UNIVERSITY OF GONDAR

COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCE
DEPARTMENT OF INFORMATION TECHNOLOGY AND COMPUTER SCIENCE(MSC)

Project title: - Distributed Database System for University of Gondar Library

By: - Alehegn Adane and

Senait Tefera

Submitted to: Ayalew Belay(PHD)

Jan, 2016

Gondar, Ethiopia

Acknowledgment

First and foremost to God who makes everything possible. We would like to thank sincerely to all University of Gondar library staffs for their necessary information. We would especially like to express our deep gratitude to our instructor Ayalew Belay (PHD) from Addis Abeba University for his instruction based project which is incorporated with new concepts. We would also like to extend our thanks to our class mates for their constructive comment and support.

Table of Content

List of Figures

CHAPTER ONE

1.1 Introduction

In today's world of emerging technologies, enterprises are moving towards the Internet for businesses. People are rushing towards the e-commerce applications for their day-today needs, which in turn are making the internet very popular and every activity is moving from manual to computer. The library played an important role in the daily teaching, scientific research and learning among teachers and students, and to promote its information was the powerful guarantee of school education, teaching and scientific research. The management of the library books using computer could reduce manual management mistakes and enhance the efficiency library service greatly. Therefore, using computer to manage library collections and member details has very significance. Databases and database technology are having a major impact on the growing use of computers. A database is a collection of related data. A database management system (DBMS) is a collection of programs that enables users to create and maintain database. Today, Information processing is distributed over several computers rather than confined to a single machine. Distributed system is therefore very important for enterprise computing systems. A distributed system consists of a collection of autonomous computers linked by a computer net work and equipped with distributed system software. Replication is having multiple copies of data and services in a distributed system. Reasons of replication are: Reliability of the system, Better protection against corrupted data, Improved Performance and faster response time, Facilitates scaling in numbers and geographical area. The basic motivations for selecting UOG library system, when we visit the library system, librarians suffer by the following main problems 1) due to the existing system is commercial software, it is challenging to extend to new branch, 2) since the system is desktop application, librarians couldn't able to check the status of the member in other branch library (clearance process is difficult) 3) frequently losing data 4) four branches libraries are still manual. The new system will minimize the above problems and it will improve performance, increased availability, share ability, expandability and access facility because distributed system has an opportunity to those problem. The basic activities in library distributed system will includes; registering books, updating books ,deleting books , borrow books, renew books, and reserve books being any branch or multiple location.

1.2 Problem Statement

Currently, UOG Library (only two branches) uses Electronic Library Management (ELM) system. Atse tewdrows library and Maraki library uses ELM software. In those libraries students who comes to the library for the first time is registered via the system by the data encoder. All books are also registered in the system. After book and member registration processed, borrowing and returning books has been made using the system for those two branches only. The problem is currently, since the system is commercial, it couldn't be extended to the new branch library as well it is challenging to modify the interface even the name of library. The rest of the library branches are working manually. As a result of single system (two branch), students are expected to clear their status in branch library whether they borrow or not. It is impossible to check status of students and books in all branches being any branch because of undistributed system. Managing those records is highly monotonous process and there are different challenges as well. Like: - difficult to check status of every student being very where, check status of books being everywhere, workload in one branch because of once branch book should enter in that branch only, inability to assure reliability of the system(frequently data is lost). In addition to the above, the existing system is highly tolerated by the following main problems:-

- Transparent problems:-because of single system, it is impossible to assure transparency of system. Branch libraries use their own database server so that branch library resource put only in that server.
- Fault tolerance problem: - difficult to provide fault tolerance because of lack of backup server. In case of failure of one server, totally all data will be losing.
- Share ability problem: it does not allow systems to use each other's resources.
- Expandability problem: it does not permit new systems to be added as members of the overall system. Example:-in UOG there are 2 new branch libraries are established before 3 years, but the system couldn't be functional in that new branch library.

1.3 Objective of the project

1.3.1 General Objective

The general objective of this project is to design distributed database system for University of Gondar library.

1.3.2 Specific Objectives

In order to achieve the general objective, the following specific objectives are specifying:

- ✓ To analysis UOG library system,
- ✓ To design user interface that can register members and books,
- ✓ To design database that can store member and book information,
- ✓ To design interface that can generate report,
- ✓ To design distributed database system,

1.4 Scope of the project

The scope of the project covers design of Distributed database for University Gondar library. The project includes the following activities;

- ✓ Analysis UOG libraries,
- ✓ Identify book information,
- ✓ Identify member (student, staff and guest) information,
- ✓ Design the database conceptually,
- ✓ Identify the relationship between the actors and use cases
- ✓ Design the database physically including the hardware implementation
- ✓ Implementation of distributed system

CHAPTER TWO

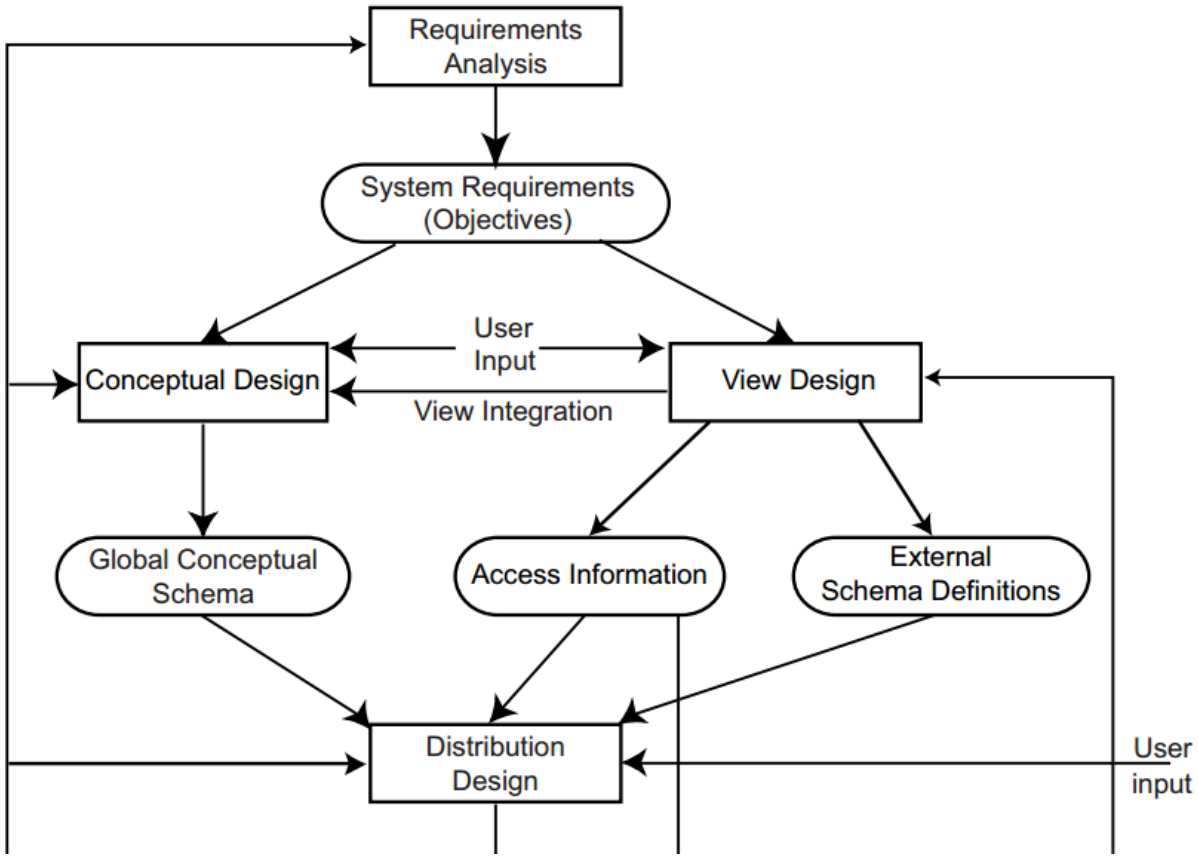
2. Methodology of the project

The process of analyzing the organization and its environment, developing database model and implementing the model requires an appropriate methodology. Traditional system analysis provides a possible approach, but a staged database design approach offers a better solution. Two major strategies that have been identified for designing distributed databases are the top-down approach and the bottom-up approach [1]. As the names indicate, they constitute very different approaches to the design process. Top down approach is more suitable for tightly integrated, homogeneous distributed DBMSs, while bottom-up design is more suited to multidatabases. Since we have used homogeneous database, this document followed top-down database design approach.

2.1 Top-Down database design process

A framework for top-down design process is shown in Figure1. The activity begins with a requirements analysis that defines the environment of the system and "elicits both the data and processing needs of all potential database users" [2]. The requirements study also specifies where the final system is expected to stand with respect to the objectives of a distributed DBMS as identified in need assessment. The requirements document is input to two parallel activities: view design and conceptual design. The view design activity deals with defining the interfaces for end users. The conceptual design, on the other hand, is the process by which the enterprise is examined to determine entity types and relationships among these entities. One can possibly divide this process into two related activity groups entity analysis and functional analysis. Entity analysis is concerned with determining the entities, their attributes, and the relationships among them. Functional analysis, on the other hand, is concerned with determining the fundamental functions with which the modeled enterprise is involved. The results of these two steps need to be cross-referenced to get a better understanding of which functions deal with which entities. There is a relationship between the conceptual design and the view design. In one sense, the conceptual design can be interpreted as being an integration of user views. Even though this view integration activity is very important, the conceptual model should support not only the existing applications, but also future applications. View integration should be used to ensure that entity and relationship requirements for all the views are covered in the conceptual schema. In conceptual design and view design activities the user needs to specify the data entities and must

determine the applications that will run on the database as well as statistical information about these applications.



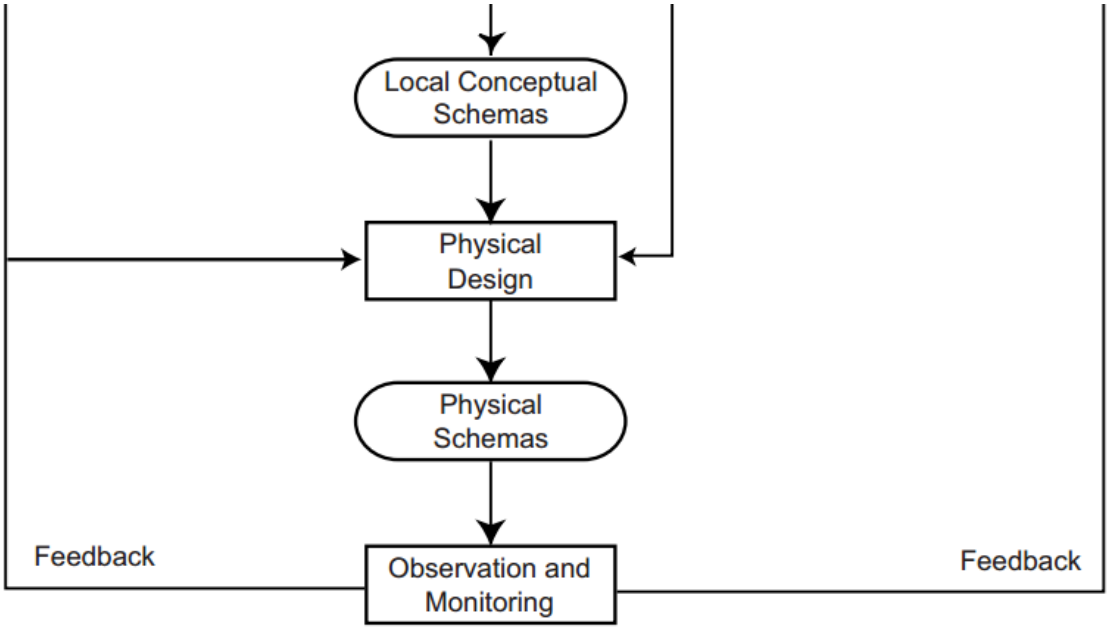


Figure 1: Top-Down Design Process [3]

The global conceptual schema (GCS) and access pattern information collected as a result of view design are inputs to the distribution design step. The objective at this stage, which is the focus of this chapter, is to design the local conceptual schemas (LCSs) by distributing the entities over the sites of the distributed system. It is possible, of course, to treat each entity as a unit of distribution. Rather than distributing relations, it is quite common to divide them into sub relations, called fragments, which are then distributed. Thus, the distribution design activity consists of two steps: fragmentation and allocation. The reason for separating the distribution design into two steps is to better deal with the complexity of the problem. The last step in the design process is the physical design, which maps the local conceptual schemas to the physical storage devices available at the corresponding sites. The inputs to this process are the local conceptual schema and the access pattern information about the fragments in them. It is well known that design and development activity of any kind is an ongoing process requiring constant monitoring and periodic adjustment and tuning. We have therefore included observation and monitoring as a major activity in this process. Note that one does not monitor only the behavior of the database implementation but also the suitability of user views.

2.2 Need assessment

The project team conducted need assessment on selected area. The assessment data were collected from key informants of the student (student library club members), instructors and staffs through unstructured interview and focused group discussion. As the respondents explain, starting from 2003, UOG use Electronic Library Management System (ELM) which is commercial and desktop application software. As its inception, it was implemented in main library (College of Medicine and Health Science Library) only. After some time, the system is implemented in two branch library. As the library staffs describe, the current system has the following problems. firstly, because of it is commercial software, it is not possible to modify or is not possible to expand to new branch library(only two branch library work).secondly, some of branch library are working manually. Thirdly, the system has no any replica, because of lack of replica and inability to take back up properly; more than 500,000 data are lost in CMHS Library last year. Because of this the reliability and the availability of the system are very low and poor. As a result, the project teams understand there is a need of developing distributed database system which able to curb the listed problems.

2.3 Proposed System Overview

The proposed system can be expanded easily to all branch libraries so that the day to day task (specially, circulation process) can be automated. Since the new system is distributed, the librarian can view the status of the user at any branch so that clearance process can be managed easily. In addition to the above, the system can manage all transaction like; book issue, return, reserve and etc. Because of the system has replica, the availability and the reliability of the system is high. The system provides the capability of generating a report of all transactions made, and all users available.

2.4 Functional and Non Functional Requirements

2.4.1. Functional Requirements

Functional requirements are the main things that the user expects from the system. It describes what this project should do and it describes the behavior of the system as it relates to the system's functionality. Hence, the functional requirements of this project are the following;

- ✓ allow the admin to add Staff
- ✓ allow the admin to update Staff Profile
- ✓ allow the librarian to clear users
- ✓ allow admin/librarian to login to the system
- ✓ allow the admin to delete Staff
- ✓ allow the librarian to issue book to the members
- ✓ Enable the librarian to return issued book
- ✓ Enable the librarian to view transaction report
- ✓ Enable the admin and the librarian to display transaction report
- ✓ allow the librarian to add book detail
- ✓ Enable the librarian to register member of the library
- ✓ Enable the librarian to issue the Book to members
- ✓ Enable the librarian to check availability of the book in the library

2.4.2 Non Functional Requirements

Non-functional requirements are not straight forward requirement of the system rather it is related to usability (in some way), for example system availability is one non functional requirement of this project. Hence, this system should be available 24/7 hour service with no down if possible. The following are some of these project non functional requirements;

- ✓ Scalability
- ✓ Capacity
- ✓ Availability
- ✓ Reliability
- ✓ Recoverability
- ✓ Maintainability
- ✓ Security
- ✓ Quality

2.5 System and Functional Model (conceptual model)

A system model is the conceptual model that describes and represents a system. In this section the system is described by showing its functionality of static and dynamic behavior of the system. Use cases, sequence, class and activity diagram has been used to show the system in detail.

A functional model in software engineering is a structured representation of the functions (activities, actions, processes, operations) within the modeled system or subject area.

2.5.1 Project actors and use cases

Actors	Use cases
✓ Administrator ✓ Librarian ✓ Member(student, staff and guest)	✓ Login ✓ Issue/Return Book ✓ Search Book ✓ Display Report ✓ Check Availability ✓ Add/update/remove Book ✓ Add/Remove Member ✓ Add/ Remove Staff

Table 1: actors and use cases

2.4.2 System use cases

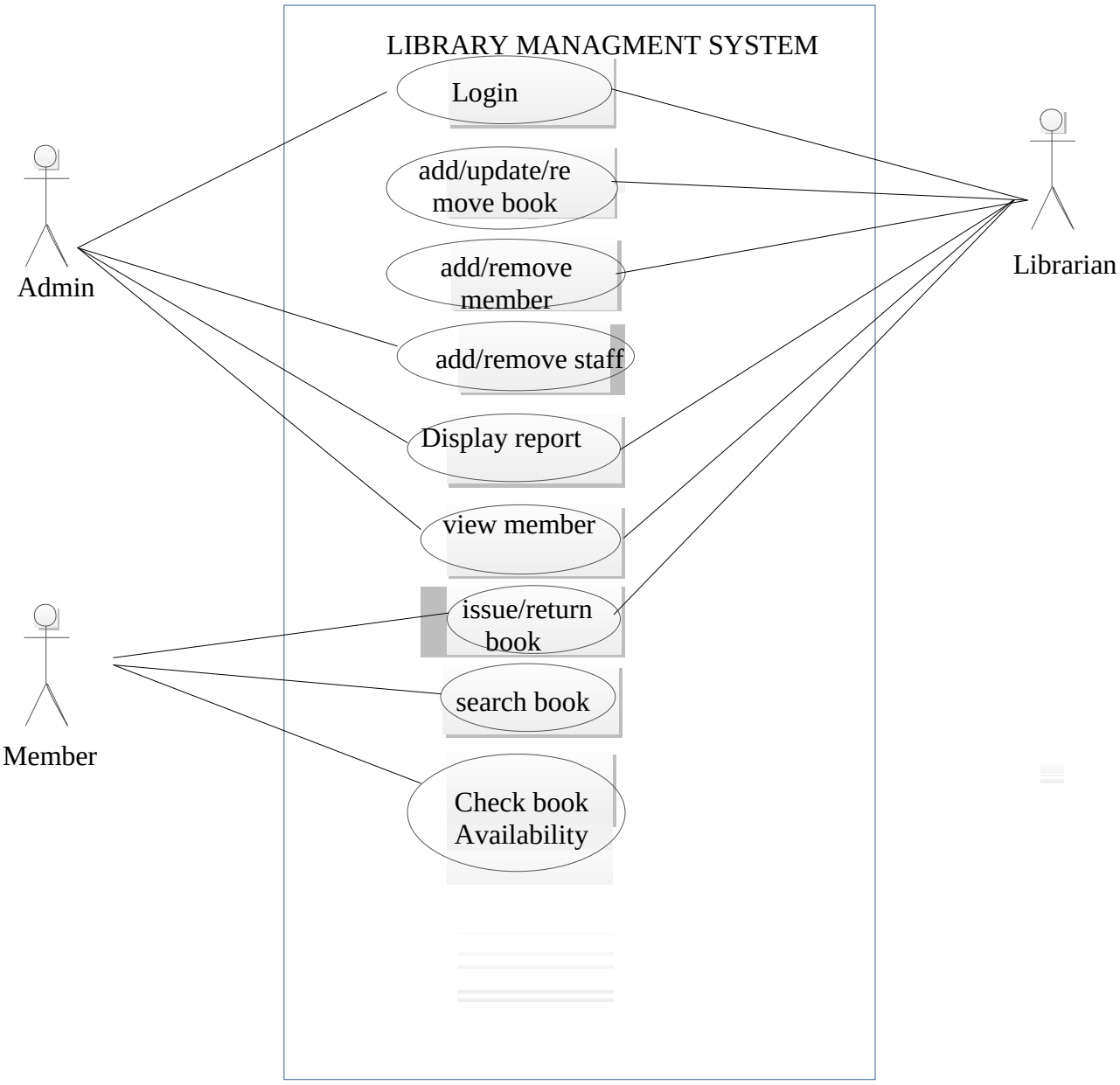


Figure 2: use case diagram

2.5.3. Description of use cases

Use case name	**Login**
Actors	Administrator/Librarian
Aim	To secure or authenticate the admin or librarian
Description	The librarian or admin enter the required information and the system will authenticate
Preconditions	Run the project
Post conditions	the system displays either of admin page or librarian page based on the selected user type
Alternative flow	Username or Password is incorrect, or miss mach in user type user will be prompted a message.
Use case name	**Check availability**
Aim	To check the issue and return details.
Actors	Librarian/member
Description	Enter the ISBN or unique identifiers
Preconditions	None
Post conditions	Information about Issue/Return should be displayed
Alternative flow	None
Use case name	**Search book**
Aim	To search book
Actors	Librarian/Member
Description	Enter search key click search button
Preconditions	None.
Post conditions	Book information should be displayed
Alternative flow	None

Use case name	**add/remove member**
Purpose	To add and remove member detail.
Actors	Librarian
Description	For Add case, enter member information, Add the member For Edit case, enter member ID, view member detail, update entry, & update profile For Remove case, enter member ID, remove member
Preconditions	Librarian should be logged into the system.
Post conditions	Member information should be displayed for view case and confirmation message should be prompted upon successful add, edit and remove.
Alternative flow	For Add case, if Member ID already in the database then prompt Duplicate Member ID message. For Edit and View, if Member ID is not valid prompt invalid Member ID message. For Remove case, if member ID is not valid then prompt invalid member ID message, else prompt book borrowed by this member is not returned message
Use case name	**add/update/remove book**
Purpose	To add, edit, remove book
Actors	Librarian
Description	To add book, enter book information, Add the book To update, enter ISBN, view book detail, update entry, & update profile To remove book, enter ISBN, and remove book
Preconditions	Librarian should be logged into the system.
Post conditions	Book information should be displayed for view case and confirmation message should be prompted upon successful add, edit and remove.
Alternative flow	Add, if ISBN already in the database then prompts Duplicate Message. For Edit and View, if ISBN is not valid then prompt Invalid ISBN message. For Remove case, if ISBN is not valid prompt invalid ISBN message, else prompt Issued message

Use case name	**Issue/Return Book**
Aim	To issue book for member or return book from member
Actors	Librarian
Description	For Issue case, enter ISBN, check book availability, enter member ID, and issue the book.
Preconditions	Librarian should be logged into the system. Books should be available to Issue in issue case, and books should be already issued in Return case.
Post conditions	Confirmation message should be prompted upon successful Issue/Return. Database should be updated.
Alternative flow	For issue case, if the ISBN not valid or book is not available or member ID is not available or member try to borrow more than three books, prompt error messages accordingly. For Return case if the book is not already issued, prompt book is returned message, and if ISBN – member ID combination is invalid prompt wrong combination message.
Use case name	**Add/Remove Staff**
Purpose	To add and remove staff detail
Actors	Administrator
Overview	For Add case, enter staff information, For Remove case, enter staff ID, remove staff
Preconditions	Administrator should be logged into the system.
Post conditions	Staff information should be displayed for view case and confirmation message should be prompted upon successful add and remove.
Alternative flow	For Add case, if staff ID already in the database then prompt Duplicate staff ID message. For Remove case, if staff ID is not valid then prompt invalid staff
Use case name	**Display report**
Aim	To view transaction report
Actors	Administrator/Librarian
Description	Click view report, display result
Preconditions	Administrator/Librarian should be logged in

Post conditions	Transaction Report should be displayed
Alternative flow	None

Table 1: use case description

2.5.3 Dynamic Model

The dynamic aspect of a system can be further described using the following techniques.

2.5.3.1 Activity Diagram

Activity diagram represent the flow form one activity to another activity in our system and captures the dynamic behavior of this system. It focuses on operation.

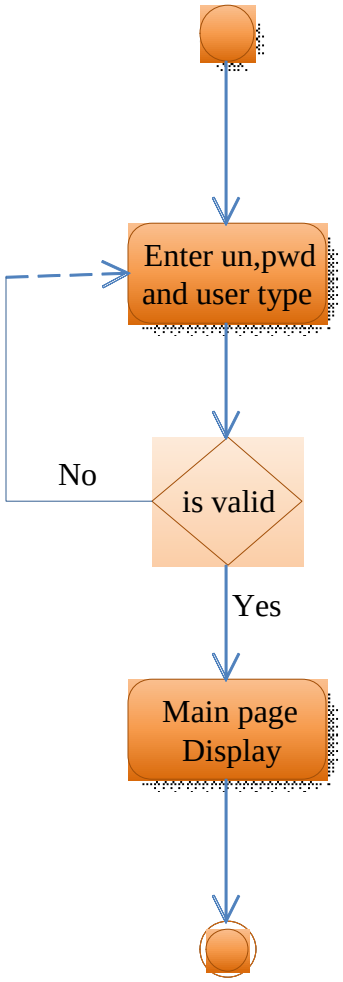


Figure 3: activity diagram for login

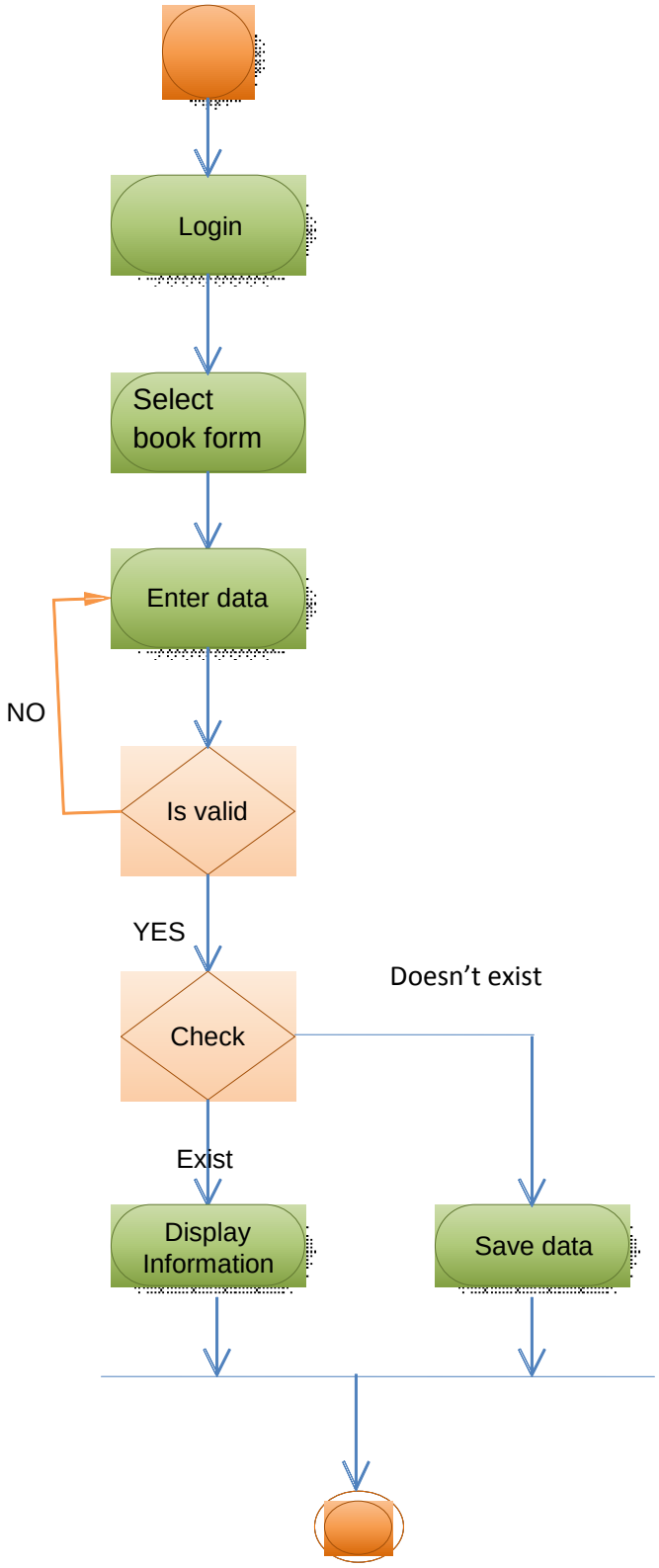


Figure 4: activity diagram for book registration

2.5.2.2. Sequence Diagram

This diagram shows which objects communicate with which other objects; and what messages trigger those communications.

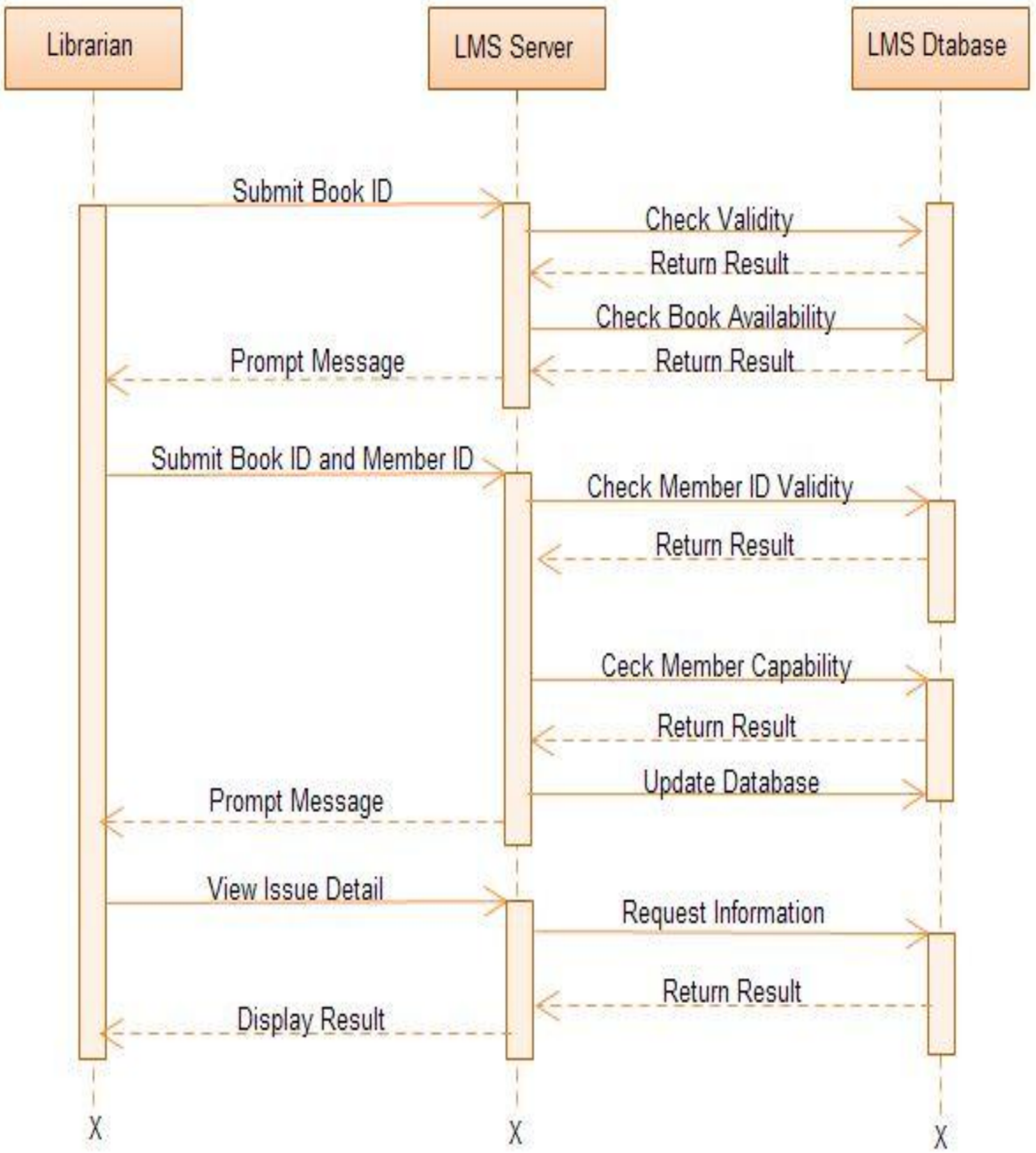


Figure 5:sequence diagram for book issue

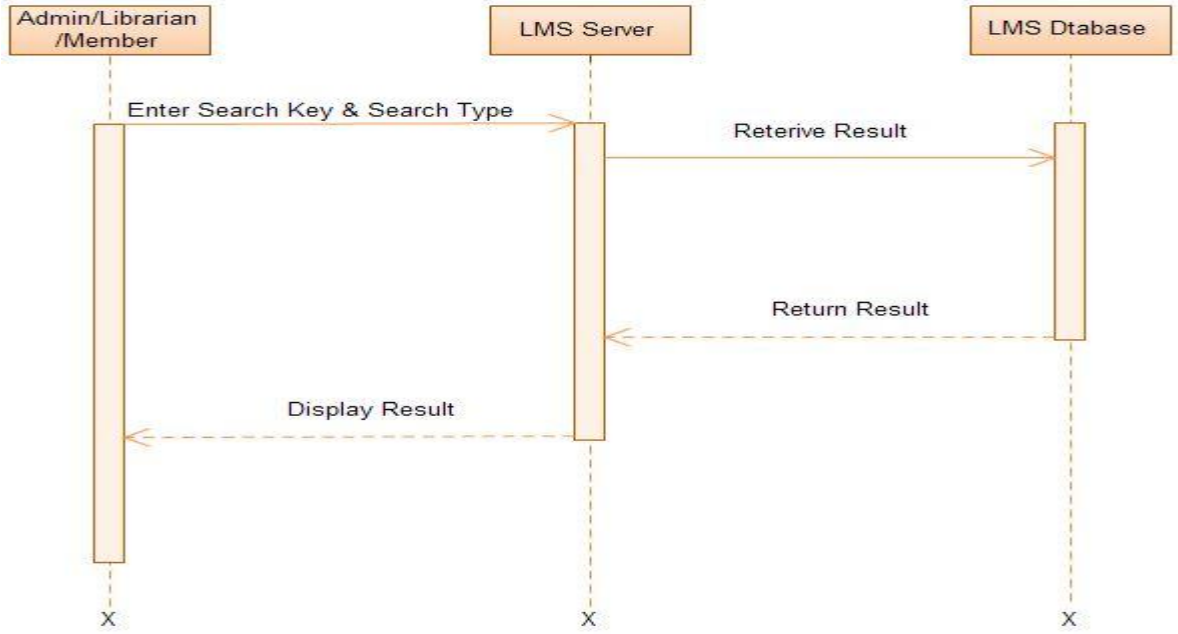


Figure 6: Sequence Diagram For Search Book

2.5.2.3 Class Diagram

Class Diagram shows a set of classes, interfaces, and collaborations and their relationships. They are important for visualizing, specifying, and documenting structural models and also for constructing executable systems through forward and reverse engineering.

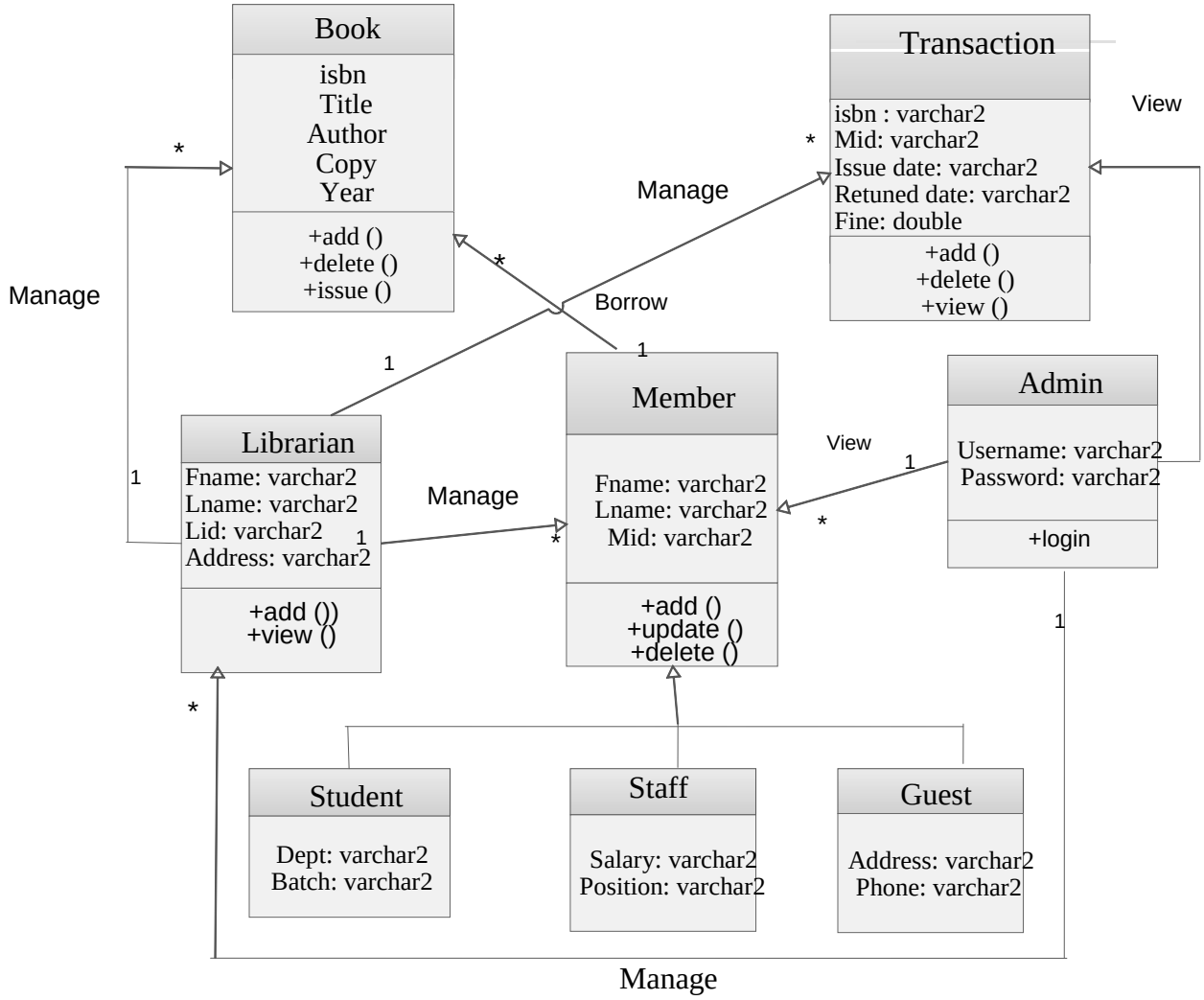


Figure 7: class diagram

2.7 Database description using pseudo code

This section describes database description using pseudo code for some selected project components which are login, member registration, book registration and issue book.

i. Pseudo code for login

Start

Enter user name and password

Check the validity

Main page display

End

ii. Pseudo code for member registration(for student)

Start
Click member menu
Select student button
Enter member detail
Click add button
End

iii. Pseudo code for book registration

Start

Enter user name and password

Select book registration page

Enter book detail

If the book detail exists, display the detail

Else add book detail

Click saves

End

iv. Pseudo code for Book issue

Start

enter ISBN and member ID

enter issue and due date

click issue

end

2.8 Database Tools

Netbeans 8.0.2 for project inteface , oracle 11g for data storage and ojdbc6 for java and oracle connection have been used.

2.9 Deployment diagram (physical data model)

Deployment diagrams are a set of nodes and their relationships. It used for visualizing deployment view of a system.

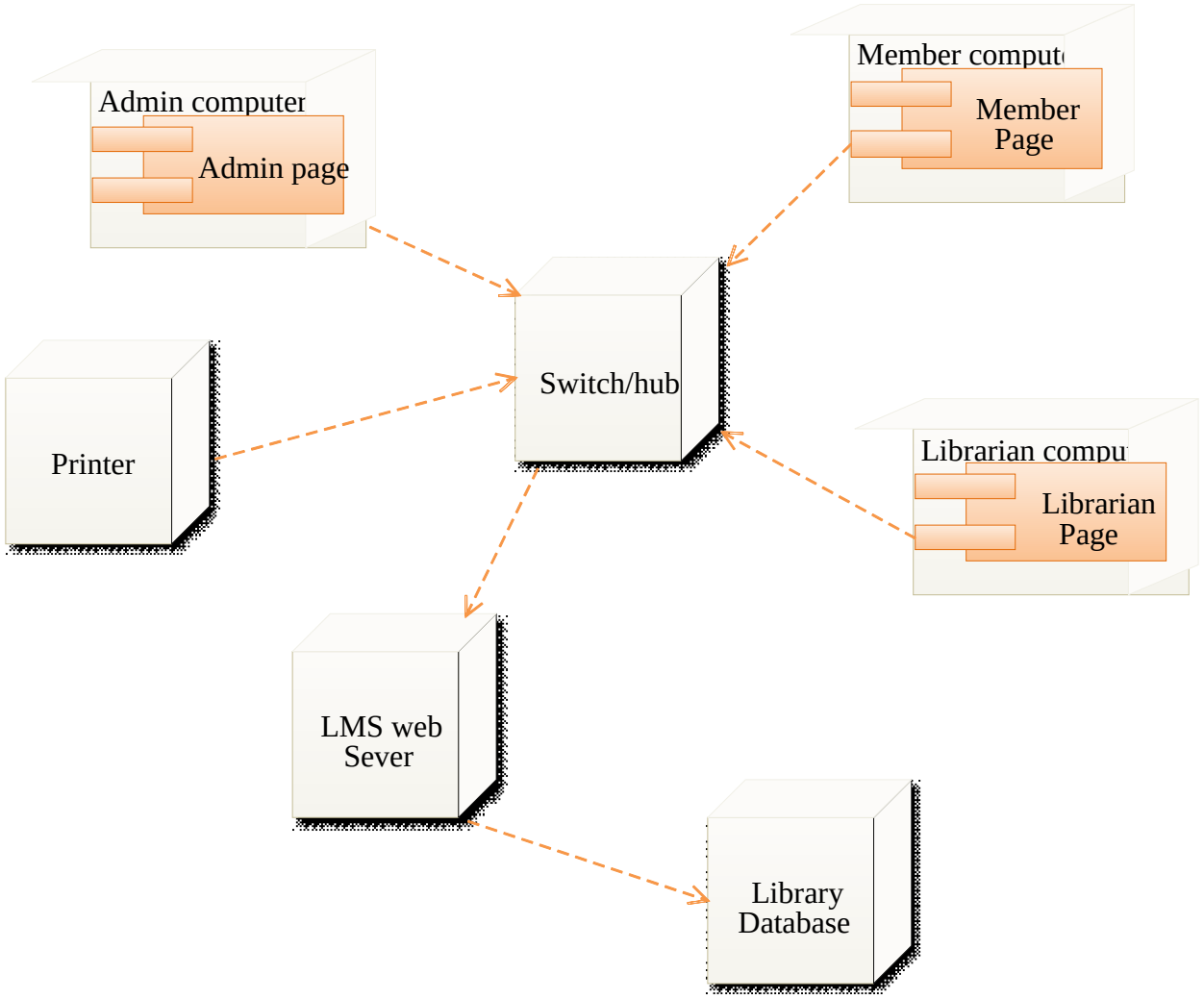


Figure 8: deployment diagram

2.10 System Implementation

This section describes the project implementation specially the object oriented concepts like; type and inheritance implementation. The followings are the project object types and some of inheritances. As we describe the project in the UML diagram (use case and class diagram), the main actor or class our project are member, librarian, admin, book and transaction in the library.

2.10.1 Inheritance implementation

The project member can be student, staff or guest. Those member types can share some attributes and behavior.

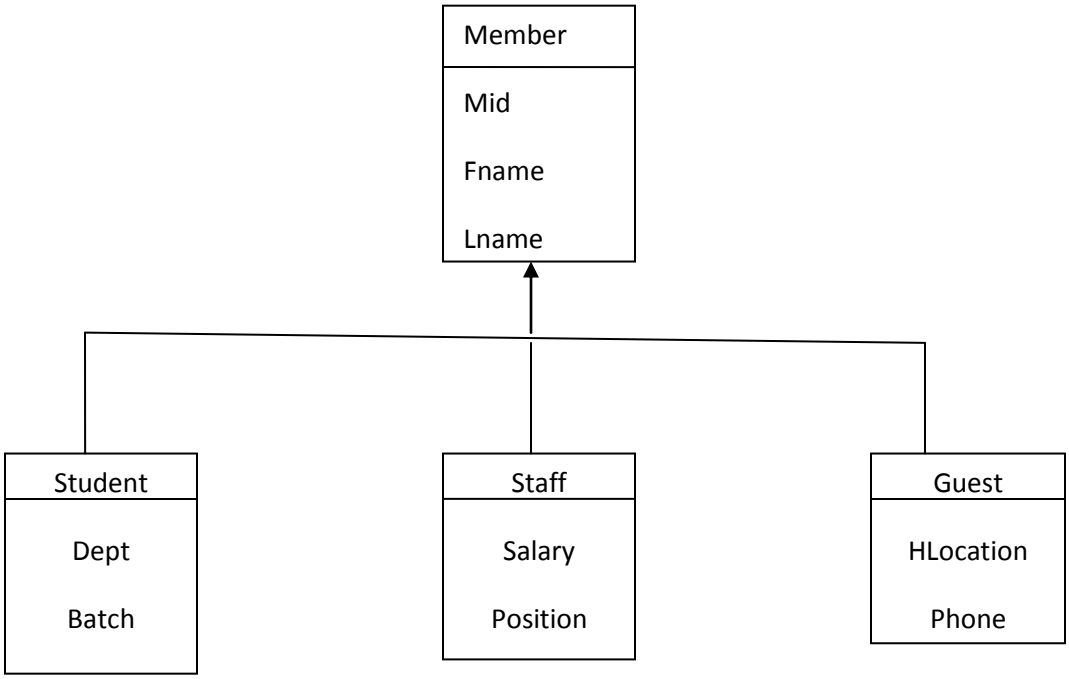


Figure: 9 inheritance implementation physically (super type and subtype)

All member types have common unique identifier, first name and last name, so they share them as super type. But there are some attributes which could not be share, example student subtype has department and batch for its own only, staff has salary and position, guest has home location and phone for its own. So, our project implements how these real world behaviors appear in the database. The following oracle sql statements shows how three users inherit some attribute from super type member.

Member type (super type)

```
CREATE or REPLACE TYPE member_T AS OBJECT

(Fname varchar2(20),

Lname varchar2(20),

mid varchar2(20)) NOT FINAL /
```

Type guest (sub type)

```
CREATE or REPLACE TYPE guest UNDER member_T
(Phone varchar2 (20),
Location varchar2 (20))
```

Type staff (sub type)

```
CREATE or REPLACE TYPE staff UNDER member_T
(salary varchar2(20),
position varchar2(20))
```

type student(sub type)

```
CREATE or REPLACE TYPE student_T UNDER member_T
(dept varchar2(20),
batch varchar2(20))
```

2.10.2 Grant of privileges implementation

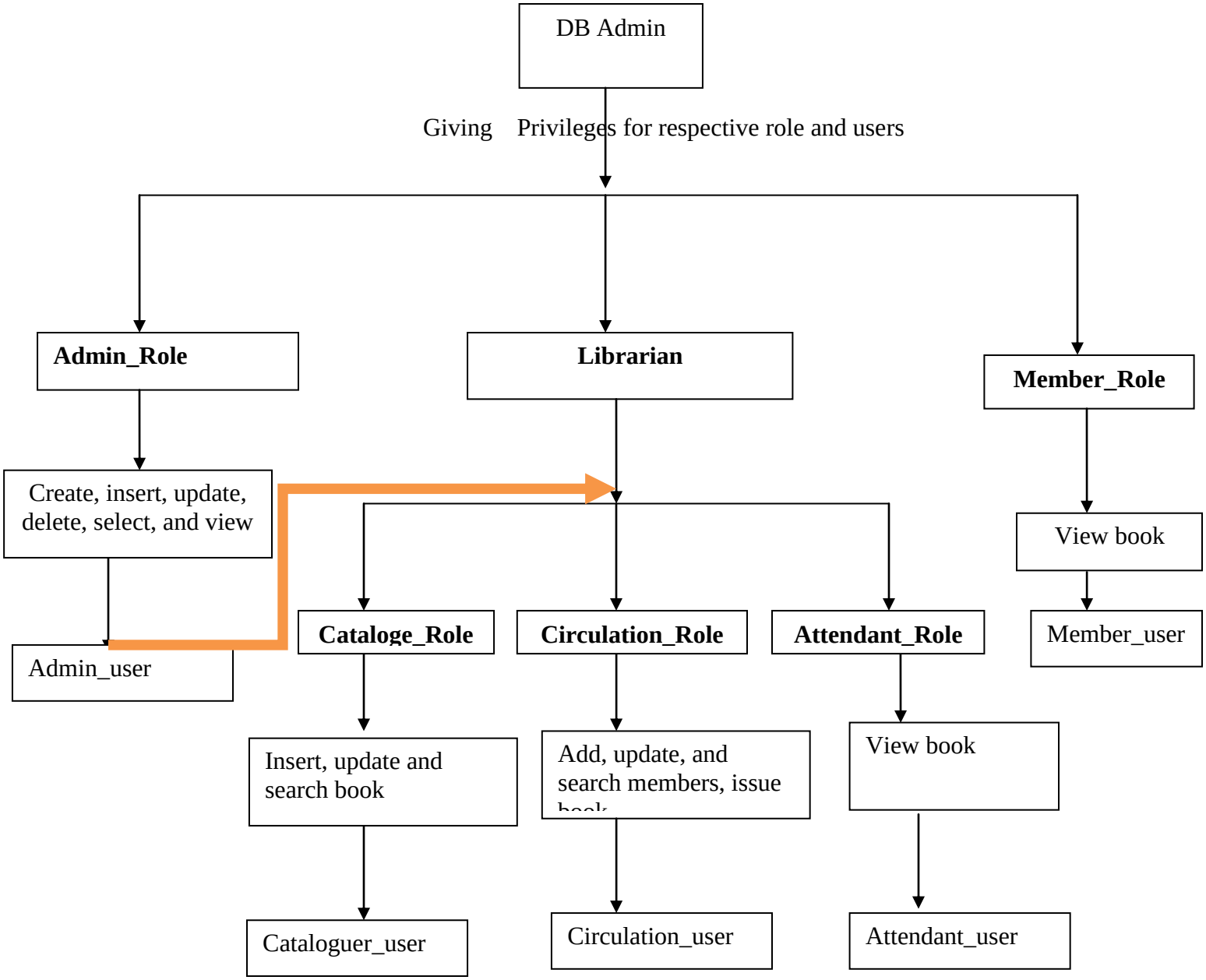


Figure 10: Grant of privileges implementation

In Oracle™ we can grant system, role, or object privileges to three different types mentioned below.

User: - The privilege is given to particular users, and the user can then exercise the privilege.

Role: - The privilege is given to particular roles, and the user who has been granted the role will be able to exercise the privilege.

Public: - The privilege is given to all users. [4]

A grant on a system privilege is the grant to carry out a basic system operation such as create table, create procedure, and so forth. A grant on a role is the grant to access the information of the particular role. Finally, a grant on an object privilege is the grant to do a particular action to a particular object. Thus, for a grant on an object privilege, we need to declare the schema of the grant object target. In this project, we identify the available roles and users so that granting is given for roles and users. To make simplify, we use the following procedures;

Step 1: identify the available roles in the library

1. Admin_role
2. Cataloguer_role
3. Circulation_role
4. attendant_role
5. member_role

Step 2: identify the users that we have

1. Admin
2. Cataloguer
3. Circulation worker
4. Attendant
5. Member

Step 3: Create role, connect the role and give specific task for role

```
->create role admin_role;//create role
Grant connect,resource to admin_role;
Grant  insert,select,update,delete on lib_t to admin_role;//giving tasks to the admin_role
->create role cataloguer_role;
Grant connect,resource to cataloguer_role;
```

Grant insert, select, update on book_t cataloguer_role;

->create role circulation_role;

Grant connect, resource to Circulation_role;

Grant insert,select, update on book_t circulation_role;

Grant select on issue to circulation_role;

->create role attendant_role;

Grant connect,resource to attendant_role;

Grant select on book_t to attendant_role;

>create role member_role;

Grant connect,resource to member_role;

Grant select on book_t to member_role;

Step 4: create user and give the assigned roles for individual users so that they can operate.

->create user admin identified by dir;

Grant admin_role to admin;

Grant circulation_role to admin with admin option;

Grant attendant_role to admin with admin option;

Grant cataloguer_role to admin with admin option;

Grant member_role to admin with admin option;

Grant create user to admin;

Grant create session to admin;

->create user circulation identified by circ;

->create user cataloguer identified by cat;

->create user attendant identified by attendant;

>create user member identified by member;

Step 5: admin can give the roles privileges to created user by his own user name and password.

Grant circulation_role to circulation;

Grant attendant_role to attendant;

Grant cataloguer_role to cataloguer;

Grant member_role to member;

If you went revoke role=>revoke circulation_role from circulation;

2.10.3 View implementation

Views are known as logical tables. They represent the data of one of more tables. Different users will have different view based on their need or functionality. Example; most of the time, the library administrator wants to know number of books available in the library, but the librarians wants to know each attribute or details of a book, on the other hand the library members may only need title, author and number of copy of the book. So, their view is different.

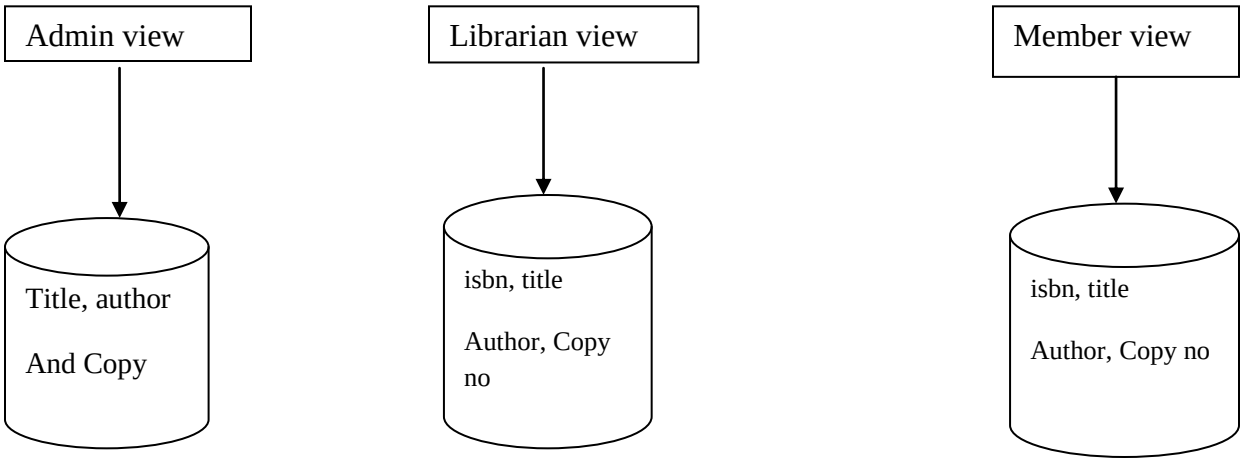
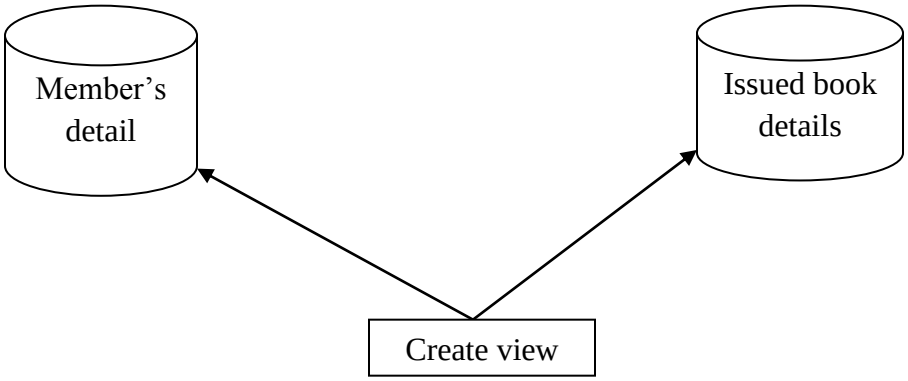


Figure 11: view implementation physically

As join operation Views play fundamental roles to access required data from multiple tables. Example:-if we are asked to display borrower's detail (name and id) with their borrowed book details (book isbn, title and author). The required data is found from two tables.



Create view issuebook select as mid,fname,lname,isbn,auhor,title from student,book_T where book_T.isbn=issue.isbn;

2.10.4 Distributed system implementation

We use homogenous distributed database system. A distributed database system consists of loosely coupled sites that share no physical component.

- ✓ Database systems that run on each site are independent of each other
- ✓ Appears to user as a single system
- ✓ All sites have identical software
- ✓ It appears to user as a single system

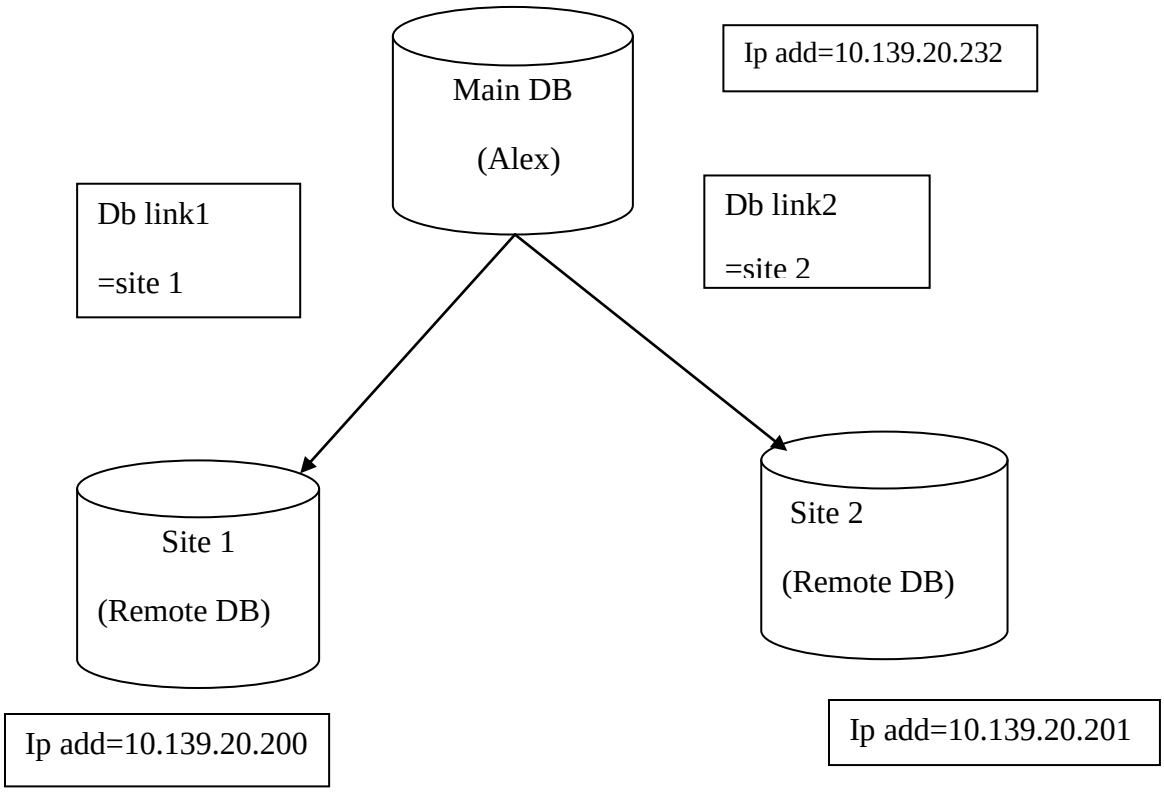


Figure 10: project distributed system model

As the above figure shows, distributed system connects three databases which are main DB(Alex) ,site1 and site 2

An application can simultaneously access or modify the data in several databases in a single distributed environment. We follow the following procedures to create distributed database;

step 1: create database models as you like(main db,site1 and site2)
stept 2:check the connection between main database and the clients or remote database using ip address(you can use wirless/wired connection or peer to peer cable) ,in our case we use peer to peer cable
step 3:configure listener.org file on remote database or you can configure the listener using graphical interface(go to all program,oracle,then net manager,net service name,here you can add the followings: net service name,ip address,port number(remote database)
the test it
step 4:configure service name on the main database or you can use graphical interface as we do step 3
step 5:create user and table on remote database
step 6:create db link on server for number of remote database
step 7:finally query your remote database using db link

3. Conclusion

The library played an important role in the daily teaching, scientific research and learning among teachers and students. The management of the library books using computer could reduce manual management mistakes and enhance the efficiency library service greatly. Databases and database technology are having a major impact on the growing use of computers I the library. A database is a collection of related data. A database management system (DBMS) is a collection of programs that enables users to create and maintain database. Today, Information processing is distributed over several computers rather than confined to a single machine. A distributed system consists of a collection ofautonomous computers linked by a computer network and equi pped with distributed system software. The library has the following main problems 1) due to the existing system is commercial software, it challenging to extend to new branch, 2) report generating problem 3) four branches libraries are still manual as result tasks are time and effort consuming. The main aim of this project is to design Distributed database system for University of Gondar Library System. The users of the system are staffs, students, guests and librarians. We have followed object oriented Top-Down database design process as a methodology. Static and dynamic diagrams have been used to understand the system in detail. We describe the project activities using pseudo code. Inheritance has been implemented in the project. We have one main database and two remote databases which are connected to the main database. Available users have been using their own view or task as the real environment reflects.

4. Reference

[1]. Ceri, S. and Pernici, B. (1985). Dataid–d: Methodology for distributed databasedesign. In Albano, V. d. A. and di Leva, A., editors, Computer-Aided Database Design, pages 157–183. North-Holland. 121

[2]. Yao, S. B., Waddle, V., and Housel, B. (1982b). View modeling and integration using the functional data model. IEEE Trans. Softw. Eng., SE-8(6):544–554. 149

[3]. M. Tamer Ozsu and Patrick Valduriex: principles of distributed database system (third edition): University of Waterloo: Canada: springer, 2011

[4].wenny rahayu,David Taniar and Eric Pardede: Object oriented oracle, La Trobe University, Australia: Hershey , London, Melbourne, Singapore 2006.

[5]. Diana Lorentz(2005), Oracle Database SQL Reference, 10gRelease 2 (10.2)

6. Appendix

```
1.----create type memebr
create or replace type member_T as object
(Fname varchar2(20),
Lname varchar2(20),
mid varchar2(20)) not final
/
--create table member
create table member of member_T
(mid not null,
primary key(mid));

2. create or replace type student_T under member_T
(dept varchar2(20),
batch varchar2(20))
/
--create table under student_T

create table studen of student_T
(mid not null,
primary key(mid));

3. create staff type
create or replace type staff under member_T
(salary varchar2(20),
position varchar2(20))
/
---create table for staff
create table staff_Table of staff
(mid not null,
primary key(mid));
```

4. ----create type guest under member_T

create or replace type guest under member_T

(phone varchar2(20),

location varchar2(20))

/

.....create table for guest

create table guest_T of guest

(mid not null,

primary key(mid));

5.type book

create or replace type book as object

(isbn varchar2(20),

title varchar2(20),

author varchar2(20),

copy varchar2(20),

year varchar2(20))

/

---table book

create table book_T of book

(isbn not null,

primary key(isbn));

6.create librarian table as table not object table

create table lib_t

(libid varchar2(10),

fname varchar2(10),

lname varchar2(10),

phone varchar2(10),

address varchar2(10));

synonym

```
create table lib
(libid varchar2(10),
fname varchar2(10),
lname varchar2(10),
phone varchar2(10),
address varchar2(10));
```

7.create table for issue

```
create table issue
(isbn varchar2(20),
mid varchar2(20),
issuedate varchar2(20),
duedate varchar2(20));
```

----steps to create distributed database

step 1:create connection between two pc
step 2:configre net service (sid,ip address,port,listner) for remote pc(listner.org,tnsname)
step 3:check the connectoin between two db
step4 :create database link at the server
step 5:test the link
you can also configure using GUI or SQL plus

----retrive data from local pc

```
select * from member;
select * from studen;
select * from staff_Table;
select * from guest_T;
select * from book_T;
select * from lib_t;
```

select * from issue;

////////

CONNECTING TO REMOTE SITES - CREATING DATABASE LINKS TO ACCESS DATA ON REMOTE - SITE 1
CREATE PUBLIC DATABASE LINK site1
CONNECT TO system
IDENTIFIED BY "122"
USING '10.139.20.230/lib';

--CONNECTING TO REMOTE SITES - CREATING DATABASE LINKS TO ACCESS DATA ON REMOTE -SITE 2
create public database link site2
connect to system
identified by TEKU
using '10.139.20.164/uog';

create table to remote database
create table lib_t@site1 (name varchar2(10));
ERROR at line 1:
ORA-02021: DDL operations are not allowed on a remote database
use this command to fix the errors happen
exec dbms_utility.exec_ddl_statement@fasil('create table lib_t (name varchar2(10))');
DESC lib_t;
exec dbms_utility.exec_ddl_statement@site2('create table f (name varchar2(10))');
DESC lib_t ;
exec dbms_utility.exec_ddl_statement@remotedb('create table min(name varchar2(10))');
DESC lib_t ;
exec dbms_utility.exec_ddl_statement@library('create table f (name varchar2(10))');

```
DESC lib_t ;

INSERT,UPDATE,VIEW,SELECT,.....THEN COMMIT;
INSERT INTO lib_t@site1 VALUES('alex');
COMMIT;
UPDATE F*site1 name='alxeo' WHERE name='alex';
COMMIT;


--ACCESSING DATA FROM LOCAL SITE USING DB LINKS
--ACCESSING DATA FROM LOCAL SITE
SELECT * FROM studen;

--ACCESSING DATA FROM REMOTE SITES  Site 1
SELECT *FROM lib_t@site1;

--ACCESSING DATA FROM REMOTE SITES - Site 2
SELECT *FROM issue@site2;

--DISPLAY data FROM ALL SITES(union)
SELECT * FROM    studen UNION SELECT *FROM lib_t@site1 UNION SELECT
*FROM issue@site2

--CREATING SYSNOYMS
CREATE OR REPLACE SYNONYM sissue FOR issue@site2;
CREATE OR REPLACE SYNONYM issue FOR issue@remotedb;

--CREATING SYSNOYMS TO ACCESS REMOTE DATA - SITE 1
CREATE OR REPLACE SYNONYM slib_t FOR lib_t@site1;
SELECT *FROM slib_t;
```

SELECT fname, lname,isbn,mid from studen,
FROM emp e, dept d
WHERE e.deptno = d.deptno;

--ACCESSING FROM REMOTE SITES USING SYNONYM - SITE 1 AND SITE 2
SELECT *FROM sissue UNION SELECT * FROM slib_t;
///
SELECT *FROM issue1@remotedb UNION SELECT * FROM issue;(correct)
select * from issue union select * from issue1@remotedb
///
union and union all
select fname from studen union select fname from staff_table
select fname from studen union all select fname from staff_table
select fname from studen intersect select fname from staff_table(common row)
select fname from studen minus select fname from staff_table(only unique data from table one)
union: select only distnict rows,but union all selects all avaliable rows
intersect:retrive data which is avaliable from two table
minus:retrive only the first data from first table not the second table
///

--ACCESSING FROM LOCAL AND REMOTE SITES USING SYNONYM - LOCAL SITE + SITE 1 + SITE 2
SELECT *FROM studen UNION SELECT *FROM sissue UNION SELECT * FROM slib_t;

---how to insert data to remote site using interface
use the following code before and after the connection:

1. con.setAutoCommit(false);(before the connection)
2. con.commit();(after the operation

---how to insert data to remote site and local site at the same time using java interface

///

-----how to create view

. for book table(book_t)

•

.CREATE VIEW vbook_t AS SELECT * from book_t WHERE copy<=20;

for studen

.CREATE VIEW vbatchstuden AS SELECT * from studen WHERE batch>=2nd ;

.CREATE VIEW vbatchstuden AS SELECT * from studen WHERE batch>=2nd ;

. for staff table

.CREATE VIEW vsalarystaff_table AS SELECT * from staff_table WHERE salary>=6000 ;

.for issue

Create view vvissue as select fname,lname,mid,isbn,title,auhor from studen,book_T;

Create view issuebook select as mid,fname,lname,isbn,auhor,title from student,book_T
where book_T.isbn=issue.isbn;

view for number of record

SELECT COUNT(*) FROM book_t union SELECT COUNT(*) FROM book_t;

///

Knowing existing database

select name from v$database;