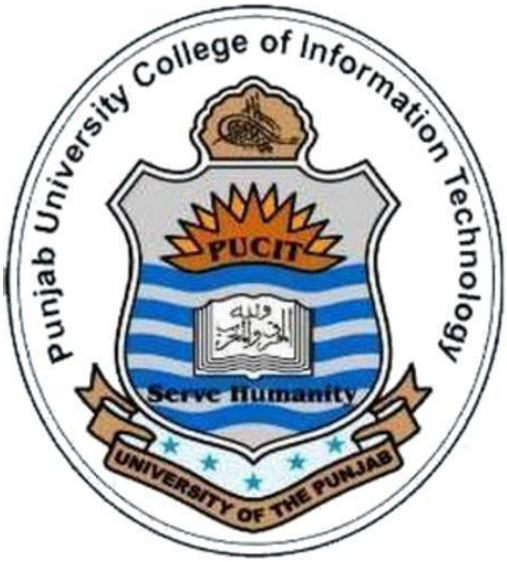
# DATABASE-PROJECT




**Topic:**

## Library Management System

**Submitted to:**

## Sir Asif Sohail

**Submitted by:**


## BCSF21M018 HUSNA SARWAR

## BCSF21M044 AYESHA SARWAR

| Sr. | Topics |
|---|---|
| 1. | Introduction |
| 2. | Relations |
| 3. | Connectivity Table |
| 4. | Transformation into Relation |
| 5. | Relationships and Connectivity |
| 6. | Normalized forms |
| 7. | Description of Relations |
| 8. | Table creation using SQL |
| 9. | Dummy Data Entry |
| 10. | Queries |
| 11. | Triggers |
| 12. | Procedures |
| 13. | Functions |

# Introduction:

This library management system is a comprehensive software solution designed to streamline and enhance the administration and operations of a library. It handles a variety of entities and their relationships, ensuring efficient management of books, members, staff, vendors, and transactions within the library.

**Members:** Each member is uniquely identified by a Mem_Id. The system captures detailed personal information such as name, area, city, house number (H_NO), and street number (ST_No). Members can have multiple phone numbers, managed through the Phone_number table, linking Mem_Id to multiple phone numbers.

**City_area:** This table maps cities to their respective areas, ensuring that members' locations are properly categorized and managed.

**Students:** Identified by Stu_ID, students' major fields of study, grades, and their associated Mem_Id are recorded.

**Teachers:** Managed through the Teacher table, where each teacher has a unique T_ID and is associated with a department and a member ID (Mem_Id).

**Staff:** Similarly, staff members are identified by Staff_ID, with details about their position, department, and associated Mem_Id.

**Teacher_subject:** This table maps teachers (T_ID) to the subjects they teach.

**Staff_position:** This table details the positions available within each department.

**Vendors:** Managed through the Vendor table, each vendor is identified by a V_id and has associated details like name and email. Vendors can have multiple phone numbers recorded in the Vendor_phone table.

**Vendor supplies books:** This relationship table records which vendors supply which books (by ISBN), along with the quantity, unit price, and delivery date.

**Books:** Detailed in multiple tables, books have unique ISBNs and attributes such as title, genre, publisher, and author.

**Copies:** Each physical copy of a book is identified by a Copy_id and its location, condition, and availability are managed through the Copy_location table.

**Book_1 and Book_2:** These tables store additional attributes of books, including their title, genre, publisher, and associated staff and vendor IDs.

## Journal, Newspaper, and Magazine Management

**Journals:** Managed in Journal_1 and Journal_2 tables, each journal is identified by a DOI and linked to an ISBN and volume/issue numbers.

**Newspapers:** Managed in Newspaper_1 and Newspaper_2 tables, each newspaper has a unique identifier (Newspaper_id) and attributes such as date and headlines.

**Magazines:** Managed in Magazine_1 and Magazine_2 tables, magazines are identified by Mag_id and attributes like cover story, date, and topic.

## Transaction and Record Management

**Borrow/Return:** This relationship table tracks book borrowing and returning activities, recording dates, member IDs, and ISBNs.

**Is issued by:** This table records the issuance of books by staff members, capturing the issue date, ISBN, and staff ID.

**Record:** This table logs records with unique Record_IDs, capturing dates and staff IDs involved in any library actions.

**Record_of_fine_action:** It details fines and actions taken on specific dates.

## Author and Publisher Management

**Authors:** Managed through the Author and Author_detail tables, each author has a unique A_ID and associated details like name, nationality, and biography.

**Written 1 and Written 2:** These tables manage the relationship between authors and books, detailing the role and contribution of each author.

**Publishers:** Managed through the Publisher and Publisher_multivalued tables, each publisher has a unique Publisher_ID, along with details about the CEO, contact information, and headquarters.

## Publishing Relationship Management

**Is published by:** This table records the publication details of books, linking the publish date, ISBN, and publisher ID.

This system provides a robust framework for library management, ensuring that all aspects of library operations are efficiently tracked and managed, from member details and roles to book transactions and vendor supplies.

# Relations:

## Member:

| Mem_Id | Name | Area | City | H_NO | ST_No |
|---|---|---|---|---|---|

## Phone_Numbers(Multi-valued_attribute):

| Mem_id | Phone |
|---|---|

## Student:

| Stu_ID | Major | Grade | Mem_id |
|---|---|---|---|

## Teacher:

| T_ID | Department | Subject | Mem_id |
|---|---|---|---|

## Staff:

| Staff_ID | Position | Department | Mem_id |
|---|---|---|---|

## Vendor:

| V_id | V_name | Phone# | email |
|---|---|---|---|

## Vendor supplies books (relationship):

| V_id | ISBN | quantity | Unit_price | Delivery_date |
|---|---|---|---|---|

## Copies:

| Copy_id | location | condition | availability |
|---|---|---|---|

## Books have copies:

| ISBN | Copy_id | Acquisition_date |
|---|---|---|

## Book:

| ISBN | Title | Genre | Publisher | Author | Mem_id | Staff_ID | V_id |
|---|---|---|---|---|---|---|---|

## Journals:

| Volume | Issue | DOI | ISBN |
|---|---|---|---|

## Newspaper:

| Headlines | Date | Addition | ISBN |
|---|---|---|---|

## Magazines:

| | | | |
|---|---|---|---|
| Issue | Cover_story | date | ISBN |

## Return/Borrow Relationship:

| | | | | |
|---|---|---|---|---|
| Borrow date | Return date | Due date | Mem_id | ISBN |

## Library_staff:

| | | | |
|---|---|---|---|
| Staff_ID | Name | Position | Access_rights |

## Is issued by Relationship:

| | | |
|---|---|---|
| Issue_date | ISBN | Staff_ID |

## Record:

| | | | | |
|---|---|---|---|---|
| Record_ID | Date | Fine | Action | Staff_ID |

## Is Written by Relationship:

| A_ID | ISBN | role | contribution |
|---|---|---|---|
| | | | |

## Author:

| A_ID | Name | Nationality | Biography |
|---|---|---|---|
| | | | |

## Publisher:

| Publisher_ID | Headquarter | CEO | Contact_Info |
|---|---|---|---|
| | | | |

## Is published by Relationship:

| Publish_date | ISBN | Publisher_ID |
|---|---|---|
| | | |

# Connectivity Table:

| Entity | Relationship | Connectivity | Entity |
|---|---|---|---|
| Vendor | Supplies | 1:M | Books |
| Books | have | M:M | Copies |

| | | | |
|---|---|---|---|
| Member | Borrow/Return | M:M | Books |
| Books | Is Issued By | M:1 | Library_Staff |
| Books | Are Managed By | M:1 | Library_Staff |
| Books | Is Published By | M:M | Publisher |
| Books | Is Written By | M:M | Author |

# Transformation into Relations:

**Vendor** ( V_id, V_name, Phone#, email ) ;*Here V_id is PK*

**Library_staff** ( Staff_ID, Name, Position, Access_rights ) *;Here Staff_ID is PK*

**Phone_Numbers** ( Mem_id, Phone ) *Here Phone is PK and Mem_id is FK*

**Copies** ( Copy_id, location, condition, availability ) ;Here Copy_id is PK

**Library_staff** ( Staff_ID, Name, Position, Access_rights ) *;Here Staff_ID is PK*

**Record** ( Record_ID, Date, Fine, Action, Staff_ID ) *;Here Record_ID is PK*

**Author** ( A_ID, Name, Nationality, Biography ) *;Here A_ID is PK*

**Publisher** ( Publisher_ID, Headquarter, CEO, Contact_Info ) *;Here Publisher_ID is PK*

## SUPER-TYPE:

**Books** ( ISBN, Title, Genre, Publisher, Author, Mem_id, Staff_ID, V_id )
*;Here ISBN is PK and Mem_id, Staff_ID, V_id are FK*

# SUB-TYPES:

**Journals** ( Volume, Issue, DOI, ISBN ) *;Here DOI is PK and ISBN is FK*

**Newspaper** ( Headlines, Date, Addition, ISBN ) *;Here Date, Addition are PK and ISBN is FK*

**Magazines** ( Issue, Cover_story, date, ISBN ) *;Here Date is PK and ISBN is FK*

# SUPER-TYPE:

**Member** ( Mem_id, Name, Area, City, H_NO, ST_No ) *;Here Mem_id is PK*

# SUB-TYPES:

**Student** ( Stu_ID, Major, Grade, Mem_id ) *;Here Stud_ID is PK and Mem_id is FK*

**Teacher** ( T_ID, Department, Subject, Mem_id ) *;Here T_ID is PK and Mem_id is FK*

**Staff** ( Staff_ID, Position, Department, Mem_id ) *;Here Staff_ID is PK and Mem_id is FK*

**Is issued by Relationship** ( Issue_date, ISBN, Staff_ID ) *;Here ISBN, Staff_ID are PK*

**Is Written by Relationship** ( A_ID, ISBN, role, contribution ) *;Here A_ID, ISBN are PK*

**Books have copies Relationship** ( ISBN, Copy_id, Acquisition_date ) *;Here ISBN, Copy_id are PK*

**Return/Borrow Relationship** ( Borrow_date, Return_date, Due, date, Mem_id, ISBN ) *;Here Borrow_date is PK and ISBN, Mem_id are FK*

**Vendor supplies books Relationship** ( V_id, ISBN, quantity, Unit_price, Delivery_date ) *;Here V_id, ISBN are PK*

**Is published by Relationship** (Publish_date, ISBN, Publisher_ID )
*;Here Publisher_ID, ISBN are PK*

# Relationships and connectivity:

## Vendor - Books:

**Relationship**: One-to-many

**Connectivity**: One vendor can supply multiple books, but each book is supplied by only one vendor. This means the "Vendor" entity is connected to many “Book” entities.

## Books - Copies:

**Relationship**: Many-to-many

**Connectivity**: Each book can have multiple copies, and each copy can be linked to multiple books. This means both the "Book" and "Copy" entities are connected to many instances of each other.

## Member - Books:

**Relationship**: Many-to-many

**Connectivity:** Each member can borrow/return multiple books, and each book can be borrowed/returned by multiple members. This means the "Member" entity is connected to many "Borrow/Return" entities and vice versa.

## Books - Library_Staff:

**Relationship:** Many-to-one

**Connectivity:** Multiple books can be issued by one library staff member, but each book is issued by only one specific staff member. This means the "Books" entity is connected to one "Library_Staff" entity.

## Books - Library_Staff

**Relationship:** Many-to-one

**Connectivity:** Many books are managed by one library staff member, but each book is managed by only one specific library staff member. This means the "Book" entity is connected to one "Library_Staff" entity.

## Books - Publisher:

**Relationship:** Many-to-many

**Connectivity:** Each book can be published by multiple publishers, and each publisher can publish multiple books. This means the "Book" entity is connected to many "Publisher" entities and vice versa.

## Books - Author:

**Relationship:** Many-to-many

**Connectivity:** Each book can be written by multiple authors, and each author can write multiple books. This means the "Book" entity is connected to many "Author" entities and vice versa.

# Normalized Forms:

## Member:

| Mem_Id | Name | Area | City | H_NO | ST_No |
|---|---|---|---|---|---|
| | | | | | |

## City_area:

| City | Area |
|---|---|
| | |

## Phone_number(Multi-valued attribute):

| Mem_id | Phone |
|---|---|
| | |

## Student:

| Stu_ID | Major | Grade | Mem_id |
|---|---|---|---|
| | | | |

## Teacher:

| T_ID | Department | Mem_id |
|---|---|---|
| | | |

## Staff:

| Staff_ID | Position | Department | Mem_id |
|---|---|---|---|
| | | | |

## Teacher_subject:

| T_ID | subject |
|---|---|
| | |

## Staff_position:

| Department | Position |
|---|---|
| | |

## Vendor:

| V_id | V_name | email |
|---|---|---|
| | | |

## Vendor_phone:

| V_id | Phone# |
|---|---|
| | |

## Vendor supplies books (relationship):

| V_id | ISBN | quantity | nit_price | Delivery_date |
|---|---|---|---|---|
| | | | | |

(vendors can supply bundle of books and as different vendor are supplying books so particular vendor id as well as particular book num both determine the price)

## Copies:

| Copy_id | location |
|---|---|

## Copy_location:

| location | Condition | availability |
|---|---|---|

## Books have copies:

| ISBN | Copy_id | Acquisition_date |
|---|---|---|

## Book_1:

| ISBN | Title | Genre | Mem_id | Staff_ID | V_id |
|---|---|---|---|---|---|

## Book_title_genre:

| Title | Genre |
|---|---|

## Book_2:

| ISBN | Publisher | Author |
|---|---|---|

## Journal_1:

| DOI | Issue | ISBN |
|---|---|---|
| | | |

## Journal_2:

| DOI | volume |
|---|---|
| | |

## Newspaper_1:

| Newpaper_id | date | ISBN |
|---|---|---|
| | | |

## Newspaper_2:

| Newpaper_id | Headlines |
|---|---|
| | |

## Magazine_1:

| Mag_id | Cover_story | date | ISBN |
|---|---|---|---|
| | | | |

## Mag_date_coverstory:

| Date | Cover_story |
|---|---|
| | |

## Magazine_2:

| Mag_id | Topic |
|---|---|

## Return/Borrow Relationship:

| Borrow date | Return_date | Mem_id | ISBN |
|---|---|---|---|

## Library_staff1:

| Staff_ID | Name | Position |
|---|---|---|

## Lib_staff_position:

| Name | Position |
|---|---|

## Library_staff2:

| Staff_ID | Access_rights |
|---|---|

## Is issued by Relationship:

| Issue_date | ISBN | Staff_ID |
|---|---|---|

## Record:

| Record_ID | Date | Staff_ID |
|---|---|---|

## Record_of_fine_action:

| Date | Fine | Action |
|---|---|---|

## Author:

| A_ID | Name |
|---|---|

## Author_detail:

| Name | Nationality | Biography |
|---|---|---|

## Written 1:

| A_ID | ISBN | role |
|---|---|---|

## Written2:

| A_ID | ISBN | contribution |
|---|---|---|

In a collaborative book where authors contribute chapters or sections, each author may have contributed different portions of the content.

## Publisher:

| Publisher_ID | CEO | Contact_Info |
|---|---|---|

## Publisher_multivalued:

| Publisher_ID | Headquarter |
|---|---|

## Is published by Relationship:

| Publish_date | ISBN | Publisher_ID |
|---|---|---|

# Description of Relations:

## Member:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Mem_id | Number | 10 | Primary Key |
| Name | Varchar2 | 20 | Not Null |
| Area | Varchar2 | 20 | Not Null |
| City | Varchar2 | 10 | Not Null |
| H_NO | Number | 5 | H_NO > 0 |
| ST_No | Number | 5 | ST_No > 0 |

## City_Area:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| City | Varchar2 | 10 | Not Null |
| Area | Varchar2 | 20 | Not Null |

## Phone_Number:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Mem_id | Number | 10 | Primary Key, Foreign Key |
| Phone | Number | 11 | Not Null |

## Student:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Stu_ID | Number | 5 | Primary Key |
| Major | Varchar2 | 10 | Not Null |
| Grade | Varchar2 | 1 | Not Null |
| Mem_id | Number | 10 | Foreign Key |

## Teacher:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| T_ID | Number | 5 | Primary Key |
| Department | Number | 2 | Not Null |
| Mem_id | Number | 10 | Foreign Key |

## Teacher_subject:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| T_ID | Number | 5 | Primary Key, Foreign Key |
| Subject | Varchar2 | 10 | Not Null |

## Staff:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Staff_ID | Number | 5 | Primary Key |
| Position | Varchar2 | 15 | Not Null |
| Department | Number | 2 | Foreign Key |
| Mem_id | Number | 10 | Foreign Key |

## Staff_position:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Department | Number | 2 | Primary Key, Foreign Key |
| Position | Varchar2 | 15 | Not Null |

## Vendor:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| V_id | Number | 5 | Primary Key |

| | | | |
|---|---|---|---|
| V_Name | Varchar2 | 20 | Not Null |
| Email | Varchar2 | 15 | Not Null |

## Vendor_phone:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| V_id | Number | 5 | Primary Key |
| Phone# | Number | 11 | Not Null |

## Vendor Supplies Books (Relationship):

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| V_id | Number | 5 | Primary Key |
| ISBN | Number | 5 | Foreign Key |
| quantity | Number | 5 | quantity > 0 |
| Unit_price | Number | 5,2 | Not Null |
| Delivery_date | Date | - | Default sysdate |

## Copies:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Copy_id | Number | 5 | Primary Key |
| Location | Varchar2 | 15 | Foreign Key |

## Copy_location:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Location | Varchar2 | 15 | Primary Key |
| Condition | Varchar2 | 20 | |
| availability | Varchar2 | 5 | Is equal to 'Yes' OR equal to 'No' |

## Books have Copies:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| ISBN | Number | 5 | Primary Key |
| Copy_id | Number | 5 | Primary Key, Foreign Key |
| Acquisition_date | Date | - | Not Null |

## Book_1:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| ISBN | Number | 5 | Primary Key |
| Title | Varchar2 | 15 | Foreign Key |
| Genre | Varchar2 | 15 | Not Null |
| Mem_id | Number | 10 | Foreign Key |
| Staff_ID | Number | 5 | Foreign Key |
| V_id | Number | 5 | Foreign Key |

## Book_title_genre:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Title | Varchar2 | 15 | Primary Key |
| Genre | Varchar2 | 15 | Not Null |

## Book_2:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| ISBN | Number | 5 | Primary Key |
| Publisher | Varchar2 | 20 | Not Null |
| Author | Varchar2 | 20 | Not Null |

## Journal_1:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| DOI | Number | 5 | Primary Key |
| Issue | Date | - | Not Null |
| ISBN | Number | 5 | Foreign Key |

## Journal_2:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| DOI | Number | 5 | Primary Key |
| Volume | Number | 5,2 | Not Null |

## Newspaper_1:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Newspaper_id | Number | 5 | Primary Key |
| date | Date | - | Not Null |
| ISBN | Number | 5 | Foreign Key |

## Newspaper_2:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Newspaper_id | Number | 5 | Primary Key |
| Headlines | Varchar2 | 30 | Not Null |

## Magazine_1:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Mag_id | Number | 5 | Primary Key |
| Cover_Story | Varchar2 | 10 | Not Null |
| date | Date | - | Foreign Key |
| ISBN | Number | 5 | Foreign Key |

## Mag_date_coverStory:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| date | Date | - | Primary Key |
| Cover_Story | Varchar2 | 10 | Not Null |

## Magazine_2:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Mag_id | Number | 5 | Primary Key |
| Topic | Varchar2 | 20 | Not Null |

## Return/Borrow Relationship:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Borrow_date | Date | - | Primary Key |
| Return_date | Date | - | Not Null |
| Mem_id | Number | 10 | Foreign Key |
| ISBN | Number | 5 | Foreign Key |

## Library_Staff1:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Staff_id | Number | 5 | Primary Key |
| Name | Varchar2 | 20 | Foreign Key |
| Position | Varchar2 | 10 | Not Null |

## Lib_staff_position:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Name | Varchar2 | 20 | Primary Key |
| Position | Varchar2 | 10 | Not Null |

## Library_Staff2:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Staff_id | Number | 5 | Primary Key |
| Access_rights | Varchar2 | 5 | Is equal to 'Yes' OR<br>Equal to 'No' |

## Is issued by Relationship:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| ISBN | Number | 5 | Primary Key, Foreign Key |
| Issue_date | Date | - | Not Null |
| Staff_ID | Number | 5 | Primary Key |

## Record:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Record_ID | Number | 5 | Primary Key |
| date | Date | - | Foreign Key |
| Staff_ID | Number | 5 | Foreign Key |

## Record_of_fine_action:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| date | Date | - | Primary Key |

| | | | |
|---|---|---|---|
| Fine | Number | 5 | Greater than 0 |
| Action | Varchar2 | 20 | - |

## Author:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| A_ID | Number | 5 | Primary Key |
| Name | Varchar2 | 20 | Foreign Key |

## Author_detail:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Name | Varchar2 | 20 | Primary Key |
| Nationality | Varchar2 | 15 | Not Null |
| Biography | Varchar2 | 15 | Not Null |

## Written1:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| A_ID | Number | 5 | Primary Key, Foreign Key |
| ISBN | Number | 5 | Primary Key, Foreign Key |
| role | Varchar2 | 10 | - |

## Written2:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| A_ID | Number | 5 | Primary Key, Foreign Key |
| ISBN | Number | 5 | Primary Key, Foreign Key |
| contribution | Varchar2 | 10 | Primary Key |

## Publisher:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Publisher_ID | Number | 5 | Primary Key |
| CEO | Varchar2 | 20 | Not Null |
| Contact_info | Number | 11 | Not Null |

## Publish_multiValued:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Publisher_ID | Number | 5 | Primary Key, Foreign Key |
| Head_quarter | Varchar2 | 20 | Not Null |

## Is_publish_by Relationship:

| Attributes | Data Type | Size | Constraint |
|---|---|---|---|
| Publisher_ID | Number | 5 | Primary Key, Foreign Key |
| ISBN | Number | 5 | Primary Key, Foreign Key |
| Publish_date | Date | - | Default sysdate |

# TABLES CREATION USING SQL:

## Member:

```
CREATE TABLE member (
    Mem_id NUMBER(10) PRIMARY KEY,
    Name VARCHAR2(20) NOT NULL,
    Area VARCHAR2(20) NOT NULL,
    City VARCHAR2(10) NOT NULL,
    H_NO NUMBER(5) CONSTRAINT chk_h_no CHECK (H_NO > 0),
    ST_No NUMBER(5) CONSTRAINT chk_st_no CHECK (ST_No > 0)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| MEMBER | MEM_ID | NUMBER | - | 10 | 0 | 1 | - | - | - |
| | NAME | VARCHAR2 | 20 | - | - | - | - | - | - |
| | AREA | VARCHAR2 | 20 | - | - | - | - | - | - |
| | CITY | VARCHAR2 | 10 | - | - | - | - | - | - |
| | H_NO | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | ST_NO | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | | | | | | | | | 1 - 6 |

# City_Area:

```
CREATE TABLE City_Area (
   City VARCHAR2(10) NOT NULL,
   Area VARCHAR2(20) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| CITY_AREA | CITY | VARCHAR2 | 10 | - | - | - | - | - | - |
| | AREA | VARCHAR2 | 20 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 2 |

# Phone_number:

```
CREATE TABLE  Phone_Number (
    mem_id NUMBER(10) PRIMARY KEY,
Phone NUMBER(11) NOT NULL ,
CONSTRAINT fk_mem_id FOREIGN KEY(mem_id) REFRENCES
Member(mem_id)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| PHONE_NUMBER | MEM_ID | NUMBER | - | 10 | 0 | 1 | - | - | - |
| | PHONE | NUMBER | - | 11 | 0 | - | - | - | - |
| | | | | | | | | | 1 - 2 |

# Student:

```
CREATE TABLE Student(
    Stu_ID NUMBER(5) PRIMARY KEY,
    Major VARCHAR2(10) NOT NULL,
    Grade VARCHAR2(1) NOT NULL,
    Mem_id NUMBER(10),
    CONSTRAINT fk_mem_id_student_new FOREIGN KEY (Mem_id)
REFERENCES member(Mem_id)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| STUDENT_NEW | STU_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | MAJOR | VARCHAR2 | 10 | - | - | - | - | - | - |
| | GRADE | VARCHAR2 | 1 | - | - | - | - | - | - |
| | MEM_ID | NUMBER | - | 10 | 0 | - | ✓ | - | - |
| | | | | | | | | | 1 - 4 |

# Teacher:

```
CREATE TABLE Teacher (
T_ID NUMBER(5) PRIMARY KEY,
Department NUMBER(2) NOT NULL,
Mem_id NUMBER(10),
CONSTRAINT fk_mem_id_teacher FOREIGN KEY (Mem_id)
```

| REFERENCES member(Mem_id) |
|---|
| ); |

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| TEACHER | T_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | DEPARTMENT | NUMBER | - | 2 | 0 | - | - | - | - |
| | MEM_ID | NUMBER | - | 10 | 0 | - | ✓ | - | - |
| | | | | | | | | | 1 - 3 |

# Teacher_subject:

| CREATE TABLE Teacher_subject ( |
|---|
| T_ID NUMBER(5) PRIMARY KEY, |
| Subject VARCHAR2(50) NOT NULL, |
| CONSTRAINT fk_t_id FOREIGN KEY (T_ID) REFERENCES Teacher(T_ID) |
| ); |

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| TEACHER_SUBJECT | T_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | SUBJECT | VARCHAR2 | 10 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 2 |

# Staff:

| CREATE TABLE Staff ( |
|---|
| Staff_ID NUMBER(5) PRIMARY KEY, |
| Position VARCHAR2(30) NOT NULL, |
| Department NUMBER(2), |
| Mem_id NUMBER(10), |

| |
|---|
| CONSTRAINT fk_department FOREIGN KEY (Department) REFERENCES Staff_position(Department), |
| CONSTRAINT fk_mem_id_staff FOREIGN KEY (Mem_id) REFERENCES member(Mem_id) |
| ); |

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| STAFF | STAFF_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | POSITION | VARCHAR2 | 15 | - | - | - | - | - | - |
| | DEPARTMENT | NUMBER | - | 2 | 0 | - | ✓ | - | - |
| | MEM_ID | NUMBER | - | 10 | 0 | - | ✓ | - | - |

1 - 4

# Staff_position:

| |
|---|
| CREATE TABLE Staff_position ( |
| Department NUMBER(2) PRIMARY KEY, |
| Position VARCHAR2(30) NOT NULL |
| ); |

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| STAFF_POSITION | DEPARTMENT | NUMBER | - | 2 | 0 | 1 | - | - | - |
| | POSITION | VARCHAR2 | 15 | - | - | - | - | - | - |

1 - 2

# Book2:

| |
|---|
| CREATE TABLE Book2 ( |
| ISBN NUMBER(5) PRIMARY KEY, |
| Publisher VARCHAR2(20) NOT NULL, |
| Author VARCHAR2(20) NOT NULL ); |

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| BOOK2 | ISBN | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | PUBLISHER | VARCHAR2 | 20 | - | - | - | - | - | - |
| | AUTHOR | VARCHAR2 | 20 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 3 |

# Book_title_genre:

```
CREATE TABLE Book_title_genre (
    Title VARCHAR2(15) PRIMARY KEY,
    Genre VARCHAR2(15) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| BOOK_TITLE_GENRE | TITLE | VARCHAR2 | 15 | - | - | 1 | - | - | - |
| | GENRE | VARCHAR2 | 15 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 2 |

# Vendor_phone:

```
CREATE TABLE Vendor_phone (
    V_id NUMBER(5),
    Phone# NUMBER(11),
    CONSTRAINT pk_vendor_phone PRIMARY KEY (V_id, Phone#)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| VENDOR_PHONE | V_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | PHONE# | NUMBER | - | 11 | 0 | 2 | - | - | - |
| | | | | | | | | | 1 - 2 |

# Vendor:

```
CREATE TABLE Vendor (
    V_id NUMBER(5) PRIMARY KEY,
    V_Name VARCHAR2(20) NOT NULL,
    Email VARCHAR2(15) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| VENDOR_BOOK | V_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | V_NAME | VARCHAR2 | 20 | - | - | - | - | - | - |
| | EMAIL | VARCHAR2 | 15 | - | - | - | - | - | - |

1 - 3

# Vendor_Supplies_Books:

```
CREATE TABLE Vendor_Supplies_Books (
    V_id NUMBER(5),
    ISBN NUMBER(5),
    Quantity NUMBER(5) CHECK (Quantity > 0),
    Unit_price NUMBER(5,2) NOT NULL,
    Delivery_date DATE DEFAULT SYSDATE,
    FOREIGN KEY (V_id) REFERENCES Vendor(V_id),
    FOREIGN KEY (ISBN) REFERENCES Book2(ISBN)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| VENDOR_SUPPLIES_BOOKS | V_ID | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | ISBN | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | QUANTITY | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | UNIT_PRICE | NUMBER | - | 5 | 2 | - | - | - | - |
| | DELIVERY_DATE | DATE | 7 | - | - | - | ✓ | SYSDATE | - |

1 - 5

# Copy_location:

```
CREATE TABLE Copy_location (
    Location VARCHAR2(20) PRIMARY KEY,
    Condition VARCHAR2(20),
    Availability VARCHAR2(5) CHECK (Availability IN ('Yes', 'No'))
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| COPY_LOCATION | LOCATION | VARCHAR2 | 15 | - | - | 1 | - | - | - |
| | CONDITION | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | AVAILABILITY | VARCHAR2 | 5 | - | - | - | ✓ | - | - |

1 - 3

# Copies:

```
CREATE TABLE Copies (
    Copy_id NUMBER(5) PRIMARY KEY,
    Location VARCHAR2(20),
    FOREIGN KEY (Location) REFERENCES Copy_location(Location)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| COPIES | COPY_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | LOCATION | VARCHAR2 | 15 | - | - | - | ✓ | - | - |

1 - 2

# Books_have_Copies:

```
CREATE TABLE Books_have_Copies (
    ISBN NUMBER(5),
    Copy_id NUMBER(5),
    Acquisition_date DATE NOT NULL,
    PRIMARY KEY (ISBN, Copy_id),
    FOREIGN KEY (ISBN) REFERENCES Book2(ISBN),
    FOREIGN KEY (Copy_id) REFERENCES Copies(Copy_id)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| BOOKS_HAVE_COPIES | ISBN | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | COPY_ID | NUMBER | - | 5 | 0 | 2 | - | - | - |
| | ACQUISITION_DATE | DATE | 7 | - | - | - | - | - | - |

1 - 3

# Book1:

```
CREATE TABLE Book1 (
    ISBN NUMBER(5) PRIMARY KEY,
    Title VARCHAR2(15),
    Genre VARCHAR2(15) NOT NULL,
    Mem_id NUMBER(10),
    Staff_ID NUMBER(5),
    V_id NUMBER(5),
    FOREIGN KEY (Title) REFERENCES Book_title_genre(Title),
    FOREIGN KEY (Mem_id) REFERENCES member(Mem_id),
```

```
    FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID),
    FOREIGN KEY (V_id) REFERENCES Vendor_book(V_id)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| BOOK1 | ISBN | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | TITLE | VARCHAR2 | 15 | - | - | - | ✓ | - | - |
| | GENRE | VARCHAR2 | 15 | - | - | - | - | - | - |
| | MEM_ID | NUMBER | - | 10 | 0 | - | ✓ | - | - |
| | STAFF_ID | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | V_ID | NUMBER | - | 5 | 0 | - | ✓ | - | - |

1 - 6

# Journal_1:

```
CREATE TABLE Journal_1 (
    DOI NUMBER(5) PRIMARY KEY,
    Issue DATE NOT NULL,
    ISBN NUMBER(5),
    FOREIGN KEY (ISBN) REFERENCES Book1(ISBN)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| JOURNAL_1 | DOI | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | ISSUE | DATE | 7 | - | - | - | - | - | - |
| | ISBN | NUMBER | - | 5 | 0 | - | ✓ | - | - |

1 - 3

# Journal_2:

```
CREATE TABLE Journal_2 (
    DOI NUMBER(5) PRIMARY KEY,
    Volume NUMBER(5,2) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| JOURNAL_2 | DOI | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | VOLUME | NUMBER | - | 5 | 2 | - | - | - | - |

1 - 2

# Newspaper_1:

```
CREATE TABLE Newspaper_1 (
    Newspaper_id NUMBER(5) PRIMARY KEY,
    Daate DATE NOT NULL,
    ISBN NUMBER(5),
    FOREIGN KEY (ISBN) REFERENCES Book2(ISBN)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| NEWSPAPER_1 | NEWSPAPER_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | DAATE | DATE | 7 | - | - | - | - | - | - |
| | ISBN | NUMBER | - | 5 | 0 | - | ✓ | - | - |

1 - 3

# Newspaper_2:

```
CREATE TABLE Newspaper_2 (
    Newspaper_id NUMBER(5),
    Headlines VARCHAR2(50),
    PRIMARY KEY (Newspaper_id, Headlines)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| NEWSPAPER_2 | NEWSPAPER_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | HEADLINES | VARCHAR2 | 30 | - | - | 2 | - | - | - |
| | | | | | | | | | 1 - 2 |

# Magazine2:

```
CREATE TABLE Magazine2 (
    Mag_id NUMBER(5) PRIMARY KEY,
    Topic VARCHAR2(20) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| MAGAZINE2 | MAG_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | TOPIC | VARCHAR2 | 20 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 2 |

# Mag_date_coverStory:

```
CREATE TABLE Mag_date_coverStory (
    Daate DATE PRIMARY KEY,
    Cover_Story VARCHAR2(10) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| MAG_DATE_COVERSTORY | DAATE | DATE | 7 | - | - | 1 | - | - | - |
| | COVER_STORY | VARCHAR2 | 10 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 2 |

# Magazine:

```
CREATE TABLE Magazine (
    Mag_id NUMBER(5) PRIMARY KEY,
    Cover_Story VARCHAR2(10) NOT NULL,
    Daate DATE,
    ISBN NUMBER(5),
    FOREIGN KEY (Daate) REFERENCES Mag_date_coverStory(Daate),
    FOREIGN KEY (ISBN) REFERENCES Book2(ISBN)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| MAGAZINE | MAG_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | COVER_STORY | VARCHAR2 | 10 | - | - | - | - | - | - |
| | DAATE | DATE | 7 | - | - | - | ✓ | - | - |
| | ISBN | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | | | | | | | | | 1 - 4 |

# Return_Borrow_Relationship:

```
CREATE TABLE Return_Borrow_Relationship (
    Borrow_date DATE PRIMARY KEY,
    Return_date DATE NOT NULL,
    Mem_id NUMBER(10),
    ISBN NUMBER(5),
    FOREIGN KEY (Mem_id) REFERENCES member(Mem_id),
```

```
    FOREIGN KEY (ISBN) REFERENCES Book2(ISBN)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| RETURN_BORROW_RELATIONSHIP | BORROW_DATE | DATE | 7 | - | - | 1 | - | - | - |
| | RETURN_DATE | DATE | 7 | - | - | - | - | - | - |
| | MEM_ID | NUMBER | - | 10 | 0 | - | ✓ | - | - |
| | ISBN | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | | | | | | | | | 1 - 4 |

# staff_position:

```
CREATE TABLE Lib_staff_position (
    stName VARCHAR2(20) PRIMARY KEY,
    Position VARCHAR2(15) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| STAFF_POSITION | DEPARTMENT | NUMBER | - | 2 | 0 | 1 | - | - | - |
| | POSITION | VARCHAR2 | 15 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 2 |

# Library_Staff1:

```
CREATE TABLE Library_Staff (
    Staff_id NUMBER(5) PRIMARY KEY,
    stName VARCHAR2(20),
    Position VARCHAR2(10) NOT NULL,
    FOREIGN KEY (stName) REFERENCES Lib_staff_position(stName)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| LIBRARY_STAFF | STAFF_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | STNAME | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | POSITION | VARCHAR2 | 10 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 3 |

# Library_Staff2:

```
CREATE TABLE Library_Staff2 (
    Staff_id NUMBER(5) PRIMARY KEY,
    Access_rights VARCHAR2(5) CHECK (Access_rights IN ('Yes', 'No'))
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| LIBRARY_STAFF2 | STAFF_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | ACCESS_RIGHTS | VARCHAR2 | 5 | - | - | - | ✓ | - | - |
| | | | | | | | | | 1 - 2 |

# Is_issued_by_Relationship:

```
CREATE TABLE Is_issued_by_Relationship (
    ISBN NUMBER(5),
    Issue_date DATE NOT NULL,
    Staff_ID NUMBER(5),
    PRIMARY KEY (ISBN, Staff_ID),
    FOREIGN KEY (ISBN) REFERENCES Book2(ISBN),
    FOREIGN KEY (Staff_ID) REFERENCES Library_Staff(Staff_id)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| IS_ISSUED_BY_RELATIONSHIP | ISBN | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | ISSUE_DATE | DATE | 7 | - | - | - | - | - | - |
| | STAFF_ID | NUMBER | - | 5 | 0 | 2 | - | - | - |
| | | | | | | | | | 1 - 3 |

# Record_of_fine_action:

```
CREATE TABLE Record_of_fine_action (
    fDate DATE PRIMARY KEY,
    Fine NUMBER(5) CHECK (Fine > 0),
    Action VARCHAR2(20)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| RECORD_OF_FINE_ACTION | FDATE | DATE | 7 | - | - | 1 | - | - | - |
| | FINE | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | ACTION | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | | | | | | | | | 1 - 3 |

# Record:

```
CREATE TABLE Record (
    Record_ID NUMBER(5) PRIMARY KEY,
    fDate DATE,
    Staff_ID NUMBER(5),
    FOREIGN KEY (fDate) REFERENCES Record_of_fine_action(fDate),
    FOREIGN KEY (Staff_ID) REFERENCES Library_Staff(Staff_ID)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| RECORD | RECORD_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | FDATE | DATE | 7 | - | - | - | ✓ | - | - |
| | STAFF_ID | NUMBER | - | 5 | 0 | - | ✓ | - | - |
| | | | | | | | | | 1 - 3 |

# Author_detail:

```
CREATE TABLE Author_detail (
    Name VARCHAR2(20) PRIMARY KEY,
    Nationality VARCHAR2(15) NOT NULL,
    Biography VARCHAR2(15) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| AUTHOR_DETAIL | NAME | VARCHAR2 | 20 | - | - | 1 | - | - | - |
| | NATIONALITY | VARCHAR2 | 15 | - | - | - | - | - | - |
| | BIOGRAPHY | VARCHAR2 | 15 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 3 |

# Author:

```
CREATE TABLE Author (
    A_ID NUMBER(5) PRIMARY KEY,
    Name VARCHAR2(20),
    FOREIGN KEY (Name) REFERENCES Author_detail(Name)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| AUTHOR | A_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | NAME | VARCHAR2 | 20 | - | - | - | ✓ | - | - |
| | | | | | | | | | 1 - 2 |

# Is_Written_by_2:

```
CREATE TABLE Is_Written_by_2 (
    A_ID NUMBER(5),
    ISBN NUMBER(5),
    Contribution VARCHAR2(10),
    PRIMARY KEY (A_ID, ISBN, Contribution),
    FOREIGN KEY (A_ID) REFERENCES Author(A_ID),
    FOREIGN KEY (ISBN) REFERENCES Book1(ISBN)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| IS_WRITTEN_BY_2 | A_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | ISBN | NUMBER | - | 5 | 0 | 2 | - | - | - |
| | CONTRIBUTION | VARCHAR2 | 10 | - | - | 3 | - | - | - |
| | | | | | | | | | 1 - 3 |

# Is_Written_by_1:

```
CREATE TABLE Is_Written_by_1 (
    A_ID NUMBER(5),
    ISBN NUMBER(5),
    Role VARCHAR2(10),
    PRIMARY KEY (A_ID, ISBN),
    FOREIGN KEY (A_ID) REFERENCES Author(A_ID),
    FOREIGN KEY (ISBN) REFERENCES Book1(ISBN)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| IS_WRITTEN_BY_1 | A_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | ISBN | NUMBER | - | 5 | 0 | 2 | - | - | - |
| | ROLE | VARCHAR2 | 10 | - | - | - | ✓ | - | - |
| | | | | | | | | | 1 - 3 |

# Publisher:

```
CREATE TABLE Publisher (
    Publisher_ID NUMBER(5) PRIMARY KEY,
    CEO VARCHAR2(20) NOT NULL,
    Contact_info NUMBER(11) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| PUBLISHER | PUBLISHER_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | CEO | VARCHAR2 | 20 | - | - | - | - | - | - |
| | CONTACT_INFO | NUMBER | - | 11 | 0 | - | - | - | - |
| | | | | | | | | | 1 - 3 |

# Publish_multiValued:

```
CREATE TABLE Publish_multiValued (
    Publisher_ID NUMBER(5) PRIMARY KEY,
    Head_quarter VARCHAR2(20) NOT NULL
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| PUBLISH_MULTIVALUED | PUBLISHER_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | HEAD_QUARTER | VARCHAR2 | 20 | - | - | - | - | - | - |
| | | | | | | | | | 1 - 2 |

# Is_publish_by_Relationship:

```
CREATE TABLE Is_publish_by_Relationship (
    Publisher_ID NUMBER(5),
    ISBN NUMBER(5),
    Publish_date DATE DEFAULT SYSDATE,
    PRIMARY KEY (Publisher_ID, ISBN),
    FOREIGN KEY (Publisher_ID) REFERENCES Publisher(Publisher_ID),
FOREIGN KEY (ISBN) REFERENCES Book1(ISBN)
);
```

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|---|---|---|---|---|---|---|---|---|---|
| IS_PUBLISH_BY_RELATIONSHIP | PUBLISHER_ID | NUMBER | - | 5 | 0 | 1 | - | - | - |
| | ISBN | NUMBER | - | 5 | 0 | 2 | - | - | - |
| | PUBLISH_DATE | DATE | 7 | - | - | - | ✓ | SYSDATE | - |

1 - 3

# Dummy Data Entry

## Member:

```
INSERT INTO member (Mem_id, Name, Area, City, H_NO, ST_No)
VALUES (1001, 'John Doe', 'Downtown', 'New York', 123, 45);
INSERT INTO member (Mem_id, Name, Area, City, H_NO, ST_No)
VALUES (1002, 'Jane Smith', 'Suburbia', 'Los Angeles', 456, 67);
INSERT INTO member (Mem_id, Name, Area, City, H_NO, ST_No)
VALUES (1003, 'Michael Johnson', 'Rural Area', 'Chicago', 789, 89);
```

## City_Area:

```
INSERT INTO Phone_Number (Mem_id, Phone)
VALUES (1001, 12345678901);
INSERT INTO Phone_Number (Mem_id, Phone)
VALUES (1002, 23456789012);
INSERT INTO Phone_Number (Mem_id, Phone)
VALUES (1003, 34567890123);
```

## Student:

```
INSERT INTO Student_new (Stu_ID, Major, Grade, Mem_id)
VALUES (1, 'Computer Science', 'A', 1001);
ALTER TABLE Student_new MODIFY Major VARCHAR2(50);
INSERT INTO Student_new (Stu_ID, Major, Grade, Mem_id)
VALUES (2, 'Engineering', 'B', 1002);
INSERT INTO Student_new (Stu_ID, Major, Grade, Mem_id)
VALUES (3, 'Mathematics', 'C', 1003);
```

## Teacher:

```
INSERT INTO Teacher (T_ID, Department, Mem_id)
VALUES (1, 1, 1001);
INSERT INTO Teacher (T_ID, Department, Mem_id)
VALUES (2, 2, 1002);
INSERT INTO Teacher (T_ID, Department, Mem_id)
VALUES (3, 3, 1003);
```

## Teacher_subject:

```
INSERT INTO Teacher_subject (T_ID, Subject)
VALUES (1, 'Mathematics');
ALTER TABLE Teacher_subject MODIFY Subject VARCHAR2(50);
INSERT INTO Teacher_subject (T_ID, Subject)
VALUES (2, 'Science');
INSERT INTO Teacher_subject (T_ID, Subject)
VALUES (3, 'English');
```

## Staff:

```
INSERT INTO Staff (Staff_ID, Position, Department, Mem_id)
VALUES (1, 'Librarian', 1, 1001);
ALTER TABLE Staff MODIFY Position VARCHAR2(30);
INSERT INTO Staff (Staff_ID, Position, Department, Mem_id)
VALUES (2, 'Assistant Librarian', 2, 1002);
INSERT INTO Staff (Staff_ID, Position, Department, Mem_id)
VALUES (3, 'Library Clerk', 3, 1003);
```

## Staff_position:

```
INSERT INTO Staff_position (Department, Position)
VALUES (1, 'Librarian');
ALTER TABLE Staff_position MODIFY Position VARCHAR2(30);
INSERT INTO Staff_position (Department, Position)
VALUES (2, 'Assistant Librarian');
INSERT INTO Staff_position (Department, Position)
VALUES (3, 'Library Clerk');
```

## Book2:

```
INSERT INTO Book2 (ISBN, Publisher, Author)
VALUES (1, 'Publisher A', 'Author X');
INSERT INTO Book2 (ISBN, Publisher, Author)
VALUES (2, 'Publisher B', 'Author Y');
INSERT INTO Book2 (ISBN, Publisher, Author)
VALUES (3, 'Publisher C', 'Author Z');
```

## Book_title_genre:

```
INSERT INTO Book_title_genre (Title, Genre)
VALUES ('Book 1', 'Fiction');
INSERT INTO Book_title_genre (Title, Genre)
VALUES ('Book 2', 'Science Fiction');
INSERT INTO Book_title_genre (Title, Genre)
VALUES ('Book 3', 'Mystery');
```

## Vendor_phone:

INSERT INTO Vendor_phone (V_id, Phone#)

VALUES (1, 12345678901);

INSERT INTO Vendor_phone (V_id, Phone#)

VALUES (2, 23456789012);

INSERT INTO Vendor_phone (V_id, Phone#)

VALUES (3, 34567890123);

## Vendor_book:

INSERT INTO Vendor_book (V_id, V_Name, Email)

VALUES (1, 'Vendor A', 'husna@gmail.com');

INSERT INTO Vendor_book (V_id, V_Name, Email)

VALUES (2, 'Vendor B', 'sadi@gmail.com');

INSERT INTO Vendor_book (V_id, V_Name, Email)

VALUES (3, 'Vendor C', 'eman@gmail.com');

## VENDOR_SUPPLIES_BOOKS:

INSERT INTO VENDOR_SUPPLIES_BOOKS (V_ID, ISBN, QUANTITY, UNIT_PRICE, DELIVERY_DATE)

VALUES (1, 1, 50, 20.00, SYSDATE);

INSERT INTO VENDOR_SUPPLIES_BOOKS (V_ID, ISBN, QUANTITY, UNIT_PRICE, DELIVERY_DATE)

VALUES (2, 2, 30, 18.50, SYSDATE);

INSERT INTO VENDOR_SUPPLIES_BOOKS (V_ID, ISBN, QUANTITY, UNIT_PRICE, DELIVERY_DATE)

VALUES (3, 3, 40, 25.99, SYSDATE);

## Copy_location:

```
INSERT INTO Copy_location (Location, Condition, Availability)
VALUES ('Main Library', 'Good', 'Yes');
ALTER TABLE Copy_location MODIFY Location VARCHAR2(20);
INSERT INTO Copy_location (Location, Condition, Availability)
VALUES ('Reference Section', 'Excellent', 'Yes');
INSERT INTO Copy_location (Location, Condition, Availability)
VALUES ('Fiction Section', 'Fair', 'No');
```

## Copies:

```
INSERT INTO Copies (Copy_id, Location)
VALUES (1, 'Main Library');
INSERT INTO Copies (Copy_id, Location)
VALUES (2, 'Fiction Section');
ALTER TABLE Copies MODIFY Location VARCHAR2(20);
INSERT INTO Copies (Copy_id, Location)
VALUES (3, 'Reference Section');
```

## Books_have_Copies:

```
INSERT INTO Books_have_Copies (ISBN, Copy_id, Acquisition_date)
VALUES (1, 1, SYSDATE);
INSERT INTO Books_have_Copies (ISBN, Copy_id, Acquisition_date)
VALUES (2, 2, SYSDATE);
INSERT INTO Books_have_Copies (ISBN, Copy_id, Acquisition_date)
VALUES (3, 3, SYSDATE);
```

## Book1:

```
INSERT INTO Book1 (ISBN, Title, Genre, Mem_id, Staff_ID, V_id)
VALUES (1, 'Book 1', 'Fiction', 1001, 1, 1);
INSERT INTO Book1 (ISBN, Title, Genre, Mem_id, Staff_ID, V_id)
VALUES (2, 'Book 2', 'Science Fiction', 1002, 2, 2);
INSERT INTO Book1 (ISBN, Title, Genre, Mem_id, Staff_ID, V_id)
VALUES (3, 'Book 3', 'Mystery', 1003, 3, 3);
```

## Journal_1:

```
INSERT INTO Journal_1 (DOI, Issue, ISBN)
VALUES (1, TO_DATE('2024-05-01', 'YYYY-MM-DD'), 1);
INSERT INTO Journal_1 (DOI, Issue, ISBN)
VALUES (2, TO_DATE('2024-05-05', 'YYYY-MM-DD'), 2);
INSERT INTO Journal_1 (DOI, Issue, ISBN)
VALUES (3, TO_DATE('2024-05-10', 'YYYY-MM-DD'), 3);
```

## Journal_2:

```
INSERT INTO Journal_2 (DOI, Volume)
VALUES (1, 10.5);
INSERT INTO Journal_2 (DOI, Volume)
VALUES (2, 15.2);
INSERT INTO Journal_2 (DOI, Volume)
VALUES (3, 20.7);
```

## Newspaper_1:

```
INSERT INTO Newspaper_1 (Newspaper_id, Daate, ISBN)
VALUES (1, TO_DATE('2024-05-01', 'YYYY-MM-DD'), 1);
INSERT INTO Newspaper_1 (Newspaper_id, Daate, ISBN)
VALUES (2, TO_DATE('2024-05-05', 'YYYY-MM-DD'), 2);
INSERT INTO Newspaper_1 (Newspaper_id, Daate, ISBN)
VALUES (3, TO_DATE('2024-05-10', 'YYYY-MM-DD'), 3);
```

## Newspaper_2:

```
INSERT INTO Newspaper_2 (Newspaper_id, Headlines)
VALUES (1, 'Breaking News: Weather Alert');
ALTER TABLE Newspaper_2 MODIFY Headlines VARCHAR2(50);
INSERT INTO Newspaper_2 (Newspaper_id, Headlines)
VALUES (2, 'Local Events: Community Gathering');
INSERT INTO Newspaper_2 (Newspaper_id, Headlines)
VALUES (3, 'Sports News: Championship Results');
```

## Magazine2:

```
INSERT INTO Magazine2 (Mag_id, Topic)
VALUES (1, 'Science');
INSERT INTO Magazine2 (Mag_id, Topic)
VALUES (2, 'Technology');
INSERT INTO Magazine2 (Mag_id, Topic)
VALUES (3, 'Art');
```

## Mag_date_coverStory:

ALTER TABLE Mag_date_coverStory MODIFY Cover_Story VARCHAR2(30);

INSERT INTO Mag_date_coverStory (Daate, Cover_Story)

VALUES (TO_DATE('2024-05-01', 'YYYY-MM-DD'), 'Space Exploration');

INSERT INTO Mag_date_coverStory (Daate, Cover_Story)

VALUES (TO_DATE('2024-05-05', 'YYYY-MM-DD'), 'Artificial Intelligence');

INSERT INTO Mag_date_coverStory (Daate, Cover_Story)

VALUES (TO_DATE('2024-05-10', 'YYYY-MM-DD'), 'Climate Change');

## Magazine1:

ALTER TABLE Magazine1 MODIFY Cover_Story VARCHAR2(30);

INSERT INTO Magazine1 (Mag_id, Cover_Story, Daate, ISBN)

VALUES (1, 'Space Exploration', TO_DATE('2024-05-01', 'YYYY-MM-DD'), 1);

INSERT INTO Magazine 1(Mag_id, Cover_Story, Daate, ISBN)

VALUES (2, 'Artificial Intelligence', TO_DATE('2024-05-05', 'YYYY-MM-DD'), 2);

INSERT INTO Magazine 1(Mag_id, Cover_Story, Daate, ISBN)

VALUES (3, 'Climate Change', TO_DATE('2024-05-10', 'YYYY-MM-DD'), 3);

## Return_Borrow_Relationship:

INSERT INTO Return_Borrow_Relationship (Borrow_date, Return_date, Mem_id, ISBN)

VALUES (TO_DATE('2024-05-01', 'YYYY-MM-DD'), TO_DATE('2024-05-15', 'YYYY-MM-DD'), 1001, 1);

INSERT INTO Return_Borrow_Relationship (Borrow_date, Return_date, Mem_id, ISBN)

VALUES (TO_DATE('2024-05-03', 'YYYY-MM-DD'), TO_DATE('2024-05-18', 'YYYY-MM-DD'), 1002, 2);

INSERT INTO Return_Borrow_Relationship (Borrow_date, Return_date, Mem_id, ISBN)

VALUES (TO_DATE('2024-05-05', 'YYYY-MM-DD'), TO_DATE('2024-05-20', 'YYYY-MM-DD'), 1003, 3);

# Lib_staff_position:

INSERT INTO Lib_staff_position (stName, Position)

VALUES ('John Doe', 'Librarian');

INSERT INTO Lib_staff_position (stName, Position)

VALUES ('Jane Smith', 'Assistant');

INSERT INTO Lib_staff_position (stName, Position)

VALUES ('Michael Johnson', 'Clerk');

# Library_Staff:

INSERT INTO Library_Staff (Staff_id, stName, Position)

VALUES (1, 'John Doe', 'Librarian');

INSERT INTO Library_Staff (Staff_id, stName, Position)

VALUES (2, 'Jane Smith', 'Assistant');

INSERT INTO Library_Staff (Staff_id, stName, Position)

VALUES (3, 'Michael Johnson', 'Clerk');

# Library_Staff2:

INSERT INTO Library_Staff2 (Staff_id, Access_rights)

VALUES (1, 'Yes');

```
INSERT INTO Library_Staff2 (Staff_id, Access_rights)
VALUES (2, 'No');
INSERT INTO Library_Staff2 (Staff_id, Access_rights)
VALUES (3, 'Yes');
```

## Is_issued_by_Relationship:

```
INSERT INTO Is_issued_by_Relationship (ISBN, Issue_date, Staff_ID)
VALUES (1, TO_DATE('2023-01-15', 'YYYY-MM-DD'), 1);
INSERT INTO Is_issued_by_Relationship (ISBN, Issue_date, Staff_ID)
VALUES (2, TO_DATE('2023-02-20', 'YYYY-MM-DD'), 2);
INSERT INTO Is_issued_by_Relationship (ISBN, Issue_date, Staff_ID)
VALUES (3, TO_DATE('2023-03-10', 'YYYY-MM-DD'), 3);
```

## Record_of_fine_action:

```
INSERT INTO Record_of_fine_action (fDate, Fine, Action)
VALUES (TO_DATE('2023-01-10', 'YYYY-MM-DD'), 50, 'Late Return');
INSERT INTO Record_of_fine_action (fDate, Fine, Action)
VALUES (TO_DATE('2023-02-05', 'YYYY-MM-DD'), 30, 'Damaged Book');
INSERT INTO Record_of_fine_action (fDate, Fine, Action)
VALUES (TO_DATE('2023-03-15', 'YYYY-MM-DD'), 20, 'Lost Book');
```

## Record:

```
INSERT INTO Record (Record_ID, fDate, Staff_ID)
VALUES (1, TO_DATE('2023-01-10', 'YYYY-MM-DD'), 1);
INSERT INTO Record (Record_ID, fDate, Staff_ID)
VALUES (2, TO_DATE('2023-02-05', 'YYYY-MM-DD'), 2);
```

```
INSERT INTO Record (Record_ID, fDate, Staff_ID)
VALUES (3, TO_DATE('2023-03-15', 'YYYY-MM-DD'), 3);
```

## Author_detail:

```
INSERT INTO Author_detail (Name, Nationality, Biography)
VALUES ('Author X', 'American', 'Bio X');
INSERT INTO Author_detail (Name, Nationality, Biography)
VALUES ('Author Y', 'British', 'Bio Y');
INSERT INTO Author_detail (Name, Nationality, Biography)
VALUES ('Author Z', 'Canadian', 'Bio Z');
```

## Author:

```
INSERT INTO Author (A_ID, Name)
VALUES (1, 'Author X');
INSERT INTO Author (A_ID, Name)
VALUES (2, 'Author Y');
INSERT INTO Author (A_ID, Name)
VALUES (3, 'Author Z');
```

## Is_Written_by_2:

```
INSERT INTO Is_Written_by_2 (A_ID, ISBN, Contribution)
VALUES (1, 1, 'Lead');
INSERT INTO Is_Written_by_2 (A_ID, ISBN, Contribution)
VALUES (2, 2, 'Co-author');
INSERT INTO Is_Written_by_2 (A_ID, ISBN, Contribution)
VALUES (3, 3, 'Editor');
```

## Is_Written_by_1:

INSERT INTO Is_Written_by_1 (A_ID, ISBN, Role)

VALUES (1, 1, 'Author');

INSERT INTO Is_Written_by_1 (A_ID, ISBN, Role)

VALUES (2, 2, 'Author');

INSERT INTO Is_Written_by_1 (A_ID, ISBN, Role)

VALUES (3, 3, 'Author');

## Publisher:

INSERT INTO Publisher (Publisher_ID, CEO, Contact_info)

VALUES (1, 'Alice Johnson', 12345678901);

INSERT INTO Publisher (Publisher_ID, CEO, Contact_info)

VALUES (2, 'Bob Smith', 23456789012);

INSERT INTO Publisher (Publisher_ID, CEO, Contact_info)

VALUES (3, 'Carol White', 34567890123);

## Publish_multiValued:

INSERT INTO Publish_multiValued (Publisher_ID, Head_quarter)

VALUES (1, 'New York');

INSERT INTO Publish_multiValued (Publisher_ID, Head_quarter)

VALUES (2, 'Los Angeles');

INSERT INTO Publish_multiValued (Publisher_ID, Head_quarter)

VALUES (3, 'Chicago');

## Is_publish_by_Relationship:

```
INSERT INTO Is_publish_by_Relationship (Publisher_ID, ISBN, Publish_date)
VALUES (1, 1, TO_DATE('2022-01-15', 'YYYY-MM-DD'));
INSERT INTO Is_publish_by_Relationship (Publisher_ID, ISBN, Publish_date)
VALUES (2, 2, TO_DATE('2023-03-22', 'YYYY-MM-DD'));
INSERT INTO Is_publish_by_Relationship (Publisher_ID, ISBN, Publish_date)
VALUES (3, 3, TO_DATE('2024-05-28', 'YYYY-MM-DD'));
```

# Queries:

1. **Find the details of members who borrowed books written by a specific author and were issued by a specific staff member within a given date range.**

```
SELECT m.Mem_id, m.Name AS Member_Name, s.stName AS Staff_Name,
b1.Title AS Book_Title, b1.Genre AS Book_Genre, rbr.Borrow_date,
rbr.Return_date
FROM Return_Borrow_Relationship rbr
JOIN Member m ON rbr.Mem_id = m.Mem_id
JOIN Is_issued_by_Relationship iibr ON rbr.ISBN = iibr.ISBN
JOIN Book1 b1 ON rbr.ISBN = b1.ISBN
JOIN Is_Written_by_1 iwb1 ON b1.ISBN = iwb1.ISBN
JOIN Author a ON iwb1.A_ID = a.A_ID
JOIN Library_Staff s ON iibr.Staff_ID = s.Staff_id
WHERE a.Name = 'Author X'
AND iibr.Staff_ID = 1
```

AND rbr.Borrow_date BETWEEN TO_DATE('2024-05-01', 'YYYY-MM-DD') AND TO_DATE('2024-05-15', 'YYYY-MM-DD');

| MEM_ID | MEMBER_NAME | STAFF_NAME | BOOK_TITLE | BOOK_GENRE | BORROW_DATE | RETURN_DATE |
|---|---|---|---|---|---|---|
| 1001 | Husna | John Doe | Book 1 | Fiction | 05/01/2024 | 05/15/2024 |

## 2. List all books along with their genres and the number of copies available in each location.

```
SELECT b.Title AS Book_Title,  b.Genre AS Book_Genre, cl.Location AS Copy_Location, COUNT(bhc.Copy_id) AS Num_of_Copies_Available
FROM Book1 b
JOIN Books_have_Copies bhc ON b.ISBN = bhc.ISBN
JOIN Copies c ON bhc.Copy_id = c.Copy_id
JOIN Copy_location cl ON c.Location = cl.Location
GROUP BY b.Title, b.Genre, cl.Location;
```

| BOOK_TITLE | BOOK_GENRE | COPY_LOCATION | NUM_OF_COPIES_AVAILABLE |
|---|---|---|---|
| Book 1 | Fiction | Main Library | 1 |
| Book 2 | Science Fiction | Fiction Section | 1 |
| Book 3 | Mystery | Reference Section | 1 |

## 3. Get the detailed list of books issued by each staff member including member details and the fine imposed if any.

```
SELECT s.Staff_ID,  ls.stName AS Staff_Name, r.Borrow_date, r.Return_date, m.Mem_id,
 m.Name AS Member_Name, m.Area AS Member_Area, m.City AS Member_City, m.H_NO AS Member_House_No, m.ST_No AS Member_Street_No,
      CASE
```

```
        WHEN r.Return_date > TO_DATE('2024-05-28', 'YYYY-MM-DD')
THEN (r.Return_date - TO_DATE('2024-05-28', 'YYYY-MM-DD')) * 10
        ELSE NULL
    END AS Fine_Imposed
FROM Is_issued_by_Relationship iibr
JOIN Library_Staff ls ON iibr.Staff_ID = ls.Staff_id
JOIN Staff s ON iibr.Staff_ID = s.Staff_ID
JOIN Return_Borrow_Relationship r ON iibr.ISBN = r.ISBN
JOIN member m ON r.Mem_id = m.Mem_id;
```

| STAFF_ID | STAFF_NAME | BORROW_DATE | RETURN_DATE | MEM_ID | MEMBER_NAME | MEMBER_AREA | MEMBER_CITY | MEMBER_HOUSE_NO | MEMBER_STREET_NO | FINE_IMPOSED |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | John Doe | 05/01/2024 | 05/15/2024 | 1001 | Husna | Downtown | New York | 123 | 45 | - |
| 2 | Jane Smith | 05/03/2024 | 05/18/2024 | 1002 | Eman | Suburbia | Los Angeles | 456 | 67 | - |
| 3 | Michael Johnson | 05/05/2024 | 05/20/2024 | 1003 | Sadia | Rural Area | Chicago | 789 | 89 | - |

## 4. List the magazines with their cover stories, topics, and the names of authors who contributed to them.

```
SELECT m.Cover_Story,
    mg.Topic,
    a.Name AS Author_Name
FROM Magazine m
JOIN Is_Written_by_1 wb1 ON m.ISBN = wb1.ISBN
JOIN Author a ON wb1.A_ID = a.A_ID
JOIN Magazine2 mg ON m.Mag_id = mg.Mag_id;
```

| COVER_STORY | TOPIC | AUTHOR_NAME |
|---|---|---|
| Space Exploration | Science | Author X |
| Artificial Intelligence | Technology | Author Y |
| Climate Change | Art | Author Z |

5. **Find the vendors who supplied books along with the quantity and total cost of books supplied.**

```
SELECT v.V_id,
    v.V_Name,
    SUM(vs.Quantity) AS Total_Quantity,
    SUM(vs.Quantity * vs.Unit_price) AS Total_Cost
FROM Vendor_book v
JOIN Vendor_Supplies_Books vs ON v.V_id = vs.V_id
GROUP BY v.V_id, v.V_Name;
```

| V_ID | V_NAME | TOTAL_QUANTITY | TOTAL_COST |
|---|---|---|---|
| 1 | Vendor A | 50 | 1000 |
| 3 | Vendor C | 40 | 1039.6 |
| 2 | Vendor B | 30 | 555 |

6. **Retrieve the borrowing history of a specific member, including book details and the staff member who issued the book.**

```
SELECT
   m.Mem_id AS Member_ID,
   m.Name AS Member_Name,
   rb.Borrow_date,
   rb.Return_date,
   b.Title,
   b2.Author,
   bg.Genre,
   ls.stName AS Issuing_Staff_Name,
   ls.Position AS Issuing_Staff_Position
FROM
```

```
    Return_Borrow_Relationship rb
JOIN
    Book1 b ON rb.ISBN = b.ISBN
JOIN
    Book2 b2 ON b.ISBN = b2.ISBN
JOIN
    Book_title_genre bg ON b.Title = bg.Title
JOIN
    Is_issued_by_Relationship iib ON rb.ISBN = iib.ISBN
JOIN
    Library_Staff ls ON iib.Staff_ID = ls.Staff_id
JOIN
    Member m ON rb.Mem_id = m.Mem_id
WHERE
    rb.Mem_id = 1003;
```

| MEMBER_ID | MEMBER_NAME | BORROW_DATE | RETURN_DATE | TITLE | AUTHOR | GENRE | ISSUING_STAFF_NAME | ISSUING_STAFF_POSITION |
|---|---|---|---|---|---|---|---|---|
| 1003 | Sadia | 05/05/2024 | 05/20/2024 | Book 3 | Author Z | Mystery | Michael Johnson | Clerk |

## 7. Retrieve the list of journals along with their volumes and issues published by a specific publisher.

```
SELECT
    j1.DOI AS Journal_ID,
    j1.Issue AS Issue_Date,
    j2.Volume AS Journal_Volume,
    p.Publisher_ID AS Publisher_ID,
    p.CEO AS Publisher_CEO,
    pmv.Head_quarter AS Publisher_HQ
```

```
FROM
   Journal_1 j1
JOIN
   Journal_2 j2 ON j1.DOI = j2.DOI
JOIN
   Is_publish_by_Relationship ipb ON j1.ISBN = ipb.ISBN
JOIN
   Publisher p ON ipb.Publisher_ID = p.Publisher_ID
JOIN
   Publish_multiValued pmv ON p.Publisher_ID = pmv.Publisher_ID
WHERE
p.Publisher_ID = 1;
```

| JOURNAL_ID | ISSUE_DATE | JOURNAL_VOLUME | PUBLISHER_ID | PUBLISHER_CEO | PUBLISHER_HQ |
|---|---|---|---|---|---|
| 1 | 05/01/2024 | 10.5 | 1 | Alice Johnson | New York |

# VIEWS

## This view provides details about books, their authors, and the number of copies available in each location.

```
CREATE VIEW Book_Details AS
SELECT b.Title, b.Genre, a.Name AS Author,
    (SELECT COUNT(*) FROM Books_have_Copies bc WHERE bc.ISBN = b.ISBN) AS Total_Copies,
    l.Location, l.Condition, l.Availability
FROM Book1 b
JOIN Is_Written_by_1 wb1 ON b.ISBN = wb1.ISBN
```

JOIN Author a ON wb1.A_ID = a.A_ID

LEFT JOIN Books_have_Copies bc ON b.ISBN = bc.ISBN

LEFT JOIN Copies c ON bc.Copy_id = c.Copy_id

LEFT JOIN Copy_location l ON c.Location = l.Location;

| TITLE | GENRE | AUTHOR | TOTAL_COPIES | LOCATION | CONDITION | AVAILABILITY |
|---|---|---|---|---|---|---|
| Book 1 | Fiction | Author X | 1 | Main Library | Good | Yes |
| Book 2 | Science Fiction | Author Y | 1 | Fiction Section | Fair | No |
| Book 3 | Mystery | Author Z | 1 | Reference Section | Excellent | Yes |

# TRIGGERS:

## 1. Comprehensive Audit Trigger

```
CREATE TRIGGER trg_audit_book_changes
AFTER INSERT OR UPDATE OR DELETE ON Book1
FOR EACH ROW
BEGIN
    DECLARE action_type VARCHAR(10);
    DECLARE user_id INT;
    DECLARE old_values TEXT;
    DECLARE new_values TEXT;

    -- Simulating user identification
    SELECT current_user_id INTO user_id FROM current_user_session;

    IF (TG_OP = 'INSERT') THEN
        action_type := 'INSERT';
```

```
        old_values := NULL;
        new_values := 'ISBN: ' || NEW.ISBN || ', Title: ' || NEW.Title || ', Author_ID: ' || NEW.Author_ID || ', Genre: ' || NEW.Genre || ', Publisher: ' || NEW.Publisher || ', Year: ' || NEW.Year || ', Quantity: ' || NEW.Quantity;
    ELSIF (TG_OP = 'UPDATE') THEN
        action_type := 'UPDATE';
        old_values := 'ISBN: ' || OLD.ISBN || ', Title: ' || OLD.Title || ', Author_ID: ' || OLD.Author_ID || ', Genre: ' || OLD.Genre || ', Publisher: ' || OLD.Publisher || ', Year: ' || OLD.Year || ', Quantity: ' || OLD.Quantity;
        new_values := 'ISBN: ' || NEW.ISBN || ', Title: ' || NEW.Title || ', Author_ID: ' || NEW.Author_ID || ', Genre: ' || NEW.Genre || ', Publisher: ' || NEW.Publisher || ', Year: ' || NEW.Year || ', Quantity: ' || NEW.Quantity;
    ELSIF (TG_OP = 'DELETE') THEN
        action_type := 'DELETE';
        old_values := 'ISBN: ' || OLD.ISBN || ', Title: ' || OLD.Title || ', Author_ID: ' || OLD.Author_ID || ', Genre: ' || OLD.Genre || ', Publisher: ' || OLD.Publisher || ', Year: ' || OLD.Year || ', Quantity: ' || OLD.Quantity;
        new_values := NULL;
    END IF;

    INSERT INTO Audit_Log (action_type, action_timestamp, user_id, old_values, new_values)
    VALUES (action_type, NOW(), user_id, old_values, new_values);
END;
```

# 2. Borrowing and Return Management Trigger

```
CREATE TRIGGER trg_manage_borrowing
BEFORE INSERT OR DELETE ON Return_Borrow_Relationship
FOR EACH ROW
BEGIN
    DECLARE book_quantity INT;
```

```
    DECLARE days_late INT;
    DECLARE fine_amount DECIMAL(10,2);
    DECLARE user_id INT;
    DECLARE due_date DATE;

    IF (TG_OP = 'INSERT') THEN
        -- Check book availability
        SELECT Quantity INTO book_quantity FROM Vendor_Supplies_Books
WHERE ISBN = NEW.ISBN;
        IF (book_quantity <= 0) THEN
            RAISE EXCEPTION 'Insufficient quantity for borrowing.';
        ELSE
            UPDATE Vendor_Supplies_Books
            SET Quantity = Quantity - 1
            WHERE ISBN = NEW.ISBN;
        END IF;

        -- Set initial due date (e.g., 14 days from borrow date)
        SET due_date = NEW.borrow_date + INTERVAL '14 days';
        UPDATE Return_Borrow_Relationship
        SET due_date = due_date
        WHERE ISBN = NEW.ISBN AND user_id = NEW.user_id;

    ELSIF (TG_OP = 'DELETE') THEN
        -- Increment book quantity on return
        UPDATE Vendor_Supplies_Books
        SET Quantity = Quantity + 1
        WHERE ISBN = OLD.ISBN;
```

```
        -- Calculate fine if return is late
        SELECT DATEDIFF(NOW(), OLD.due_date) INTO days_late;
        IF (days_late > 0) THEN
            fine_amount := days_late * 0.50; -- Assume $0.50 per day fine
            UPDATE Users
            SET fine_balance = fine_balance + fine_amount
            WHERE user_id = OLD.user_id;
        END IF;
    END IF;
END;
```

# Trigger for Recording Fine Actions:

```
CREATE TRIGGER trg_record_fine_action
AFTER INSERT ON Return_Borrow_Relationship
FOR EACH ROW
BEGIN

    INSERT INTO Record_of_fine_action (fDate, Fine, Action)
    VALUES (NEW.Return_date, CASE WHEN NEW.Return_date >
NEW.Borrow_date THEN 0 ELSE 50 END, 'Late Return');
END;
```

# PROCEDURES:

## Generate Book Report using procedures:

```
CREATE OR REPLACE PROCEDURE Generate_Book_Report AS
   CURSOR book_cursor IS
      SELECT b.Title, b.Genre, a.Name AS Author, cl.Location, cl.Condition,
cl.Availability
      FROM Book1 b
      JOIN Books_have_Copies bh ON b.ISBN = bh.ISBN
      JOIN Copies c ON bh.Copy_id = c.Copy_id
      JOIN Copy_location cl ON c.Location = cl.Location
      JOIN Is_Written_by_1 wb ON b.ISBN = wb.ISBN
      JOIN Author a ON wb.A_ID = a.A_ID;
BEGIN
   FOR book_rec IN book_cursor LOOP
      DBMS_OUTPUT.PUT_LINE('Title: ' || book_rec.Title || ', Genre: ' ||
book_rec.Genre ||
                           ', Author: ' || book_rec.Author || ', Location: ' ||
book_rec.Location ||
                           ', Condition: ' || book_rec.Condition || ', Availability: ' ||
book_rec.Availability);
   END LOOP;
END Generate_Book_Report;
```

```
Title: Book 1, Genre: Fiction, Author: Author X, Location: Main Library, Condition: Good, Availability: Yes
Title: Book 2, Genre: Science Fiction, Author: Author Y, Location: Fiction Section, Condition: Fair, Availability: No
Title: Book 3, Genre: Mystery, Author: Author Z, Location: Reference Section, Condition: Excellent, Availability: Yes
```

# Procedure : Retrieve Library information:

```
CREATE OR REPLACE PROCEDURE retrieve_library_info AS
BEGIN
   -- Retrieve member details and their borrowed books without directly referencing the Book2 table
   DBMS_OUTPUT.PUT_LINE('Member Details and Borrowed Books:');
   FOR r IN (
      SELECT m.Mem_id, m.Name, m.Area, m.City, rb.ISBN, rb.Borrow_date, rb.Return_date
      FROM member m
      JOIN Return_Borrow_Relationship rb ON m.Mem_id = rb.Mem_id
   ) LOOP
      DBMS_OUTPUT.PUT_LINE('Member ID: ' || r.Mem_id || ', Name: ' || r.Name || ', Area: ' || r.Area ||
                           ', City: ' || r.City || ', Borrowed Book ISBN: ' || r.ISBN ||
                           ', Borrow Date: ' || r.Borrow_date || ', Return Date: ' || r.Return_date);
   END LOOP;

   -- Retrieve book copies and their availability without directly referencing the Book2 table
   DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Book Copies and Availability:');
   FOR r IN (
      SELECT bhc.ISBN, c.Copy_id, c.Location, cl.Availability
      FROM Books_have_Copies bhc
      JOIN Copies c ON bhc.Copy_id = c.Copy_id
      JOIN Copy_location cl ON c.Location = cl.Location
   ) LOOP
```

```
        DBMS_OUTPUT.PUT_LINE('ISBN: ' || r.ISBN || ', Copy ID: ' || r.Copy_id
||
                        ', Location: ' || r.Location || ', Availability: ' || r.Availability);
    END LOOP;

    -- Retrieve fines and actions taken against members
    DBMS_OUTPUT.PUT_LINE(CHR(10) || 'Fines and Actions Taken:');
    FOR r IN (
        SELECT rfa.fDate, rfa.Fine, rfa.Action, m.Mem_id, m.Name
        FROM Record_of_fine_action rfa
        JOIN Record r ON rfa.fDate = r.fDate
        JOIN member m ON r.Staff_ID = m.Mem_id
    ) LOOP
        DBMS_OUTPUT.PUT_LINE('Fine Date: ' || r.fDate || ', Fine: ' || r.Fine || ',
Action: ' || r.Action ||
                        ', Member ID: ' || r.Mem_id || ', Member Name: ' || r.Name);
    END LOOP;
END retrieve_library_info;
```

```
Member Details and Borrowed Books:
Member ID: 1001, Name: Husna, Area: Downtown, City: New York, Borrowed Book ISBN: 1, Borrow Date: 05/01/2024, Return Date: 05/15/2024
Member ID: 1002, Name: Eman, Area: Suburbia, City: Los Angeles, Borrowed Book ISBN: 2, Borrow Date: 05/03/2024, Return Date: 05/18/2024
Member ID: 1003, Name: Sadia, Area: Rural Area, City: Chicago, Borrowed Book ISBN: 3, Borrow Date: 05/05/2024, Return Date: 05/20/2024

Book Copies and Availability:
ISBN: 1, Copy ID: 1, Location: Main Library, Availability: Yes
ISBN: 2, Copy ID: 2, Location: Fiction Section, Availability: No
ISBN: 3, Copy ID: 3, Location: Reference Section, Availability: Yes

Fines and Actions Taken:

Statement processed.
```

# FUNCTIONS:

## Function to Get Book Details with Author and Genre:

```
CREATE OR REPLACE FUNCTION get_book_details(isbn IN VARCHAR2)
RETURN VARCHAR2 IS
   book_details VARCHAR2(500);
BEGIN
   SELECT 'Title: ' || b.Title || ', Author: ' || a.Name || ', Genre: ' || g.Genre
   INTO book_details
   FROM Book b
   JOIN Author a ON b.Author_ID = a.Author_ID
   JOIN Genre g ON b.Genre_ID = g.Genre_ID
   WHERE b.ISBN = isbn;

   RETURN book_details;
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      RETURN 'Book not found';
   WHEN OTHERS THEN
      RETURN 'Error retrieving book details';
END get_book_details;
```

**Function to Check Member's Borrowing History:**

```
CREATE OR REPLACE FUNCTION get_borrowing_history(member_id IN
NUMBER) RETURN VARCHAR2 IS
   borrowing_history VARCHAR2(1000);
BEGIN
```

```
    SELECT LISTAGG('Title: ' || b.Title || ', Borrow Date: ' || br.Borrow_Date || ', Return Date: ' || br.Return_Date, '; ') WITHIN GROUP (ORDER BY br.Borrow_Date)
    INTO borrowing_history
    FROM Borrowing br
    JOIN Book b ON br.ISBN = b.ISBN
    WHERE br.Member_ID = member_id;

    RETURN NVL(borrowing_history, 'No borrowing history for Member ID ' || member_id);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN 'No borrowing history for Member ID ' || member_id;
    WHEN OTHERS THEN
        RETURN 'Error retrieving borrowing history';
END get_borrowing_history;
```

## Function to Get Member Information:

```
CREATE OR REPLACE FUNCTION get_member_info(member_id IN NUMBER) RETURN VARCHAR2 IS
    member_info VARCHAR2(200);
BEGIN
    SELECT 'Member ID: ' || Member_ID || ', Name: ' || Name || ', Area: ' || Area || ', City: ' || City
    INTO member_info
    FROM Member
    WHERE Member_ID = member_id;

    RETURN member_info;
```

```
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      RETURN 'Member not found';
   WHEN OTHERS THEN
      RETURN 'Error retrieving member information';
END get_member_info;
```

←----------→