp,

fine\_for\_students varchar(20),

can\_borrow varchar2(20));

****

**:: their over\_duedate is the primary key to connecting with other tables**

**This table also contains limit time of students, staff id, student id, limit time for professors, fine for students if they do not return or re-issue the books, and can borrow or not columns**

**Table3: BOOK DETAILS**

create Table book\_details (book\_id int primary key,

book\_title varchar2(50),

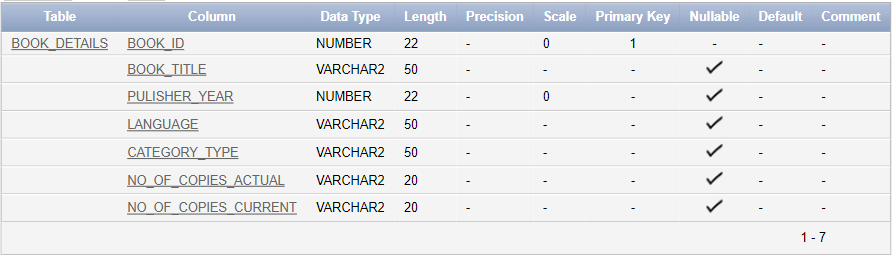
pulisher\_year int,

language varchar(50),

category\_type varchar(50),

no\_of\_copies\_actual varchar(20),

no\_of\_copies\_current varchar2(20));



**:: their book\_id is the primary key to connecting with other tables**

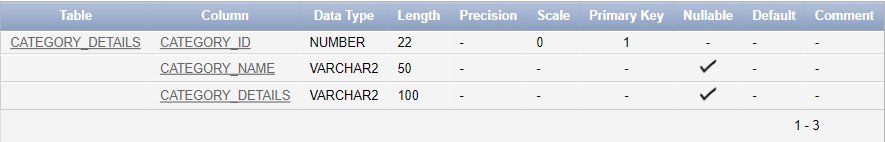
**This table also contain book title, publisher year, language,** **category\_type , no\_of\_copies\_actual , no\_of\_copies\_current** **columns.**

**Table4: CATEGORY DETAILS**

create Table category\_details (category\_id int primary key,

category\_name varchar2(50),

category\_details varchar2(100));



**:: their category\_id is the primary key to connecting with other tables**

**This table also contains category name and category details** **columns.**

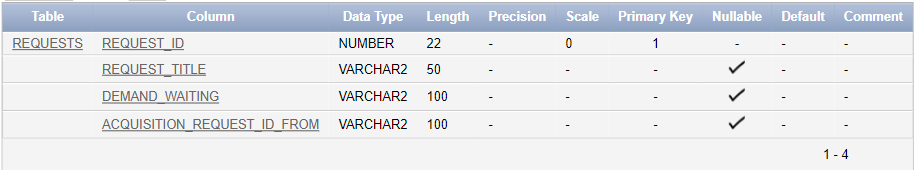
**Table5: REQUESTS**

create Table Requests (Request\_id int primary key,

Request\_title varchar2(50),

Demand\_waiting varchar2(100),

acquisition\_request\_id\_from varchar2(100));



**:: their request\_id is the primary key to connecting with other tables**

**This table also contains request title demand waiting and acquisition request-id columns.**

**Table6: SHELF DETAILS**

create Table Shelf\_details (shelf\_id int primary key,

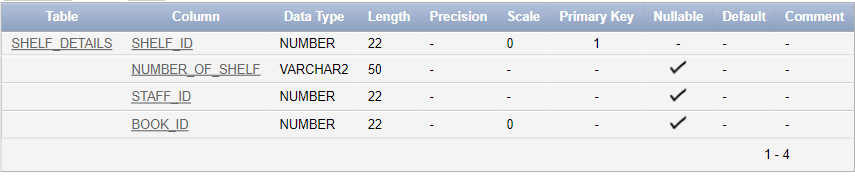
number\_of\_shelf varchar2(50),

Staff\_id number,

book\_id int,

foreign key(staff\_id) references staff\_details(staff\_id),

foreign key(book\_id) references book\_details(book\_id));



**:: their shelf\_id is the primary key to connecting with other tables**

**This table also contains the number of shelf book id and staff id columns**

**The foreign key is used on book\_id to connect this table with the book\_details table and the Foreign key is used on staff\_id to connect this table with the staff\_details table**

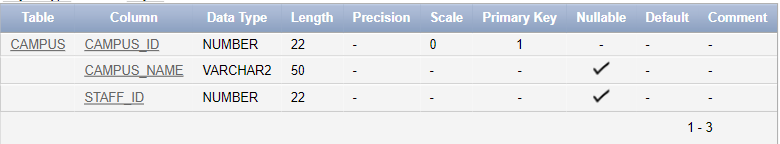
**Table7: CAMPUS**

create Table campus (campus\_id int primary key,

campus\_name varchar2(50),

Staff\_id number,

foreign key(staff\_id) references staff\_details(staff\_id));



**:: their campus\_id is the primary key to connecting with other tables**

**This table also contains the number of campus name and staff id columns**

**The foreign key is used on staff\_id to connect this table with the staff\_details table.**

**Table8: MAIN COMMUNITY MEETING**

create Table main\_comunity\_meeting (campus\_id int,

often\_checked\_title varchar2(50),

type\_borrower varchar2(50),

top\_ten\_titles varchar2(50),

last\_com\_meeting varchar2(50),

last\_highest\_demand\_title varchar2(50),

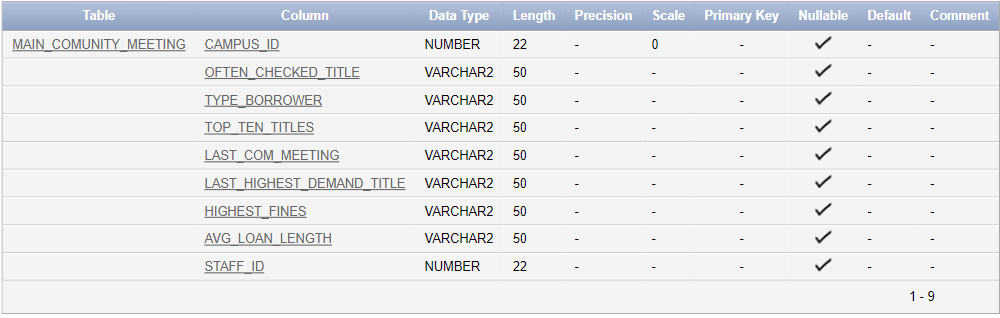
highest\_fines varchar2(50),

avg\_loan\_length varchar2(50),

Staff\_id number,

foreign key(staff\_id) references staff\_details(staff\_id),

foreign key(campus\_id) references campus(campus\_id));



**:: This table contains campus\_id, often\_checked\_title, type\_borrower, top\_ten\_titles, last\_com\_meeting, last\_highest\_demand\_title, highest\_fines, avg\_loan\_length, Staff\_id** **columns**

**The foreign key is used on campus\_id to connect this table with the campus table and the Foreign key is used on staff\_id to connect this table with the staff\_details table.**

**Table9: BORROW DETAILS**

create Table borrow\_details (borrow\_id int primary key,

actual\_return\_date date,

issued\_by varchar2(50),

borrowed\_time\_student date,

book\_id int,

category\_id int,

Request\_id int,

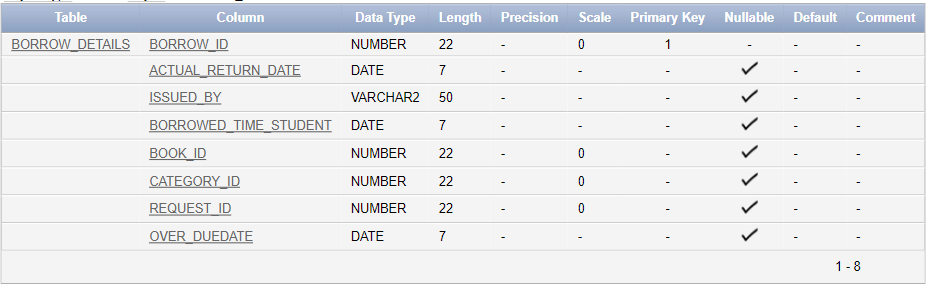
over\_duedate date,

foreign key(book\_id) references book\_details(book\_id),

foreign key(category\_id) references (category\_id),

foreign key(request\_id) references requests(request\_id),

foreign key(over\_duedate) references over\_due\_details(over\_duedate));

****

**:: their borrow\_id is the primary key to connecting with other tables**

**This table also contains actual\_return\_date, issued\_by, borrowed\_time\_student, book\_id, category\_id, Request\_id, over\_duedate columns**

**The foreign key is used on book\_id to connect this table with the book\_details table,**

**The foreign key is used on category\_id to connect this table with the category\_details table,**

**The foreign key is used on request\_id to connect this table with the requests table and**

**The foreign key is used on over\_duedate to connect this table with the over\_due\_details table.**

**Table10: STUDENT DETAILS**

create Table student\_details (student\_id int primary key,

student\_name varchar2(50),

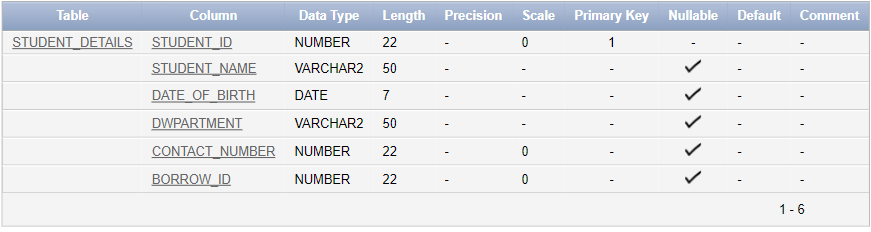
date\_of\_birth date,

department varchar2(50),

contact\_number int,

borrow\_id int,

foreign key(borrow\_id) references borrow\_details(borrow\_id));



**:: their student\_id is the primary key to connecting with other tables**

**This table also contains student name, date of birth, department, contact number, and borrow id columns**

**The foreign key is used on borrow\_id to connect this table with the borrow\_details table.**

**Table11: PROFESSOR DETAILS**

create Table professor\_details (professor\_id int primary key,

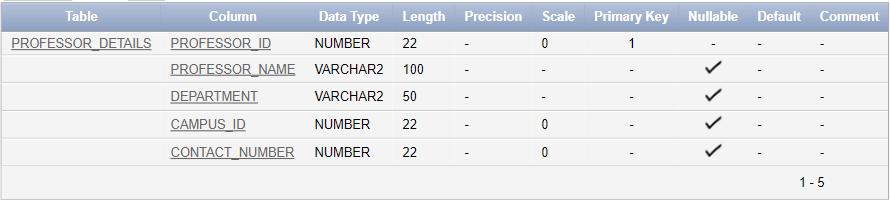
professor\_name varchar2(100),

department varchar2(50),

campus\_id int,

contact\_number int,

foreign key(campus\_id) references campus(campus\_id));



**:: their professor\_id is the primary key to connecting with other tables**

**This table also contains professor name, department, campus id contact number columns**

**The foreign key is used on campus\_id to connect this table with the campus table.**

**Table12: DETAILS OF CHECKED BOOKS**

create Table detail\_of\_checked\_books (student\_id int,

professor\_id int,

book\_title varchar2(100),

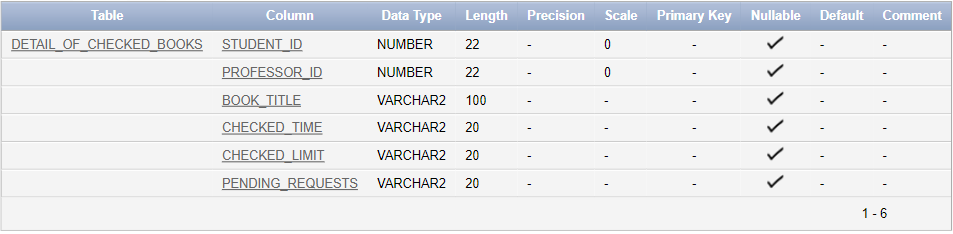
checked\_time varchar(20),

checked\_limit varchar(20),

pending\_requests varchar(20),

foreign key(student\_id) references student\_details(student\_id),

foreign key(professor\_id) references professor\_details(professor\_id));

****

**:: This table contains student id, professor id, book title, checked time, checked limit, and pending requests columns**

**The foreign key is used on student\_id to connect this table with the student\_details table**

**And Foreign key is used on professor\_id to connect this table with the professor\_details table.**

# References

Software 1 ; moon modeler by Datensen .

Software 2 ; 11g by oracle