School of Information Technology and Engineering (SITE)
Vellore Institute of Technology
Vellore, Tamil Nadu – 632014

ITE1003 – DATABASE MANAGEMENT SYSTEM
FALL SEMESTER 2021-22
SLOT: G2
CLASS ID: VL2021220100612

Library Management System

PROJECT REVIEW

Date: 10.12.2021

Tisha Anil Chordia
(20BIT0212)

Tarunika Agarwal
(20BIT0227)

Akshat Khandelwal
(20BIT0259)

Nimish Jain
(20BIT0270)

Contents

Problem Statement

Library Management System is a database application containing details of all the necessary affairs regarding the day to day functioning of a library. Ensuring a well updated and planned out system results in the highly efficient running of a library and affairs like managing books, authors, members, library staff, issue, return and fine.

Introduction

A library is a collection of organized information and resources which is made accessible to a well-defined community for their borrowing or reference's sake. The manual process of keeping student records, book records, account details, and managing employees is very difficult. There are various problems also faced by the members in the library such as finding any particular book, information whether the book is available or not, for what time this book will be available, searching for books using book id, etc. To eliminate this manual system, a library management system has been developed which will handle all the current issues faced by the members and by its admin personnel in the most efficient way possible.

The library management system is a project which aims at developing a database system to maintain all the daily work of the library. It allows users to store book details, member details, author details, and various other information. It has a facility of admin login through which the admin can monitor the whole system. It has a facility where members after logging in to their accounts can see the list of books issued and its issue date and return date. The system is strong enough to withstand regressive yearly operation under conditions where the database is maintained and cleared over a certain span of time.

Overall, the implementation of this system will considerably reduce data entry time and also provide readily calculated reports.

Functional requirement

The system must only allow a librarian with a valid ID and password to enter the system. The librarian must be able to log out after they finish using the system. The system must be able to not allow two books having the same book id. The system must be able to search if the book is available or not before issuing the books.

The user must be able to see the book details of each book in the database and the number of books issued by the user must be visible. It should display the information on whether the book is available or not and at what time a particular book will be available. The user should be able to see the issuing date of the book and expiry date of the book as well as the author details of the book. The system must display the user details along with member id.

Data Requirements

Librarian

The library will be managed by the Librarian. Each librarian will have a unique librarian id and password. The librarian will be managing the books and the members.

Book

This table contains a complete list of books available in the library. Each book has been provided a unique book_id which serves as a primary key. The book details include book_name, category, rack_no, price, purchase_date. This table contains member_id and author_id as the foreign key.

Member

The members are the ones who would be accessing the library system for issuing and returning books. This table contains details of all members having a unique member_id as the primary key. The member details contain gender, dob, email_id, and phone_no. It also contains name as a composite attribute which consists of first_name and last_name. This table has lib_id as a foreign key.

Author

This table contains the list of authors whose books are present in the library. The authors are uniquely identified with author_id which serves as the primary key. The author details include author_name, gender, dob, and experience.

Issue_Status

Members issue books whose record is maintained in the library. This table keeps the record of all the issue details giving each transaction a issue_date which acts as the primary key when paired with book_id. The issue details include date_of_return and expiry_date. It contains member_id as the foreign key.

Analysis

Entities and Attributes

LIBRARIAN

Strong entity: LIB_ID is used as the primary key.

LIB_ID, ADMIN_PASS

BOOK

Strong entity: BOOK_ID is used as the primary key.

BOOK_ID, RACK_NO, PRICE, PURCHASE_DATE, CATEGORY, BOOK_NAME, MEMBER_ID, AUTHOR_ID

MEMBER

Strong entity: MEMBER_ID is used as the primary key.

MEMBER_ID, GENDER, DOB, EMAIL_ID, PHONE_NO, FIRST_NAME, LAST_NAME, LIB_ID

AUTHOR

Strong entity: AUTHOR_ID is used as the primary key.

AUTHOR_ID, GENDER, AUTHOR_NAME, DOB, EXPERIENCE

ISSUE_STATUS

Weak entity: DATE_OF_ISSUE, BOOK_ID is used as the primary key.

ISSUE_DATE, BOOK_ID, DATE_OF_RETURN, EXPIRY_DATE, MEMBER_ID

Relationships, Cardinality, Participation

MEMBER MANAGED_BY LIBRARIAN (N:1)

Managed_by is the relation between member and librarian. It is a many-to-one relationship as a librarian manages many members, but a member is only managed by one librarian. Each member is managed by at least one librarian, but a librarian can exist without a member.

LIBRARIAN MANAGES BOOK (M:N)

Manages are the relation between book and librarian. It is a many-to-many relationship as 1 librarian manages many books and 1 book is managed by many librarians. There is partial participation from both sides.

AUTHOR WRITES BOOK (1:N)

Writing is the relationship between author and book. It is a one-to-many relationship as one author can write multiple books, but one book is written by only one author. Each and every book (total participation) in the library is written by an author but it is not necessary that every book written by an author is present in the library.

BOOK ISSUED_BY MEMBER (N:1)

Issued_by is the relationship between book and author. It is a many-to-one relationship as a member can issue multiple books, but a book can only be issued by one member at a time. Partial participation from both sides as a book and a member can exist without each other.

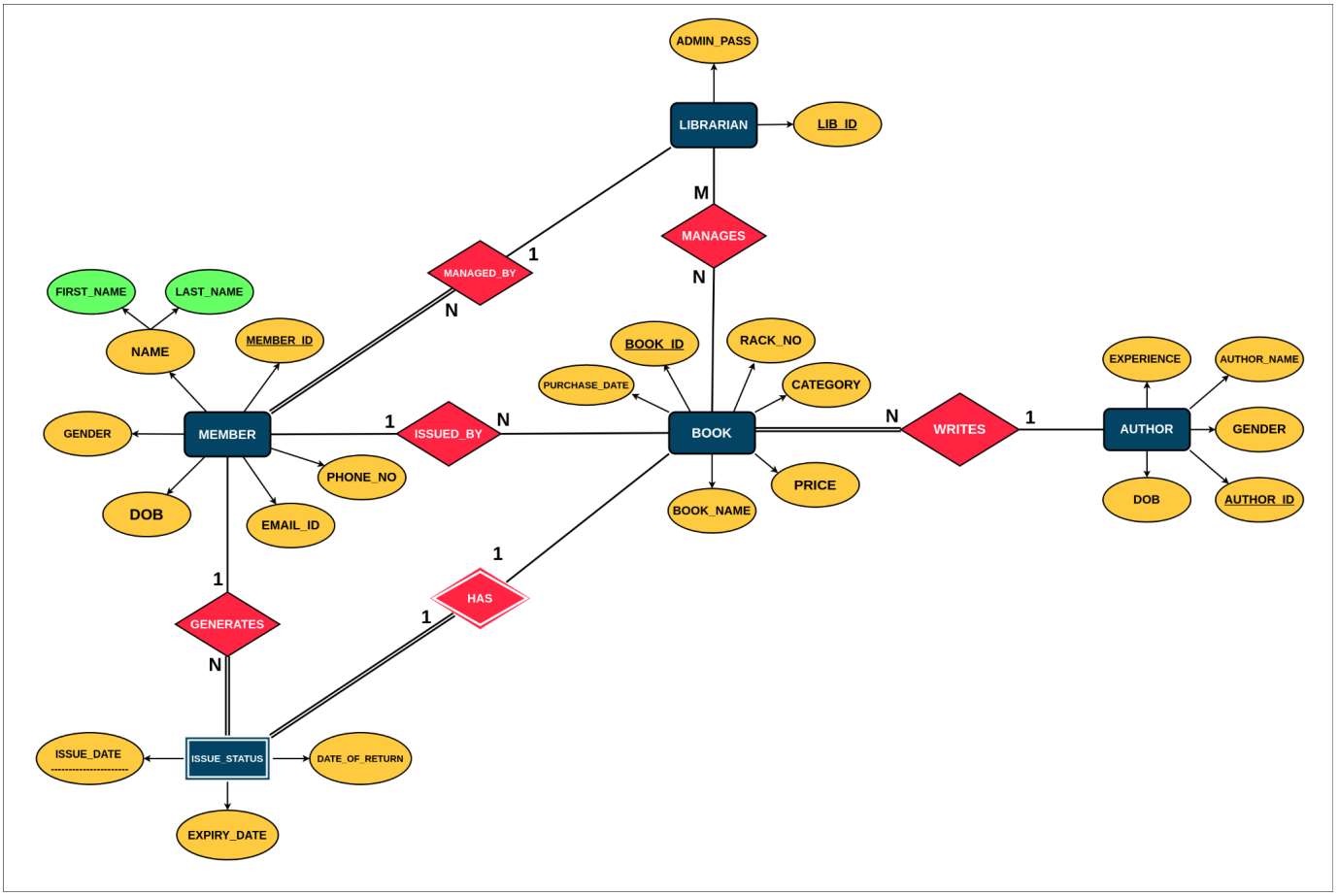
MEMBER GENERATES ISSUE_STATUS (1:N)

Generates is the relationship between member and issue_status. It is a one-to-many relationship as one member can generate many issue status, but an issue status can only be generated by one member. Issue status (total participation) can only exist if a member issues a book, but a member can exist without an issue status.

BOOK HAS ISSUE_STATUS (1:1)

Has is the relationship between issue_status and book. It is a one-to-one relationship as one issue status is linked to only one book and one book has only one issue status at a time. Partial participation from both sides.

Design



Mapping

Step 1: Mapping of Regular Entity Types.

- For each regular (strong) entity type E in the ER diagram, create a relation R that includes all the simple attributes of E.
- Choose one of the key attributes of E as the primary key for R.

If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.

Step 2: Mapping of Weak Entity Types.

- For each weak entity type W in the ER schema with owner entity type E, create a relation R & include all simple attributes (or simple components of composite attributes) of W as attributes of R.

- Also, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s).

The primary key of R is the combination of the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any.

Step 3: Mapping of Binary 1:1 Relation Types.

- For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R.
- There are three possible approaches:
    - **Foreign Key approach:** Choose one of the relations-say S-and include a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S.
    - **Merged relation option:** An alternate mapping of a 1:1 relationship type is possible by merging the two entity types and the relationship into a single relation. This may be appropriate when both participations are total.
    - **Cross-reference or relationship relation option:** The third alternative is to set up a third relation R for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types.

Step 4: Mapping of Binary 1:N Relationship Types.

- For each regular binary 1:N relationship type R, identify the relation S that represents the participating entity type at the N-side of the relationship type.
- Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R.
- Include any simple attributes of the 1:N relation type as attributes of S.

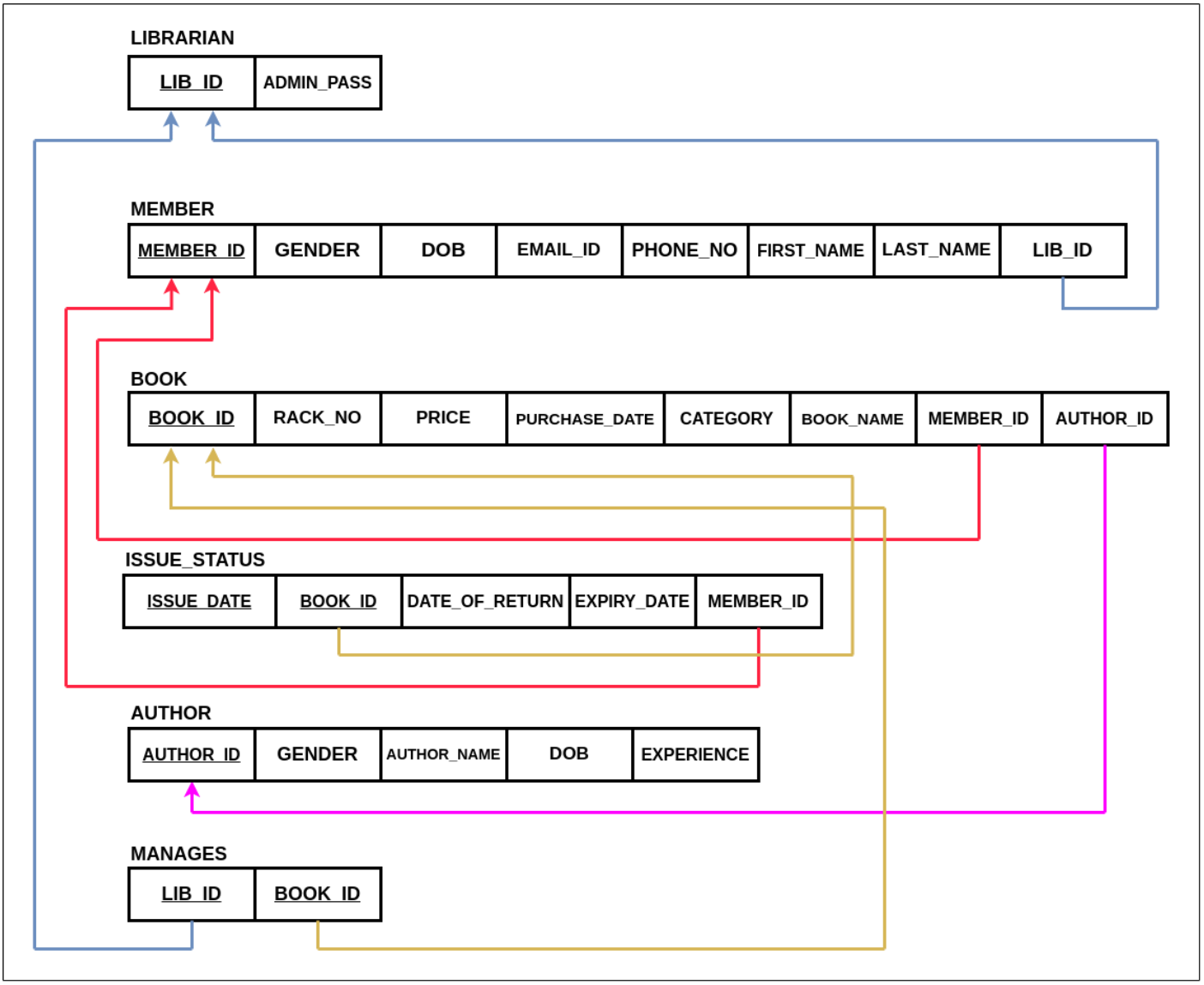
Step 5: Mapping of Binary M:N Relationship Types.

- For each regular binary M:N relationship type R, create a new relation S to represent R.
- Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S.

- Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S.

Step 6: Mapping of Multivalued attributes.

- For each multivalued attribute A, create a new relation R.
- This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type of relationship type that has A as an attribute.
- The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.



Normalization

Normalization is the process of organizing the data in the database. Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization divides the larger table into the smaller table and links them using relationships. The normal form is used to reduce redundancy from the database table.

1NF

A relation is in 1NF if all values stored in a relation are single-valued and atomic i.e. there are no multi-valued attributes.

2NF

A relation is in 2NF if it is in 1NF and every non-key attribute is fully dependent on each candidate key i.e. we don't have any partial functional dependency.

3NF

A relation is in 3NF, if it is in 2NF and there are no transitive functional dependencies.

Applying normalization on each table

- **LIBRARIAN (lib_id, admin_pass)**

  Functional Dependency:

  lib_id → admin pass

  All the attributes are atomic because it is taken care of in the design phase. All the multi-valued attributes are mapped into single-valued attributes during the ER table conversion. There are no partial and transitive functional dependencies. Hence, the relation is already in 1NF, 2NF and 3NF and it cannot be normalized further.

- **MEMBER (member_id, gender, dob, email_id, phone_no, first_name, last_name, lib_id)**

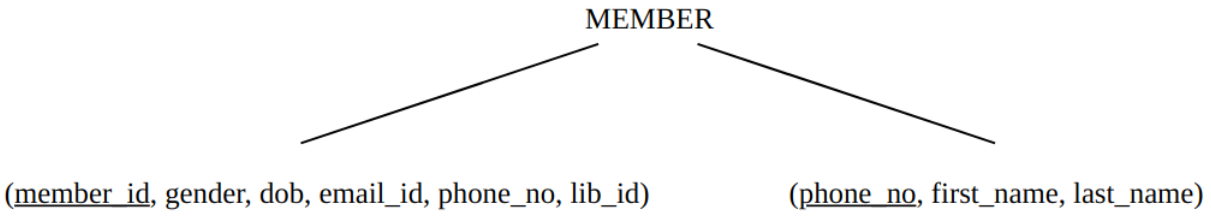
  Functional dependencies:

member_id → gender, dob, email_id, phone_no, first_name, last_name, lib_id

phone_no → first_name, last_name

All the attributes are atomic because it is taken care of in the design phase. All the multi-valued attributes are mapped into single-valued attributes during the ER table conversion. There are no partial functional dependencies. Hence, the relation is in 1NF and 2NF.

Since there is a transitive functional dependency, the relation is not in 3NF. So, we will decompose the relation as



The normalized relation after 3NF are:

- (member_id, gender, dob, email_id, phone_no, lib_id)
- (phone_no, first_name, last_name)

- **BOOK (book_id, rack_no, price, purchase_date, category, book_name, member_id, author_id)**

  Functional Dependency:

  book_id → rack_no, price, purchase_date, category, book_name, member_id, author_id

  All the attributes are atomic because it is taken care of in the design phase. All the multi-valued attributes are mapped into single-valued attributes during the ER table conversion. There are no partial and transitive functional dependencies. Hence, the relation is already in 1NF, 2NF and 3NF and it cannot be normalized further.

- **ISSUE_STATUS (issue_date, book_id, date_of_return, expiry_date, member_id)**

  Functional Dependency:

  (issue_date, book_id) → date_of_return, expiry_date, member_id

All the attributes are atomic because it is taken care of in the design phase. All the multi-valued attributes are mapped into single-valued attributes during the ER table conversion. There are no partial and transitive functional dependencies. Hence, the relation is already in 1NF, 2NF and 3NF and it cannot be normalized further.

- **AUTHOR (author_id, gender, author_name, dob, experience)**

  Functional Dependency:

  author_id → gender, author_name, dob, experience

  All the attributes are atomic because it is taken care of in the design phase. All the multi-valued attributes are mapped into single-valued attributes during the ER table conversion. There are no partial and transitive functional dependencies. Hence, the relation is already in 1NF, 2NF and 3NF and it cannot be normalized further.

The tables after normalization are:

LIBRARIAN

LIB_ID	ADMIN_PASS

MEMBER

MEMBER_ID	GENDER	DOB	EMAIL_ID	PHONE_NO	LIB_ID

DETAILS

PHONE_NO	FIRST_NAME	LAST_NAME

BOOK

BOOK_ID	RACK_NO	PRICE	PURCHASE_DATE	CATEGORY	BOOK_NAME	MEMBER_ID	AUTHOR_ID

ISSUE_STATUS

ISSUE_DATE	BOOK_ID	DATE_OF_RETURN	EXPIRY_DATE	MEMBER_ID

AUTHOR

AUTHOR_ID	GENDER	AUTHOR_NAME	DOB	EXPERIENCE

MANAGES

LIB_ID	BOOK_ID

Implementation

Create table author

```
CREATE TABLE author (
    author_id INT,
    gender VARCHAR(5) NOT NULL,
    author_name VARCHAR(30) NOT NULL,
    dob DATE NOT NULL,
    experience INT,
    CONSTRAINT author_pk PRIMARY KEY (author_id)
);


DESC author;
```

```
 3  CREATE TABLE author (
 4      author_id INT,
 5      gender VARCHAR(5) NOT NULL,
 6      author_name VARCHAR(30) NOT NULL,
 7      dob DATE NOT NULL,
 8      experience INT,
 9      CONSTRAINT author_pk PRIMARY KEY (author_id)
10  );
11
12  DESC author;
```

```
mysql> desc author;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| author_id   | int         | NO   | PRI | NULL    |       |
| gender      | varchar(10) | YES  |     | NULL    |       |
| author_name | varchar(30) | NO   |     | NULL    |       |
| dob         | date        | NO   |     | NULL    |       |
| experience  | int         | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Create table librarian

```
CREATE TABLE librarian (
    lib_id INT,
    admin_pass VARCHAR(20) NOT NULL,
    CONSTRAINT librarian_pk PRIMARY KEY (lib_id)
);

DESC librarian;
```

```
14  CREATE TABLE librarian (
15      lib_id INT,
16      admin_pass VARCHAR(20) NOT NULL,
17      CONSTRAINT librarian_pk PRIMARY KEY (lib_id)
18  );
19
20  DESC librarian;
```

```
mysql> desc librarian;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| lib_id     | int         | NO   | PRI | NULL    |       |
| admin_pass | varchar(20) | NO   |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Create table details

```
CREATE TABLE details (
    phone_no INT,
    fname VARCHAR(20) NOT NULL,
    lname VARCHAR(20) NOT NULL,
    CONSTRAINT details_pk PRIMARY KEY (phone_no)
);


DESC details;
```

```
22  CREATE TABLE details (
23      phone_no INT,
24      fname VARCHAR(20) NOT NULL,
25      lname VARCHAR(20) NOT NULL,
26      CONSTRAINT details_pk PRIMARY KEY (phone_no)
27  );
28
29  DESC details;
```

```
mysql> desc details;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| phone_no | int         | NO   | PRI | NULL    |       |
| fname    | varchar(20) | NO   |     | NULL    |       |
| lname    | varchar(20) | NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Create table member

```
CREATE TABLE member (
    member_id INT,
    gender VARCHAR(10) NOT NULL,
    dob DATE NOT NULL,
    email_id VARCHAR(20) NOT NULL,
    phone_no INT NOT NULL,
    lib_id INT NOT NULL,
    CONSTRAINT member_pk PRIMARY KEY (member_id),
    CONSTRAINT member_fk FOREIGN KEY (lib_id)
        REFERENCES librarian (lib_id),
    CONSTRAINT member_fk1 FOREIGN KEY (phone_no)
        REFERENCES details (phone_no)
);

DESC member;
```

```sql
CREATE TABLE member (
    member_id INT,
    gender VARCHAR(10) NOT NULL,
    dob DATE NOT NULL,
    email_id VARCHAR(20) NOT NULL,
    phone_no INT NOT NULL,
    lib_id INT NOT NULL,
    CONSTRAINT member_pk PRIMARY KEY (member_id),
    CONSTRAINT member_fk FOREIGN KEY (lib_id)
        REFERENCES librarian (lib_id),
    CONSTRAINT member_fk1 FOREIGN KEY (phone_no)
        REFERENCES details (phone_no)
);

DESC member;
```

```
mysql> desc member;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| member_id | int         | NO   | PRI | NULL    |       |
| gender    | varchar(10) | NO   |     | NULL    |       |
| dob       | date        | NO   |     | NULL    |       |
| email_id  | varchar(20) | NO   |     | NULL    |       |
| phone_no  | int         | NO   | MUL | NULL    |       |
| lib_id    | int         | NO   | MUL | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Create table book

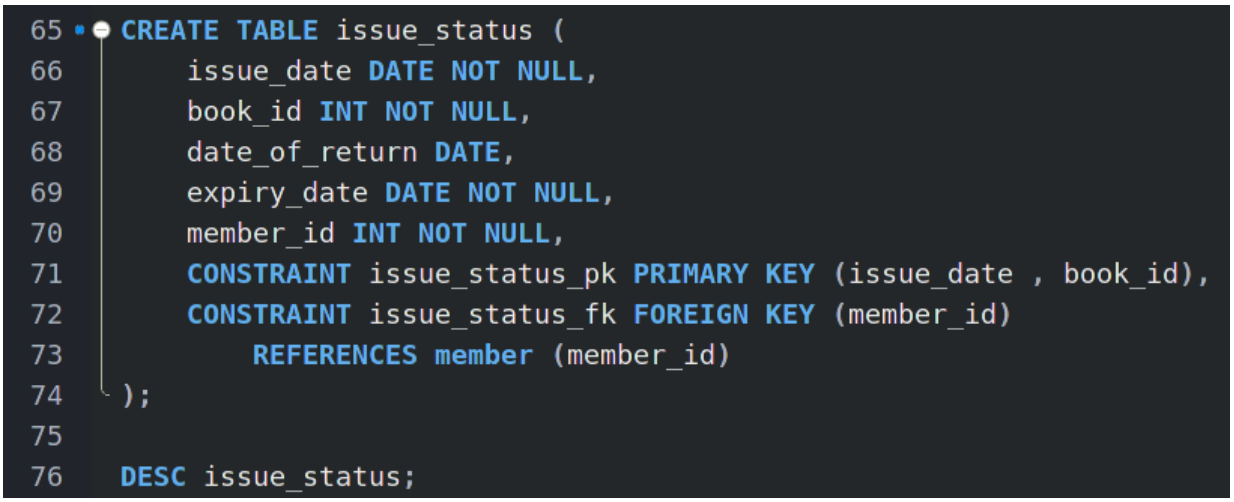
```
CREATE TABLE book (
    book_id INT,
    rack_no INT NOT NULL,
    price NUMERIC(7 , 2 ),
    purchase_date DATE NOT NULL,
    category VARCHAR(20) NOT NULL,
    book_name VARCHAR(50) NOT NULL,
    member_id INT NOT NULL,
    author_id INT NOT NULL,
    CONSTRAINT book_pk PRIMARY KEY (book_id),
    CONSTRAINT book_fk FOREIGN KEY (member_id)
        REFERENCES member (member_id),
    CONSTRAINT book_fk1 FOREIGN KEY (author_id)
        REFERENCES author (author_id)
);

DESC book;
```

```sql
CREATE TABLE book (
    book_id INT,
    rack_no INT NOT NULL,
    price NUMERIC(7 , 2 ),
    purchase_date DATE NOT NULL,
    category VARCHAR(20) NOT NULL,
    book_name VARCHAR(50) NOT NULL,
    member_id INT NOT NULL,
    author_id INT NOT NULL,
    CONSTRAINT book_pk PRIMARY KEY (book_id),
    CONSTRAINT book_fk FOREIGN KEY (member_id)
        REFERENCES member (member_id),
    CONSTRAINT book_fk1 FOREIGN KEY (author_id)
        REFERENCES author (author_id)
);

DESC book;
```

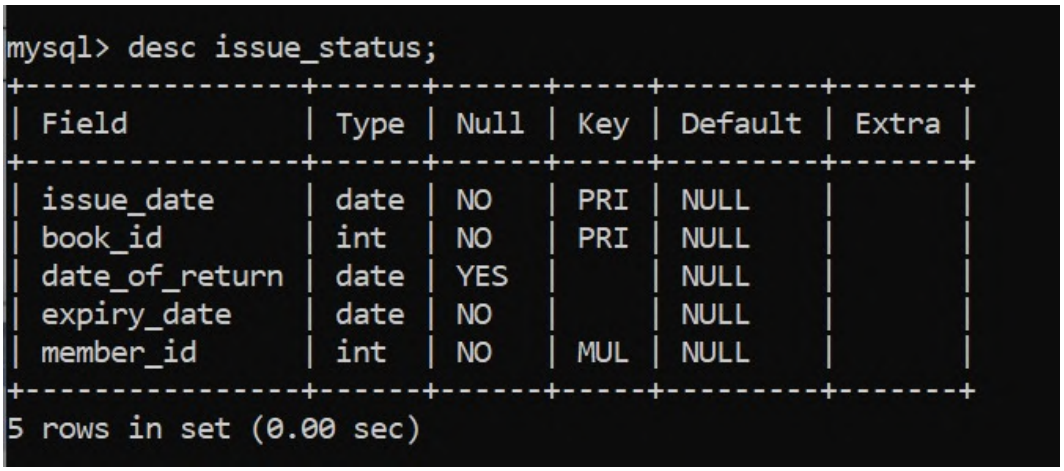
```
mysql> desc book;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| book_id       | int         | NO   | PRI | NULL    |       |
| rack_no       | int         | NO   |     | NULL    |       |
| price         | decimal(7,2)| YES  |     | NULL    |       |
| purchase_date | date        | NO   |     | NULL    |       |
| category      | varchar(20) | NO   |     | NULL    |       |
| book_name     | varchar(50) | NO   |     | NULL    |       |
| member_id     | int         | NO   | MUL | NULL    |       |
| author_id     | int         | NO   | MUL | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)
```

Create table issue_status

```
CREATE TABLE issue_status (
    issue_date DATE NOT NULL,
    book_id INT NOT NULL,
    date_of_return DATE,
    expiry_date DATE NOT NULL,
    member_id INT NOT NULL,
    CONSTRAINT issue_status_pk PRIMARY KEY (issue_date , book_id),
    CONSTRAINT issue_status_fk FOREIGN KEY (member_id)
        REFERENCES member (member_id)
);

DESC issue_status;
```



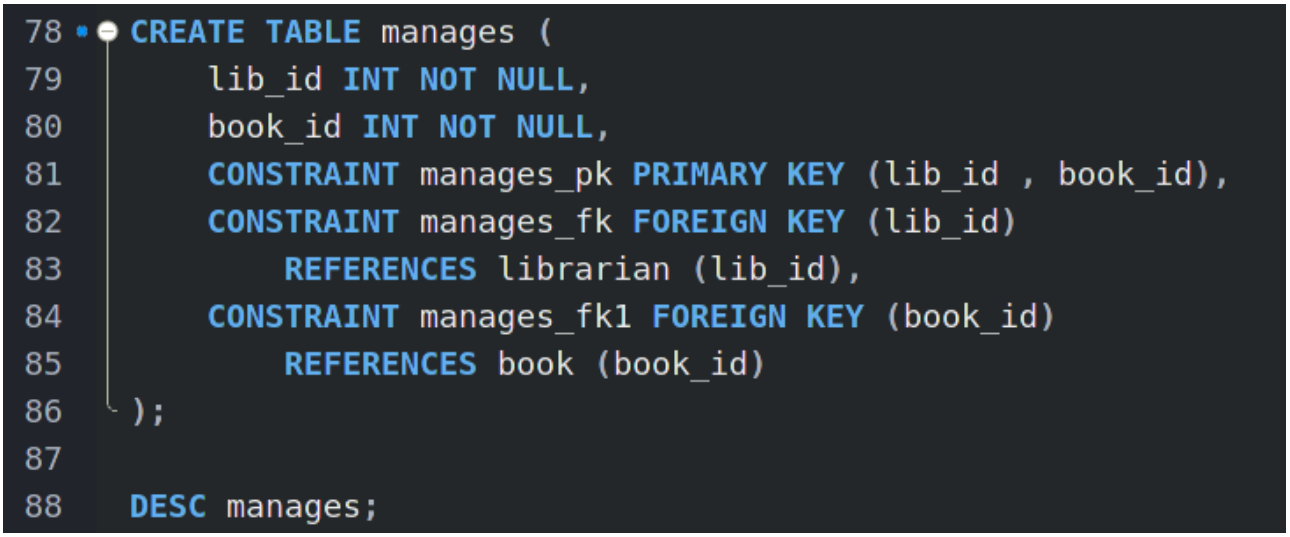
```
65  CREATE TABLE issue_status (
66      issue_date DATE NOT NULL,
67      book_id INT NOT NULL,
68      date_of_return DATE,
69      expiry_date DATE NOT NULL,
70      member_id INT NOT NULL,
71      CONSTRAINT issue_status_pk PRIMARY KEY (issue_date , book_id),
72      CONSTRAINT issue_status_fk FOREIGN KEY (member_id)
73          REFERENCES member (member_id)
74  );
75
76  DESC issue_status;
```



```
mysql> desc issue_status;
+---------------+------+------+-----+---------+-------+
| Field         | Type | Null | Key | Default | Extra |
+---------------+------+------+-----+---------+-------+
| issue_date    | date | NO   | PRI | NULL    |       |
| book_id       | int  | NO   | PRI | NULL    |       |
| date_of_return| date | YES  |     | NULL    |       |
| expiry_date   | date | NO   |     | NULL    |       |
| member_id     | int  | NO   | MUL | NULL    |       |
+---------------+------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Create table manages

```
CREATE TABLE manages (
    lib_id INT NOT NULL,
    book_id INT NOT NULL,
    CONSTRAINT manages_pk PRIMARY KEY (lib_id , book_id),
    CONSTRAINT manages_fk FOREIGN KEY (lib_id)
        REFERENCES librarian (lib_id),
    CONSTRAINT manages_fk1 FOREIGN KEY (book_id)
        REFERENCES book (book_id)
);

DESC manages;
```




```
mysql> desc manages;
+---------+------+------+-----+---------+-------+
| Field   | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| lib_id  | int  | NO   | PRI | NULL    |       |
| book_id | int  | NO   | PRI | NULL    |       |
+---------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Insert into table book

```
insert into book values(5001,08,5055,'2020-07-23','Memoir','The Story of My Life',59,1001);

insert into book values(5010,02,4655,'1990-12-29','Fiction','The Hunger Games',10,1002);

insert into book values(5050,06,6700,'1999-11-11',"Childern's Novel",'Geronimo Stilton',25,1003);

insert into book values(5012,01,7000,'2001-09-04','Fiction','The Great Train Journey',98,1004);

insert into book values(5017,06,9000,'2005-04-30','Non-Fiction','Becoming',10,1005);

select * from book;
```

```
32 • insert into book values(5001,08,5055,'2020-07-23','Memoir','The Story of My Life',59,1001);
33 • insert into book values(5010,02,4655,'1990-12-29','Fiction','The Hunger Games',10,1002);
34 • insert into book values(5050,06,6700,'1999-11-11',"Childern's Novel",'Geronimo Stilton',25,1003);
35 • insert into book values(5012,01,7000,'2001-09-04','Fiction','The Great Train Journey',98,1004);
36 • insert into book values(5017,06,9000,'2005-04-30','Non-Fiction','Becoming',10,1005);
37 • select * from book;
```

```
mysql> select * from book;
+---------+---------+---------+---------------+-----------------+-------------------------+-----------+-----------+
| book_id | rack_no | price   | purchase_date | category        | book_name               | member_id | author_id |
+---------+---------+---------+---------------+-----------------+-------------------------+-----------+-----------+
|    5001 |       8 | 5055.00 | 2020-07-23    | Memoir          | The Story of My Life    |        59 |      1001 |
|    5010 |       2 | 4655.00 | 1990-12-29    | Fiction         | The Hunger Games        |        10 |      1002 |
|    5012 |       1 | 7000.00 | 2001-09-04    | Fiction         | The Great Train Journey |        98 |      1004 |
|    5017 |       6 | 9000.00 | 2005-04-30    | Non-Fiction     | Becoming                |        10 |      1005 |
|    5050 |       6 | 6700.00 | 1999-11-11    | Childern's Novel | Geronimo Stilton        |        25 |      1003 |
+---------+---------+---------+---------------+-----------------+-------------------------+-----------+-----------+
5 rows in set (0.01 sec)
```

Insert into table member

```
insert into member values
(99,'male','1999-05-05','ramkumar@gmail.com',900897885,101);

insert into member values
(98,'male','1999-08-05','shyam1999@gmail.com',800897444,102);

insert into member values
(59,'male','2002-03-09','rajuuu@gmail.com',700897344,103);

insert into member values
(25,'female','2000-04-10','raniraja@gmail.com',600897885,101);

insert into member values
(10,'female','1990-01-11','siddhi101@gmail.com',880089785,104);

select * from member;
```

```
18 • insert into member values (99,'male','1999-05-05','ramkumar@gmail.com',900897885,101);
19 • insert into member values (98,'male','1999-08-05','shyam1999@gmail.com',800897444,102);
20 • insert into member values (59,'male','2002-03-09','rajuuu@gmail.com',700897344,103);
21 • insert into member values (25,'female','2000-04-10','raniraja@gmail.com',600897885,101);
22 • insert into member values (10,'female','1990-01-11','siddhi101@gmail.com',880089785,104);
23 • select * from member;
```

```
mysql> select * from member;
+-----------+--------+------------+---------------------+-----------+--------+
| member_id | gender | dob        | email_id            | phone_no  | lib_id |
+-----------+--------+------------+---------------------+-----------+--------+
|        10 | female | 1990-01-11 | siddhi101@gmail.com | 880089785 |    104 |
|        25 | female | 2000-04-10 | raniraja@gmail.com  | 600897885 |    101 |
|        59 | male   | 2002-03-09 | rajuuu@gmail.com    | 700897344 |    103 |
|        98 | male   | 1999-08-05 | shyam1999@gmail.com | 800897444 |    102 |
|        99 | male   | 1999-05-05 | ramkumar@gmail.com  | 900897885 |    101 |
+-----------+--------+------------+---------------------+-----------+--------+
5 rows in set (0.00 sec)
```

Insert into details

```
insert into details values (900897885,'Ram','Kumar');
insert into details values (800897444,'Shyam','Agarwal');
insert into details values (700897344,'Raj','Gupta');
insert into details values (600897885,'Rani','Jain');
insert into details values (880089785,'Siddhi','Kumari');
select * from details;
```

```
11 •  insert into details values (900897885,'Ram','Kumar');
12 •  insert into details values (800897444,'Shyam','Agarwal');
13 •  insert into details values (700897344,'Raj','Gupta');
14 •  insert into details values (600897885,'Rani','Jain');
15 •  insert into details values (880089785,'Siddhi','Kumari');
16 •  select * from details;
```

```
mysql> select * from details;
+-----------+--------+---------+
| phone_no  | fname  | lname   |
+-----------+--------+---------+
| 600897885 | Rani   | Jain    |
| 700897344 | Raj    | Gupta   |
| 800897444 | Shyam  | Agarwal |
| 880089785 | Siddhi | Kumari  |
| 900897885 | Ram    | Kumar   |
+-----------+--------+---------+
5 rows in set (0.00 sec)
```

Insert into issue_status

```
insert into issue_status values(
'2021-09-23',5001,'2021-09-30','2021-10-23',98);

insert into issue_status values(
'2021-06-13',5050,'2021-06-28','2021-07-13',10);

insert into issue_status values(
'2021-11-03',5012,Null,'2021-12-03',25);

insert into issue_status values(
'2021-03-06',5017,'2021-03-24','2021-04-05',99);

insert into issue_status values(
'2021-11-15',5010,Null,'2021-12-15',59);

select * from issue_status;
```

```
46    insert into issue_status values( '2021-09-23',5001,'2021-09-30','2021-10-23',98);
47    insert into issue_status values( '2021-06-13',5050,'2021-06-28','2021-07-13',10);
48    insert into issue_status values( '2021-11-03',5012,Null,'2021-12-03',25);
49    insert into issue_status values( '2021-03-06',5017,'2021-03-24','2021-04-05',99);
50 •  insert into issue_status values( '2021-11-15',5010,Null,'2021-12-15',59);
51 •  select * from issue_status;
```

```
mysql> select * from issue_status;
+------------+---------+----------------+-------------+-----------+
| issue_date | book_id | date_of_return | expiry_date | member_id |
+------------+---------+----------------+-------------+-----------+
| 2021-03-06 |    5017 | 2021-03-24     | 2021-04-05  |        99 |
| 2021-06-13 |    5050 | 2021-06-28     | 2021-07-13  |        10 |
| 2021-09-23 |    5001 | 2021-09-30     | 2021-10-23  |        98 |
| 2021-11-03 |    5012 | NULL           | 2021-12-03  |        25 |
| 2021-11-15 |    5010 | NULL           | 2021-12-15  |        59 |
+------------+---------+----------------+-------------+-----------+
5 rows in set (0.00 sec)
```

Insert into manages

```
insert into manages values(103,5001);
insert into manages values(104,5010);
insert into manages values(101,5050);
insert into manages values(102,5012);
insert into manages values(104,5017);
select * from manages;
```

```
39   insert into manages values(103,5001);
40   insert into manages values(104,5010);
41   insert into manages values(101,5050);
42   insert into manages values(102,5012);
43   insert into manages values(104,5017);
44   select * from manages;
```

```
mysql> select * from manages;
+--------+---------+
| lib_id | book_id |
+--------+---------+
|    103 |    5001 |
|    104 |    5010 |
|    102 |    5012 |
|    104 |    5017 |
|    101 |    5050 |
+--------+---------+
5 rows in set (0.01 sec)
```

Insert into author

```
insert into author values (1001,'female','Helen
Keller','1880-06-27',40);

insert into author values (1002,'female','Suzanne
Collins','1962-08-10',20);

insert into author values (1003,'female','Elisabetta
Dami','1958-02-12',22);

insert into author values (1004,'male','Ruskin Bond','1934-05-19',46);

insert into author values (1005,'female','Michelle
Obama','1964-01-17',21);

select * from author;
```

```
25   insert into author values (1001,'female','Helen Keller','1880-06-27',40);
26   insert into author values (1002,'female','Suzanne Collins','1962-08-10',20);
27   insert into author values (1003,'female','Elisabetta Dami','1958-02-12',22);
28   insert into author values (1004,'male','Ruskin Bond','1934-05-19',46);
29   insert into author values (1005,'female','Michelle Obama','1964-01-17',21);
30   select * from author;
```

```
mysql> select * from author;
+-----------+--------+-----------------+------------+------------+
| author_id | gender | author_name     | dob        | experience |
+-----------+--------+-----------------+------------+------------+
|      1001 | female | Helen Keller    | 1880-06-27 |         40 |
|      1002 | female | Suzanne Collins | 1962-08-10 |         20 |
|      1003 | female | Elisabetta Dami | 1958-02-12 |         22 |
|      1004 | male   | Ruskin Bond     | 1934-05-19 |         46 |
|      1005 | female | Michelle Obama  | 1964-01-17 |         21 |
+-----------+--------+-----------------+------------+------------+
5 rows in set (0.00 sec)
```

Insert into librarian

```
insert into librarian values (101,'admin101');
insert into librarian values (102,'admin102');
insert into librarian values (103,'admin103');
insert into librarian values (104,'admin104');
insert into librarian values (105,'admin105');
select * from librarian;
```

```
 6   insert into librarian values (101,'admin101');
 7   insert into librarian values (102,'admin102');
 8   insert into librarian values (103,'admin103');
 9   insert into librarian values (104,'admin104');
10 • insert into librarian values (105,'admin105');
11 • select * from librarian;
```

```
mysql> select * from librarian;
+--------+------------+
| lib_id | admin_pass |
+--------+------------+
|    101 | admin101   |
|    102 | admin102   |
|    103 | admin103   |
|    104 | admin104   |
|    105 | admin105   |
+--------+------------+
5 rows in set (0.01 sec)
```

Alter operations on tables

```
alter table author modify gender varchar(10);

DESC author;
```

```
4 •  alter table author modify gender varchar(10);
5 •  DESC author;
```

```
mysql> desc author;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| author_id   | int(11)     | NO   | PRI | NULL    |       |
| gender      | varchar(10) | YES  |     | NULL    |       |
| author_name | varchar(30) | NO   |     | NULL    |       |
| dob         | date        | NO   |     | NULL    |       |
| experience  | int(11)     | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.05 sec)
```

```
alter table book add constraint book_chk check (price>=0);

DESC book;
```

```
7 •  alter table book add constraint book_chk check (price>=0);
8 •  DESC book;
```

```
mysql> desc book;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| book_id       | int(11)      | NO   | PRI | NULL    |       |
| rack_no       | int(11)      | NO   |     | NULL    |       |
| price         | decimal(7,2) | YES  |     | NULL    |       |
| purchase_date | date         | NO   |     | NULL    |       |
| category      | varchar(20)  | NO   |     | NULL    |       |
| book_name     | varchar(50)  | NO   |     | NULL    |       |
| member_id     | int(11)      | NO   | MUL | NULL    |       |
| author_id     | int(11)      | NO   | MUL | NULL    |       |
+---------------+--------------+------+-----+---------+-------+
8 rows in set (0.03 sec)
```

Delete operation on tables

```
insert into issue_status values(
'2021-10-12',5001,'2021-10-25','2021-11-11',10);

select * from issue_status;

delete from issue_status where issue_date = '2021-10-12';
```

```
56    insert into issue_status values( '2021-10-12',5001,'2021-10-25','2021-11-11',10);
57 •  select * from issue_status;
58 •  delete from issue_status where issue_date = '2021-10-12';
```

```
mysql> insert into issue_status values( '2021-10-12',5001,'2021-10-25','2021-11-11',10);
Query OK, 1 row affected (0.02 sec)

mysql> select * from issue_status;
+------------+---------+----------------+-------------+-----------+
| issue_date | book_id | date_of_return | expiry_date | member_id |
+------------+---------+----------------+-------------+-----------+
| 2021-03-06 |    5017 | 2021-03-24     | 2021-04-05  |        99 |
| 2021-06-13 |    5050 | 2021-06-28     | 2021-07-13  |        10 |
| 2021-09-23 |    5001 | 2021-09-30     | 2021-10-23  |        98 |
| 2021-10-12 |    5001 | 2021-10-25     | 2021-11-11  |        10 |
| 2021-11-03 |    5012 | NULL           | 2021-12-03  |        25 |
| 2021-11-15 |    5010 | NULL           | 2021-12-15  |        59 |
+------------+---------+----------------+-------------+-----------+
6 rows in set (0.00 sec)
```

```
mysql> delete from issue_status where issue_date = '2021-10-12';
Query OK, 1 row affected (0.01 sec)

mysql> select * from issue_status;
+------------+---------+----------------+-------------+-----------+
| issue_date | book_id | date_of_return | expiry_date | member_id |
+------------+---------+----------------+-------------+-----------+
| 2021-03-06 |    5017 | 2021-03-24     | 2021-04-05  |        99 |
| 2021-06-13 |    5050 | 2021-06-28     | 2021-07-13  |        10 |
| 2021-09-23 |    5001 | 2021-09-30     | 2021-10-23  |        98 |
| 2021-11-03 |    5012 | NULL           | 2021-12-03  |        25 |
| 2021-11-15 |    5010 | NULL           | 2021-12-15  |        59 |
+------------+---------+----------------+-------------+-----------+
5 rows in set (0.00 sec)
```

Update operation

```
update member set email_id = "rani.501@gmail.com" where member_id=25;

update book set rack_no = 03 where book_id=5010;
```

```
61 •  update member set email_id = "rani.501@gmail.com" where member_id=25;
62 •  update book set rack_no = 03 where book_id=5010;
```

```
mysql> update member set email_id = "rani.501@gmail.com" where member_id=25;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from member;
+-----------+--------+------------+---------------------+-----------+--------+
| member_id | gender | dob        | email_id            | phone_no  | lib_id |
+-----------+--------+------------+---------------------+-----------+--------+
|        10 | female | 1990-01-11 | siddhi101@gmail.com | 880089785 |    104 |
|        25 | female | 2000-04-10 | rani.501@gmail.com  | 600897885 |    101 |
|        59 | male   | 2002-03-09 | rajuuu@gmail.com    | 700897344 |    103 |
|        98 | male   | 1999-08-05 | shyam1999@gmail.com | 800897444 |    102 |
|        99 | male   | 1999-05-05 | ramkumar@gmail.com  | 900897885 |    101 |
+-----------+--------+------------+---------------------+-----------+--------+
5 rows in set (0.00 sec)
```

```
mysql> update book set rack_no = 03 where book_id=5010;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from book;
+---------+---------+---------+--------------+-----------------+-------------------------+-----------+-----------+
| book_id | rack_no | price   | purchase_date | category        | book_name               | member_id | author_id |
+---------+---------+---------+--------------+-----------------+-------------------------+-----------+-----------+
|    5001 |       8 | 5055.00 | 2020-07-23    | Memoir          | The Story of My Life    |        59 |      1001 |
|    5010 |       3 | 4655.00 | 1990-12-29    | Fiction         | The Hunger Games        |        10 |      1002 |
|    5012 |       1 | 7000.00 | 2001-09-04    | Fiction         | The Great Train Journey |        98 |      1004 |
|    5017 |       6 | 9000.00 | 2005-04-30    | Non-Fiction     | Becoming                |        10 |      1005 |
|    5050 |       6 | 6700.00 | 1999-11-11    | Childern's Novel | Geronimo Stilton        |        25 |      1003 |
+---------+---------+---------+--------------+-----------------+-------------------------+-----------+-----------+
5 rows in set (0.00 sec)
```

Select operations with where clause

```
select * from book where price>6000;
select author_name from author where experience=20;
select * from member where gender='female';
```

```
14 •  select * from book where price>6000;
15 •  select author_name from author where experience=20;
16 •  select * from member where gender='female';
```

```
mysql> select * from book where price>6000;
+---------+---------+---------+--------------+-----------------+-----------------------+-----------+-----------+
| book_id | rack_no | price   | purchase_date | category        | book_name             | member_id | author_id |
+---------+---------+---------+--------------+-----------------+-----------------------+-----------+-----------+
|    5012 |       1 | 7000.00 | 2001-09-04    | Fiction         | The Great Train Journey |        98 |      1004 |
|    5017 |       6 | 9000.00 | 2005-04-30    | Non-Fiction     | Becoming              |        10 |      1005 |
|    5050 |       6 | 6700.00 | 1999-11-11    | Childern's Novel | Geronimo Stilton      |        25 |      1003 |
+---------+---------+---------+--------------+-----------------+-----------------------+-----------+-----------+
3 rows in set (0.00 sec)
```

```
mysql> select author_name from author where experience=20;
+-----------------+
| author_name     |
+-----------------+
| Suzanne Collins |
+-----------------+
1 row in set (0.00 sec)
```

```
mysql> select * from member where gender='female';
+-----------+--------+------------+---------------------+-----------+--------+
| member_id | gender | dob        | email_id            | phone_no  | lib_id |
+-----------+--------+------------+---------------------+-----------+--------+
|        10 | female | 1990-01-11 | siddhi101@gmail.com | 880089785 |    104 |
|        25 | female | 2000-04-10 | rani.501@gmail.com  | 600897885 |    101 |
+-----------+--------+------------+---------------------+-----------+--------+
2 rows in set (0.00 sec)
```

Order by clause

```
select book_name,price,category,rack_no from book order by rack_no;
select * from book where price>5000 order by member_id;
select * from issue_status order by member_id;
```

```
19 •  select book_name,price,category,rack_no from book order by rack_no;
20 •  select * from book where price>5000 order by member_id;
21 •  select * from issue_status order by member_id;
```

```
mysql> select book_name,price,category,rack_no from book order by rack_no;
+-------------------------+---------+------------------+---------+
| book_name               | price   | category         | rack_no |
+-------------------------+---------+------------------+---------+
| The Great Train Journey | 7000.00 | Fiction          |       1 |
| The Hunger Games        | 4655.00 | Fiction          |       3 |
| Becoming                | 9000.00 | Non-Fiction      |       6 |
| Geronimo Stilton        | 6700.00 | Childern's Novel |       6 |
| The Story of My Life    | 5055.00 | Memoir           |       8 |
+-------------------------+---------+------------------+---------+
5 rows in set (0.00 sec)
```

```
mysql> select * from book where price>5000 order by member_id;
+---------+---------+---------+---------------+------------------+-----------------------+-----------+-----------+
| book_id | rack_no | price   | purchase_date | category         | book_name             | member_id | author_id |
+---------+---------+---------+---------------+------------------+-----------------------+-----------+-----------+
|    5017 |       6 | 9000.00 | 2005-04-30    | Non-Fiction      | Becoming              |        10 |      1005 |
|    5050 |       6 | 6700.00 | 1999-11-11    | Childern's Novel | Geronimo Stilton      |        25 |      1003 |
|    5001 |       8 | 5055.00 | 2020-07-23    | Memoir           | The Story of My Life  |        59 |      1001 |
|    5012 |       1 | 7000.00 | 2001-09-04    | Fiction          | The Great Train Journey |        98 |      1004 |
+---------+---------+---------+---------------+------------------+-----------------------+-----------+-----------+
4 rows in set (0.00 sec)
```

```
mysql> select * from issue_status order by member_id;
+------------+---------+----------------+-------------+-----------+
| issue_date | book_id | date_of_return | expiry_date | member_id |
+------------+---------+----------------+-------------+-----------+
| 2021-06-13 |    5050 | 2021-06-28     | 2021-07-13  |        10 |
| 2021-11-03 |    5012 | NULL           | 2021-12-03  |        25 |
| 2021-11-15 |    5010 | NULL           | 2021-12-15  |        59 |
| 2021-09-23 |    5001 | 2021-09-30     | 2021-10-23  |        98 |
| 2021-03-06 |    5017 | 2021-03-24     | 2021-04-05  |        99 |
+------------+---------+----------------+-------------+-----------+
5 rows in set (0.00 sec)
```

Like clause

```
select * from details where fname like '%i';
select * from author where author_name like 'Ruskin%';
select * from book where category like 'F%';
```

```
24 •  select * from details where fname like '%i';
25 •  select * from author where author_name like 'Ruskin%';
26 •  select * from book where category like 'F%';
```

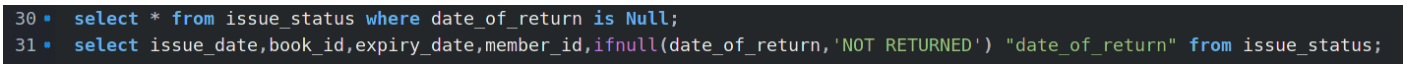
```
mysql> select * from details where fname like '%i';
+-----------+--------+--------+
| phone_no  | fname  | lname  |
+-----------+--------+--------+
| 600897885 | Rani   | Jain   |
| 880089785 | Siddhi | Kumari |
+-----------+--------+--------+
2 rows in set (0.00 sec)
```

```
mysql> select * from author where author_name like 'Ruskin%';
+-----------+--------+-------------+------------+------------+
| author_id | gender | author_name | dob        | experience |
+-----------+--------+-------------+------------+------------+
|      1004 | male   | Ruskin Bond | 1934-05-19 |         46 |
+-----------+--------+-------------+------------+------------+
1 row in set (0.00 sec)
```

```
mysql> select * from book where category like 'F%';
+---------+---------+---------+---------------+----------+-----------------------+-----------+-----------+
| book_id | rack_no | price   | purchase_date | category | book_name             | member_id | author_id |
+---------+---------+---------+---------------+----------+-----------------------+-----------+-----------+
|    5010 |       3 | 4655.00 | 1990-12-29    | Fiction  | The Hunger Games      |        10 |      1002 |
|    5012 |       1 | 7000.00 | 2001-09-04    | Fiction  | The Great Train Journey |        98 |      1004 |
+---------+---------+---------+---------------+----------+-----------------------+-----------+-----------+
2 rows in set (0.00 sec)
```

Is null / Is not null operations

```
select * from issue_status where date_of_return is Null;

select
issue_date,book_id,expiry_date,member_id,ifnull(date_of_return,'NOT
RETURNED') "date_of_return" from issue_status;
```




```
mysql> select * from issue_status where date_of_return is Null;
+------------+---------+----------------+-------------+-----------+
| issue_date | book_id | date_of_return | expiry_date | member_id |
+------------+---------+----------------+-------------+-----------+
| 2021-11-03 |    5012 | NULL           | 2021-12-03  |        25 |
| 2021-11-15 |    5010 | NULL           | 2021-12-15  |        59 |
+------------+---------+----------------+-------------+-----------+
2 rows in set (0.00 sec)
```

```
mysql> select issue_date,book_id,expiry_date,member_id,ifnull(date_of_return,'NOT RETURNED') "date_of_return" from issue_status;
+------------+---------+------------+-----------+----------------+
| issue_date | book_id | expiry_date | member_id | date_of_return |
+------------+---------+------------+-----------+----------------+
| 2021-03-06 |    5017 | 2021-04-05 |        99 | 2021-03-24     |
| 2021-06-13 |    5050 | 2021-07-13 |        10 | 2021-06-28     |
| 2021-09-23 |    5001 | 2021-10-23 |        98 | 2021-09-30     |
| 2021-11-03 |    5012 | 2021-12-03 |        25 | NOT RETURNED   |
| 2021-11-15 |    5010 | 2021-12-15 |        59 | NOT RETURNED   |
+------------+---------+------------+-----------+----------------+
5 rows in set (0.00 sec)
```

Aggregate functions

```
select avg(price) from book;

select min(price) from book;

select gender, count(lib_id) from member group by gender;

select category,max(price) from book group by category;

select count(date_of_return) from issue_status where date_of_return is
not null;
```

```
1 •  select avg(price) from book;
2 •  select min(price) from book;
3 •  select gender, count(lib_id) from member group by gender;
4 •  select category,max(price) from book group by category;
5 •  select count(date_of_return) from issue_status where date_of_return is not null;
```

```
mysql> select avg(price) from book;
+--------------+
| avg(price)   |
+--------------+
| 6482.000000 |
+--------------+
1 row in set (0.04 sec)
```

```
mysql> select min(price) from book;
+------------+
| min(price) |
+------------+
|    4655.00 |
+------------+
1 row in set (0.00 sec)
```

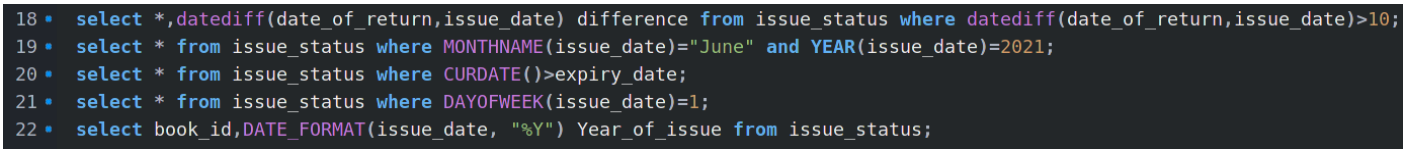
```
mysql> select gender, count(lib_id) from member group by gender;
+--------+--------------+
| gender | count(lib_id) |
+--------+--------------+
| female |            2 |
| male   |            3 |
+--------+--------------+
2 rows in set (0.00 sec)
```

```
mysql> select category,max(price) from book group by category;
+------------------+------------+
| category         | max(price) |
+------------------+------------+
| Memoir           |    5055.00 |
| Fiction          |    7000.00 |
| Non-Fiction      |    9000.00 |
| Childern's Novel |    6700.00 |
+------------------+------------+
4 rows in set (0.00 sec)
```

```
mysql> select count(date_of_return) from issue_status where date_of_return is not null;
+----------------------+
| count(date_of_return) |
+----------------------+
|                    3 |
+----------------------+
1 row in set (0.01 sec)
```

Date functions

```
select *,datediff(date_of_return,issue_date) difference from
issue_status where datediff(date_of_return,issue_date)>10;

select * from issue_status where MONTHNAME(issue_date)="June" and
YEAR(issue_date)=2021;

select * from issue_status where CURDATE()>expiry_date;

select * from issue_status where DAYOFWEEK(issue_date)=1;

select book_id,DATE_FORMAT(issue_date, "%Y") Year_of_issue from
issue_status;
```




```
mysql> select *,datediff(date_of_return,issue_date) difference from issue_status where datediff(date_of_return,issue_date)>10;
+------------+---------+----------------+-------------+-----------+------------+
| issue_date | book_id | date_of_return | expiry_date | member_id | difference |
+------------+---------+----------------+-------------+-----------+------------+
| 2021-03-06 |    5017 | 2021-03-24     | 2021-04-05  |        99 |         18 |
| 2021-06-13 |    5050 | 2021-06-28     | 2021-07-13  |        10 |         15 |
+------------+---------+----------------+-------------+-----------+------------+
2 rows in set (0.02 sec)
```

```
mysql> select * from issue_status where MONTHNAME(issue_date)="June" and YEAR(issue_date)=2021;
+------------+---------+----------------+-------------+-----------+
| issue_date | book_id | date_of_return | expiry_date | member_id |
+------------+---------+----------------+-------------+-----------+
| 2021-06-13 |    5050 | 2021-06-28     | 2021-07-13  |        10 |
+------------+---------+----------------+-------------+-----------+
1 row in set (0.01 sec)
```

```
mysql> select * from issue_status where CURDATE()>expiry_date;
+------------+---------+----------------+-------------+-----------+
| issue_date | book_id | date_of_return | expiry_date | member_id |
+------------+---------+----------------+-------------+-----------+
| 2021-03-06 |    5017 | 2021-03-24     | 2021-04-05  |        99 |
| 2021-06-13 |    5050 | 2021-06-28     | 2021-07-13  |        10 |
| 2021-09-23 |    5001 | 2021-09-30     | 2021-10-23  |        98 |
+------------+---------+----------------+-------------+-----------+
3 rows in set (0.00 sec)
```

```
mysql> select * from issue_status where DAYOFWEEK(issue_date)=1;
+------------+---------+----------------+-------------+-----------+
| issue_date | book_id | date_of_return | expiry_date | member_id |
+------------+---------+----------------+-------------+-----------+
| 2021-06-13 |    5050 | 2021-06-28     | 2021-07-13  |        10 |
+------------+---------+----------------+-------------+-----------+
1 row in set (0.00 sec)
```

```
mysql> select book_id,DATE_FORMAT(issue_date, "%Y") Year_of_issue from issue_status;
+---------+---------------+
| book_id | Year_of_issue |
+---------+---------------+
|    5050 | 2021          |
|    5012 | 2021          |
|    5010 | 2021          |
|    5001 | 2021          |
|    5017 | 2021          |
+---------+---------------+
5 rows in set (0.00 sec)
```

Numeric functions

```
select book_id,book_name,ceil(price) from book;

select member.member_id,book_id,book_name,round(price) from book, member
where book.member_id=member.member_id;

select book_name,max(purchase_date) from book;

select book_id,max(expiry_date) from issue_status;
```

```
 7 • select book_id,book_name,ceil(price) from book;
 8 • select member.member_id,book_id,book_name,round(price) from book, member where book.member_id=member.member_id;
 9 • select book_name,max(purchase_date) from book;
10 • select book_id,max(expiry_date) from issue_status;
```

```
mysql> select book_id,book_name,ceil(price) from book;
+---------+-------------------------+-------------+
| book_id | book_name               | ceil(price) |
+---------+-------------------------+-------------+
|    5001 | The Story of My Life    |        5055 |
|    5010 | The Hunger Games        |        4551 |
|    5012 | The Great Train Journey |        7000 |
|    5017 | Becoming                |        9951 |
|    5050 | Geronimo Stilton        |        6700 |
+---------+-------------------------+-------------+
5 rows in set (0.00 sec)
```

```
mysql> select member.member_id,book_id,book_name,round(price) from book, member where book.member_id=member.member_id;
+-----------+---------+-------------------------+-------------+
| member_id | book_id | book_name               | round(price) |
+-----------+---------+-------------------------+-------------+
|        59 |    5001 | The Story of My Life    |        5055 |
|        10 |    5010 | The Hunger Games        |        4551 |
|        98 |    5012 | The Great Train Journey |        7000 |
|        10 |    5017 | Becoming                |        9951 |
|        25 |    5050 | Geronimo Stilton        |        6700 |
+-----------+---------+-------------------------+-------------+
5 rows in set (0.01 sec)
```

```
mysql> select book_name,max(purchase_date) from book;
+----------------------+--------------------+
| book_name            | max(purchase_date) |
+----------------------+--------------------+
| The Story of My Life | 2020-07-23         |
+----------------------+--------------------+
1 row in set (0.00 sec)
```

```
mysql> select book_id,max(expiry_date) from issue_status;
+---------+------------------+
| book_id | max(expiry_date) |
+---------+------------------+
|    5017 | 2021-12-15       |
+---------+------------------+
1 row in set (0.00 sec)
```

String functions

```
select book_name,book_id,rack_no,length(book_name) from book;
select phone_no,upper(fname), upper(lname) from details;
select author_id,lcase(author_name),experience from author;
select concat(fname,' ',lname)"member name",phone_no from details;
select book_name,locate('The',book_name) from book;
```

```
12 • select book_name,book_id,rack_no,length(book_name) from book;
13 • select phone_no,upper(fname), upper(lname) from details;
14 • select author_id,lcase(author_name),experience from author;
15 • select concat(fname,' ',lname)"member name",phone_no from details;
16 • select book_name,locate('The',book_name) from book;
```

```
mysql> select book_name,book_id,rack_no,length(book_name) from book;
+------------------------+---------+---------+------------------+
| book_name              | book_id | rack_no | length(book_name) |
+------------------------+---------+---------+------------------+
| The Story of My Life   |    5001 |       8 |               20 |
| The Hunger Games       |    5010 |       3 |               16 |
| The Great Train Journey |    5012 |       1 |               23 |
| Becoming               |    5017 |       6 |                8 |
| Geronimo Stilton       |    5050 |       6 |               16 |
+------------------------+---------+---------+------------------+
5 rows in set (0.00 sec)
```

```
mysql> select phone_no,upper(fname), upper(lname) from details;
+-----------+--------------+--------------+
| phone_no  | upper(fname) | upper(lname) |
+-----------+--------------+--------------+
| 600897885 | RANI         | JAIN         |
| 700897344 | RAJ          | GUPTA        |
| 800897444 | SHYAM        | AGARWAL      |
| 880089785 | SIDDHI       | KUMARI       |
| 900897885 | RAM          | KUMAR        |
+-----------+--------------+--------------+
5 rows in set (0.00 sec)
```

```
mysql> select author_id,lcase(author_name),experience from author;
+-----------+--------------------+------------+
| author_id | lcase(author_name) | experience |
+-----------+--------------------+------------+
|      1001 | helen keller       |         40 |
|      1002 | suzanne collins    |         20 |
|      1003 | elisabetta dami    |         22 |
|      1004 | ruskin bond        |         46 |
|      1005 | michelle obama     |         21 |
+-----------+--------------------+------------+
5 rows in set (0.08 sec)
```

```
mysql> select concat(fname,' ',lname)"member name",phone_no from details;
+---------------+-----------+
| member name   | phone_no  |
+---------------+-----------+
| Rani Jain     | 600897885 |
| Raj Gupta     | 700897344 |
| Shyam Agarwal | 800897444 |
| Siddhi Kumari | 880089785 |
| Ram Kumar     | 900897885 |
+---------------+-----------+
5 rows in set (0.00 sec)
```

```
mysql> select book_name,locate('The',book_name) from book;
+------------------------+------------------------+
| book_name              | locate('The',book_name) |
+------------------------+------------------------+
| The Story of My Life   |                      1 |
| The Hunger Games       |                      1 |
| The Great Train Journey |                     1 |
| Becoming               |                      0 |
| Geronimo Stilton       |                      0 |
+------------------------+------------------------+
5 rows in set (0.00 sec)
```

Group by and having functions

```
select * from book group by rack_no;

select book_name,category,price,member_id from book where price>5000
group by member_id;

select * from book group by category having max (price);
```

```
24 •  select * from book group by rack_no;
25 •  select book_name,category,price,member_id from book where price>5000 group by member_id;
26 •  select * from book group by category having max (price);
```

```
mysql> select * from book group by rack_no;
+---------+---------+---------+---------------+-------------+-------------------------+-----------+-----------+
| book_id | rack_no | price   | purchase_date | category    | book_name               | member_id | author_id |
+---------+---------+---------+---------------+-------------+-------------------------+-----------+-----------+
|    5001 |       8 | 5055.00 | 2020-07-23    | Memoir      | The Story of My Life    |        59 |      1001 |
|    5010 |       2 | 4655.00 | 1990-12-29    | Fiction     | The Hunger Games        |        10 |      1002 |
|    5012 |       1 | 7000.00 | 2001-09-04    | Fiction     | The Great Train Journey |        98 |      1004 |
|    5017 |       6 | 9000.00 | 2005-04-30    | Non-Fiction | Becoming                |        10 |      1005 |
+---------+---------+---------+---------------+-------------+-------------------------+-----------+-----------+
4 rows in set (0.01 sec)
```

```
mysql> select book_name,category,price,member_id from book where price>5000 group by member_id;
+-------------------------+-----------------+---------+-----------+
| book_name               | category        | price   | member_id |
+-------------------------+-----------------+---------+-----------+
| Becoming                | Non-Fiction     | 9000.00 |        10 |
| Geronimo Stilton        | Childern's Novel | 6700.00 |        25 |
| The Story of My Life    | Memoir          | 5055.00 |        59 |
| The Great Train Journey | Fiction         | 7000.00 |        98 |
+-------------------------+-----------------+---------+-----------+
4 rows in set (0.00 sec)
```

```
mysql> select * from book group by category having max(price);
+---------+---------+---------+---------------+-----------------+---------------------+-----------+-----------+
| book_id | rack_no | price   | purchase_date | category        | book_name           | member_id | author_id |
+---------+---------+---------+---------------+-----------------+---------------------+-----------+-----------+
|    5001 |       8 | 5055.00 | 2020-07-23    | Memoir          | The Story of My Life |        59 |      1001 |
|    5010 |       2 | 4655.00 | 1990-12-29    | Fiction         | The Hunger Games    |        10 |      1002 |
|    5017 |       6 | 9000.00 | 2005-04-30    | Non-Fiction     | Becoming            |        10 |      1005 |
|    5050 |       6 | 6700.00 | 1999-11-11    | Childern's Novel | Geronimo Stilton    |        25 |      1003 |
+---------+---------+---------+---------------+-----------------+---------------------+-----------+-----------+
4 rows in set (0.00 sec)
```

Join operation

```
select b.book_id, m.member_id, b.book_name, i.date_of_return from member
m, book b, issue_status i where m.member_id=b.member_id and
i.book_id=b.book_id and date_of_return is not null;

select author_name, book_name, category, price, m.member_id from member
m, book b, author a where m.member_id = b.member_id and b.author_id =
a.author_id and experience < 30 and rack_no = 6;
```

```
33 •  select b.book_id, m.member_id, b.book_name, i.date_of_return from member m, book b, issue_status i
34        where m.member_id=b.member_id and i.book_id=b.book_id and date_of_return is not null;
35 •  select author_name, book_name, category, price, m.member_id from member m, book b, author a
36        where m.member_id = b.member_id and b.author_id = a.author_id and experience < 30 and rack_no = 6;
```

```
mysql> select b.book_id, m.member_id, b.book_name, i.date_of_return from member m, book b, issue_status i where m.member_id=b.member_id and i.book_id=b.book_id and date_of_
return is not null;
+---------+-----------+---------------------+----------------+
| book_id | member_id | book_name           | date_of_return |
+---------+-----------+---------------------+----------------+
|    5017 |        10 | Becoming            | 2021-03-24     |
|    5050 |        25 | Geronimo Stilton    | 2021-06-28     |
|    5001 |        59 | The Story of My Life | 2021-09-30     |
+---------+-----------+---------------------+----------------+
3 rows in set (0.00 sec)
```

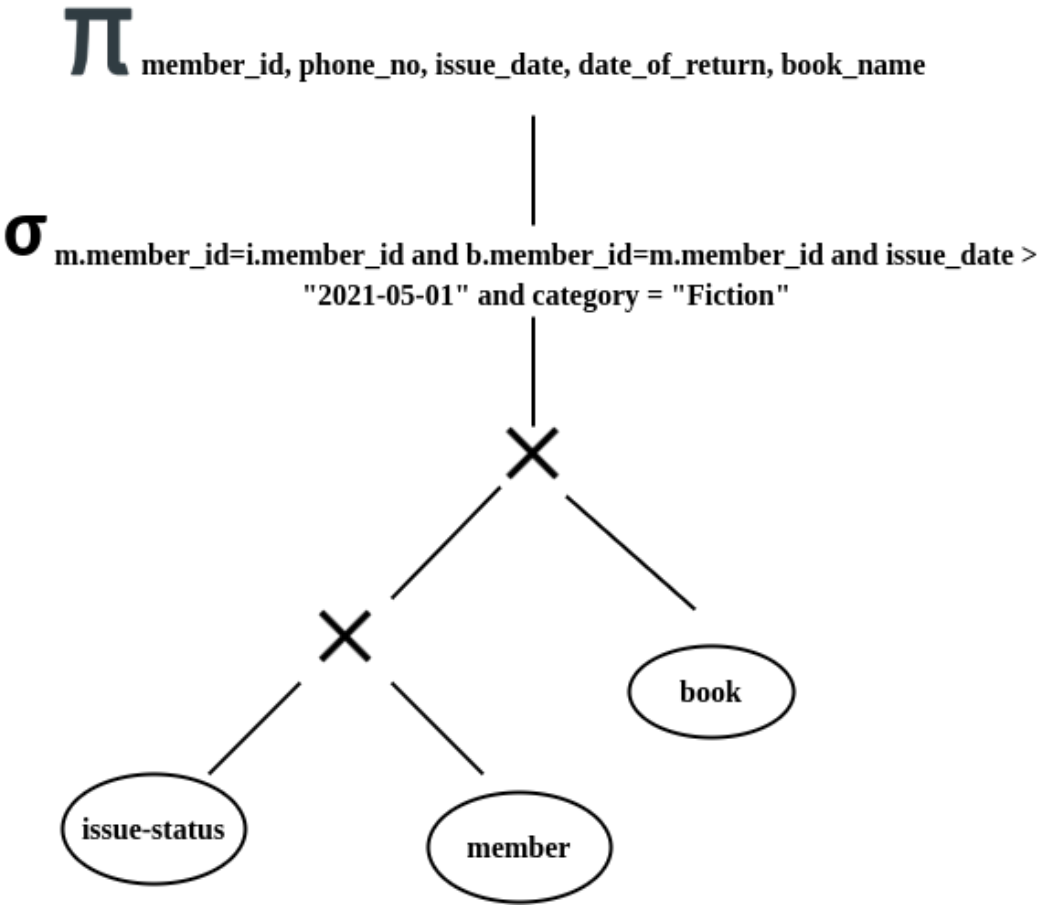
```
mysql> select author_name, book_name, category, price, m.member_id from member m, book b, author a where m.member_id = b.member_id and b.author_id = a.author_id and experie
nce < 30 and rack_no = 6;
+----------------+----------------+-----------------+---------+-----------+
| author_name    | book_name      | category        | price   | member_id |
+----------------+----------------+-----------------+---------+-----------+
| Michelle Obama | Becoming       | Non-Fiction     | 9000.00 |        10 |
| Elisabetta Dami | Geronimo Stilton | Childern's Novel | 6700.00 |        25 |
+----------------+----------------+-----------------+---------+-----------+
2 rows in set (0.00 sec)
```

Query optimization

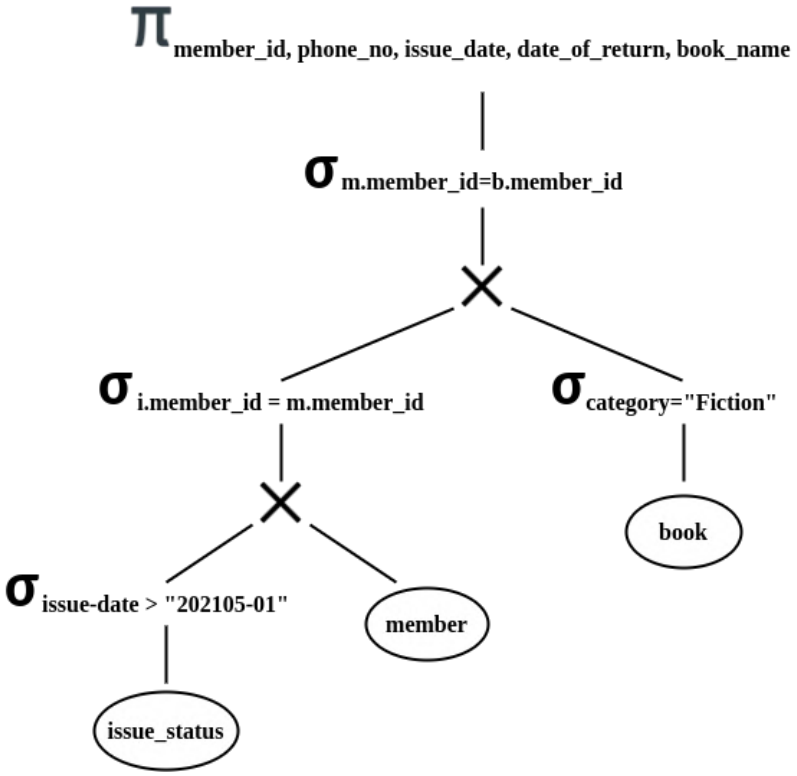
Query 1

```
SELECT
    m.member_id, phone_no, issue_date, date_of_return, book_name
FROM
    member m,
    issue_status i,
    book b
WHERE
    m.member_id = i.member_id
        AND b.member_id = m.member_id
        AND issue_date > '2021-05-01'
        AND category = 'Fiction';
```

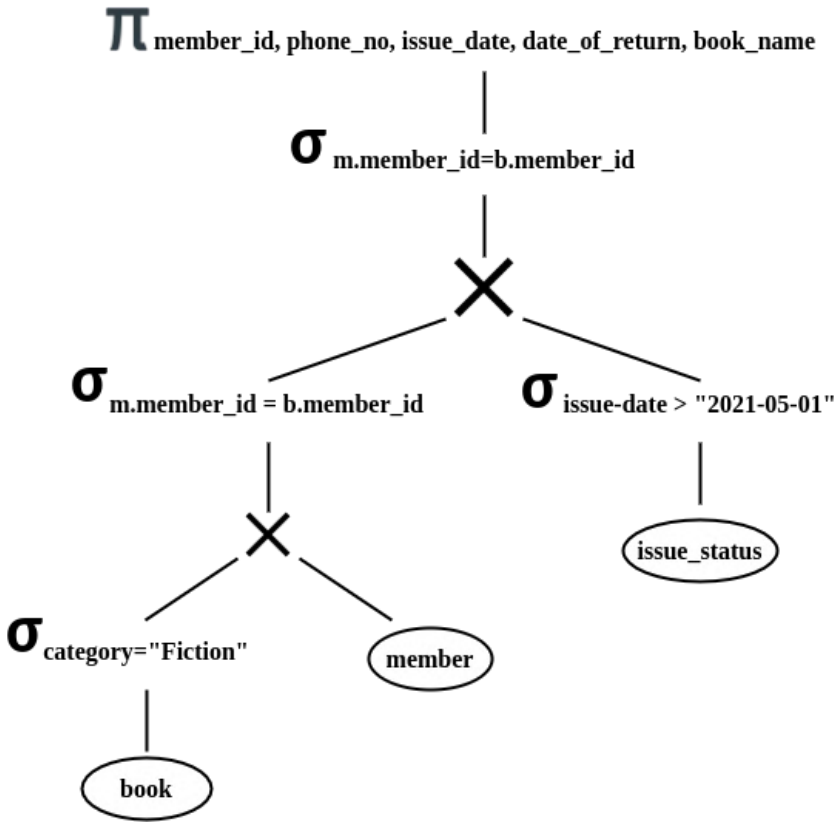
Step 1: Canonical form.



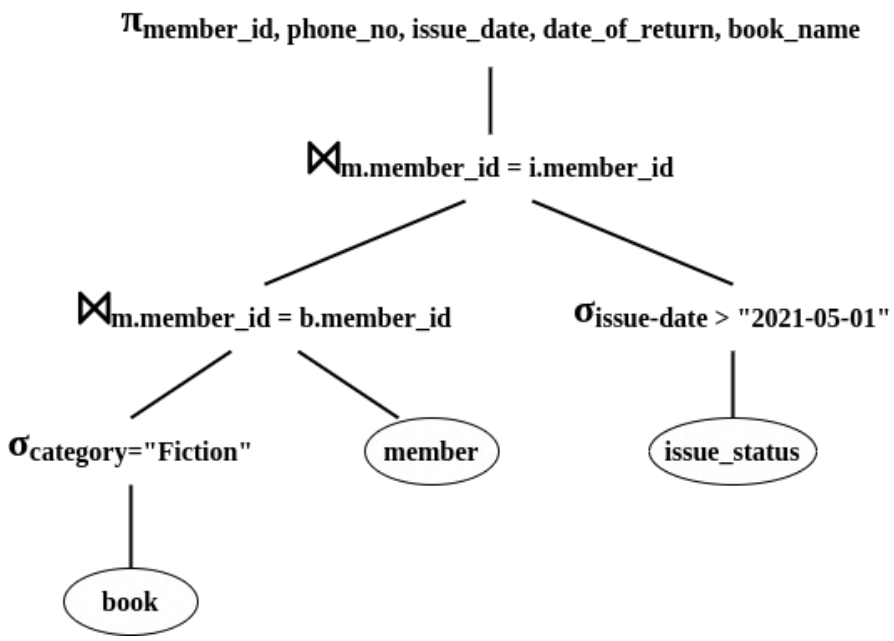
Step 2: Move select operations down the tree.



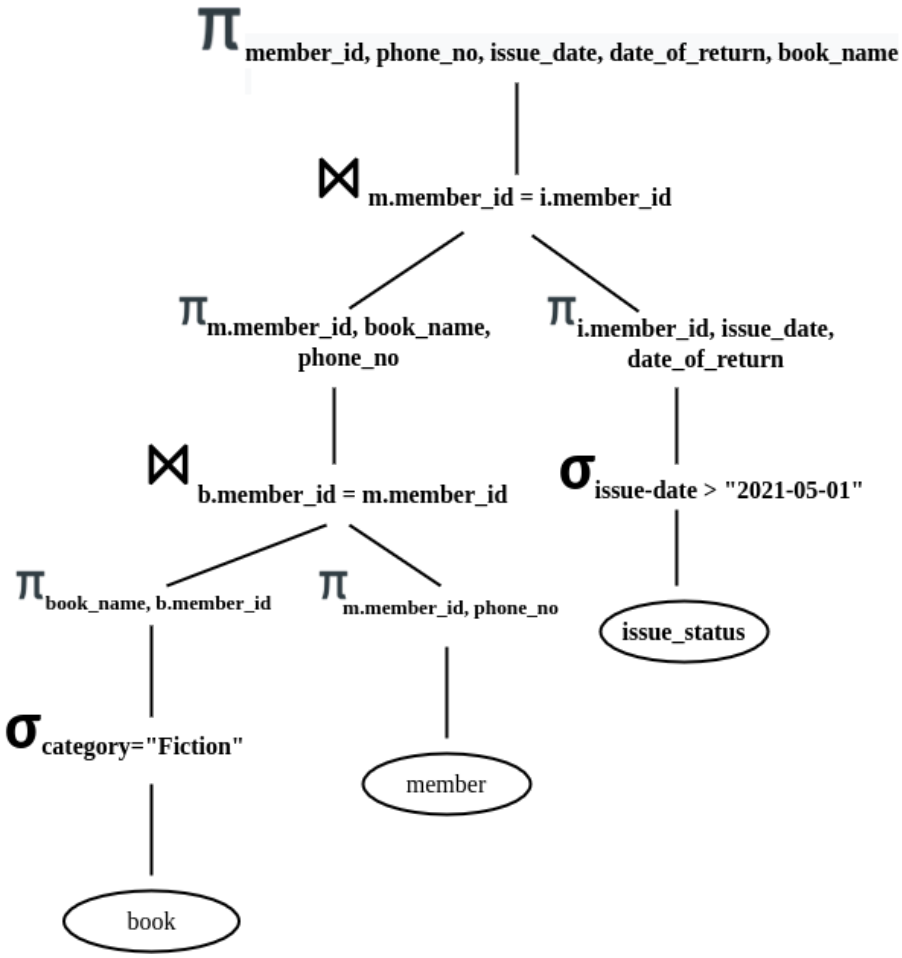
Step 3: Applying the most restrictive select operation first.



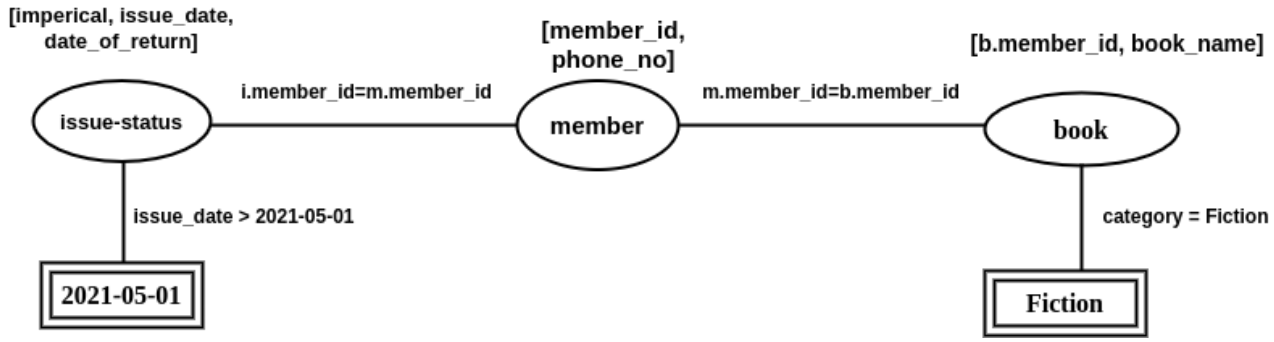
Step 4: Replace cartesian product and select with join operation.



Step 5: Moving project operation down the query tree.



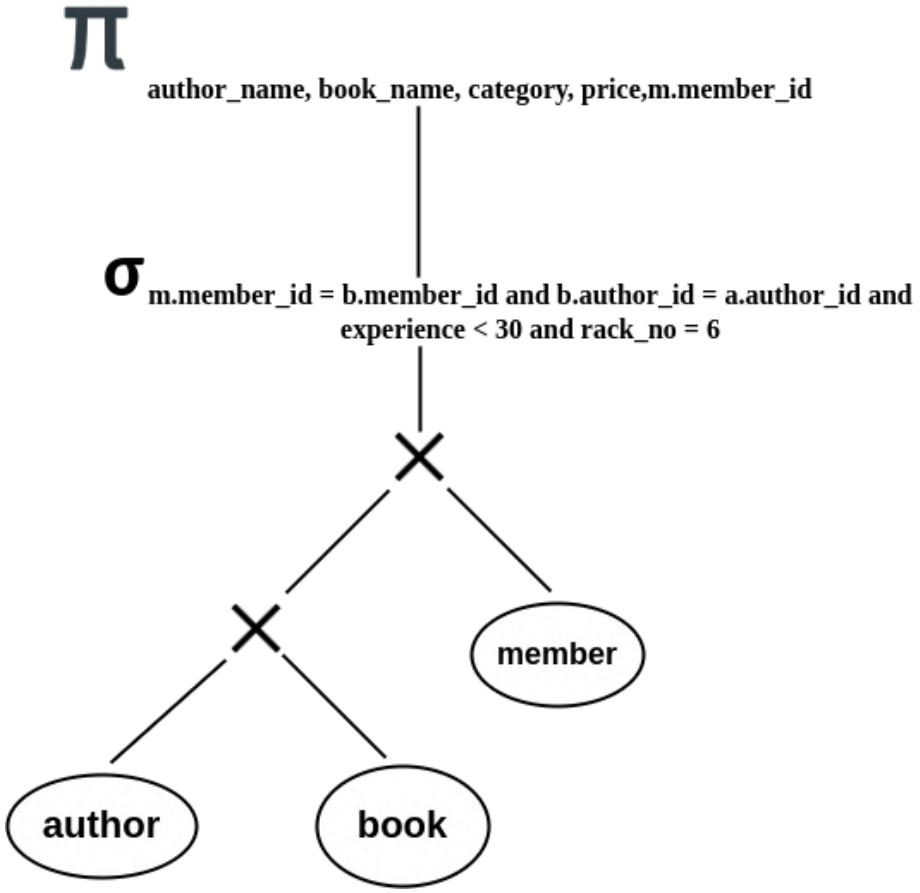
Query Graph:


[imperical, issue_date, date_of_return]
[member_id, phone_no]
[b.member_id, book_name]
issue-status
i.member_id=m.member_id
member
m.member_id=b.member_id
book
issue_date > 2021-05-01
category = Fiction
2021-05-01
Fiction

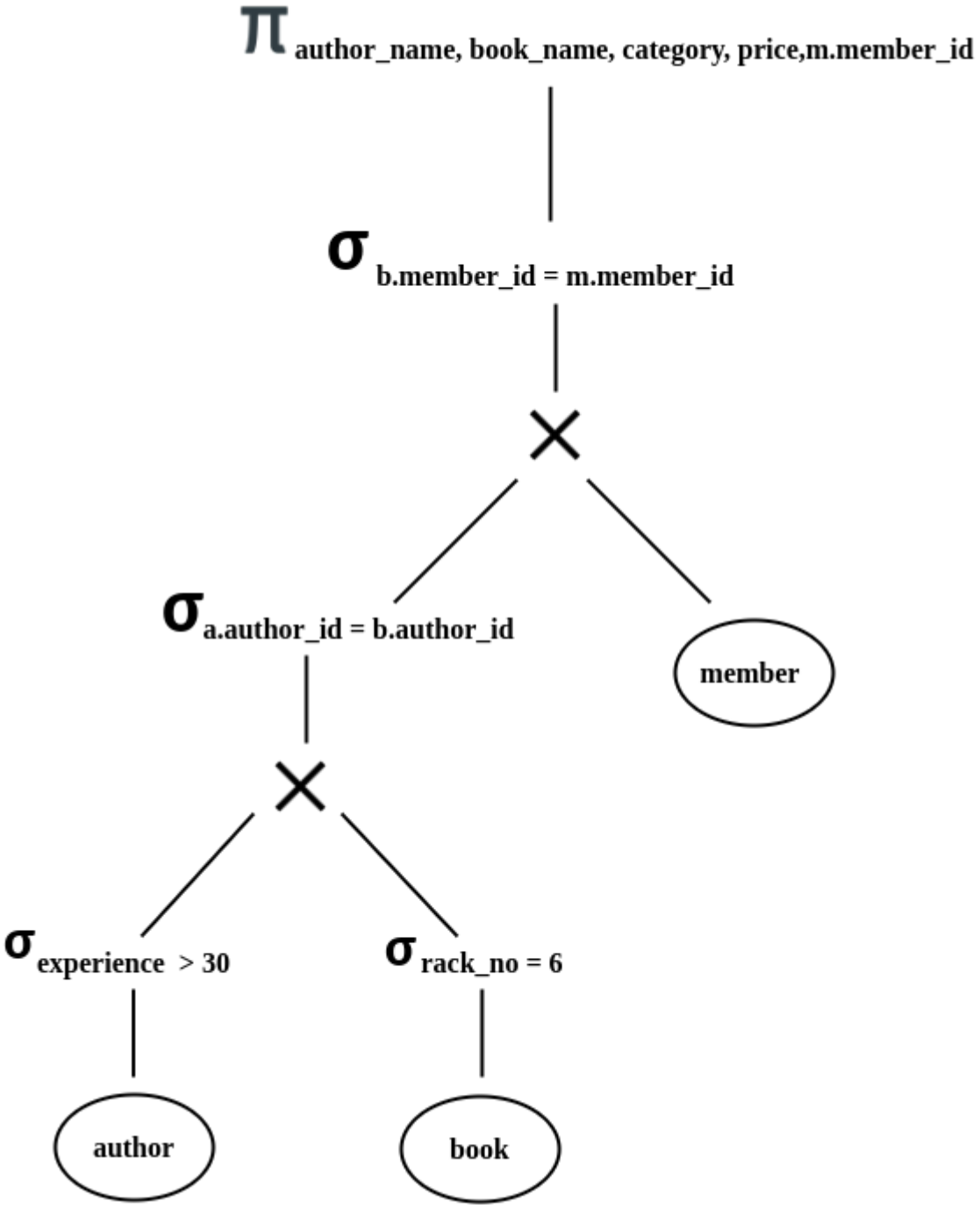
Query 2

```
SELECT
    author_name, book_name, category, price, m.member_id
FROM
    member m,
    book b,
    author a
WHERE
    m.member_id = b.member_id
        AND b.author_id = a.author_id
        AND experience < 30
        AND rack_no = 6;
```

Step 1: Canonical form.



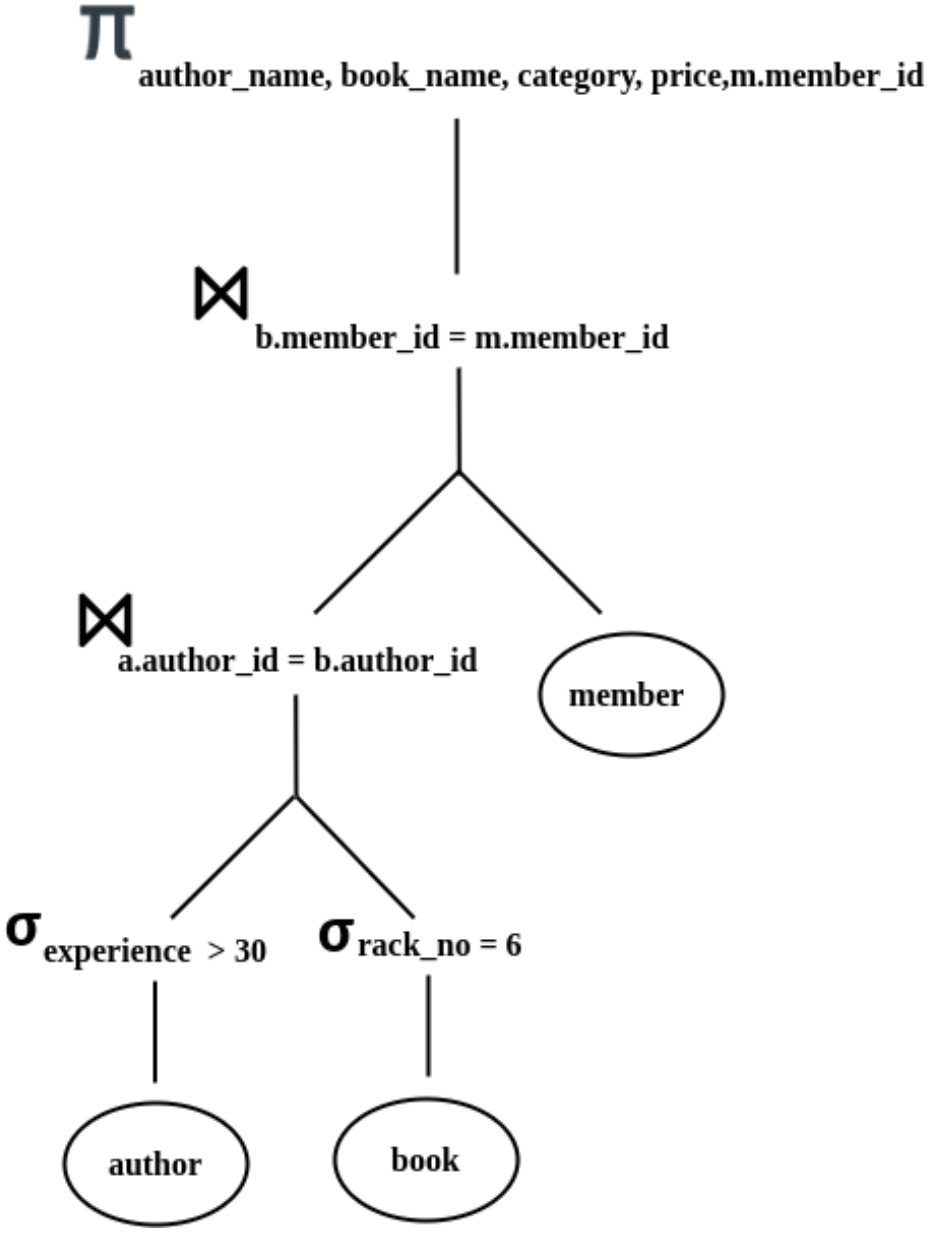
Step 2: Move select operations down the tree.



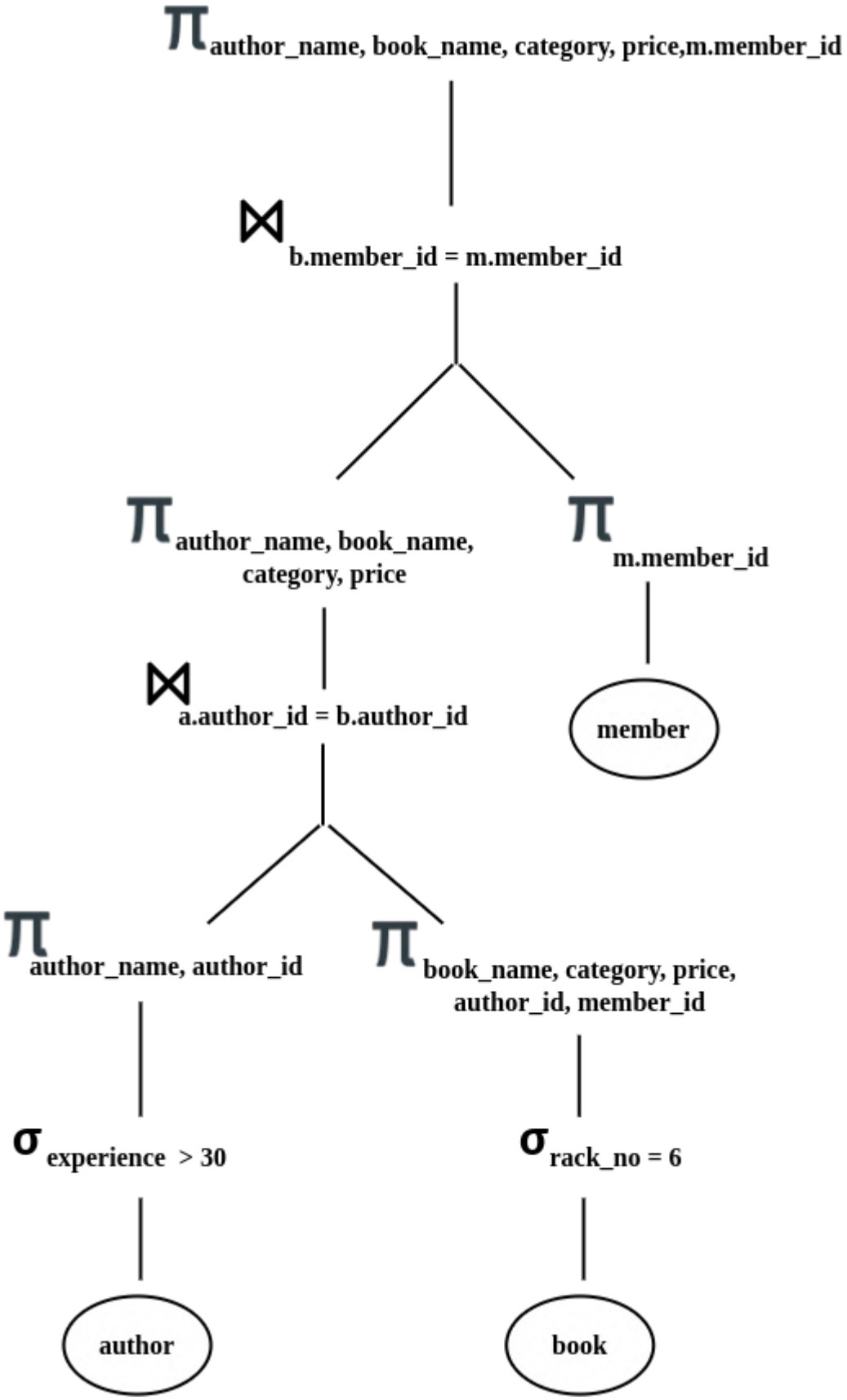
Step 3: Applying the most restrictive select operation first.

In this case, it is already taken care during the designing phase that the number of records will be fixed prior and while selecting the query it is kept in mind that upon applying step 3 the number of rows will remain the same after the Cartesian product.

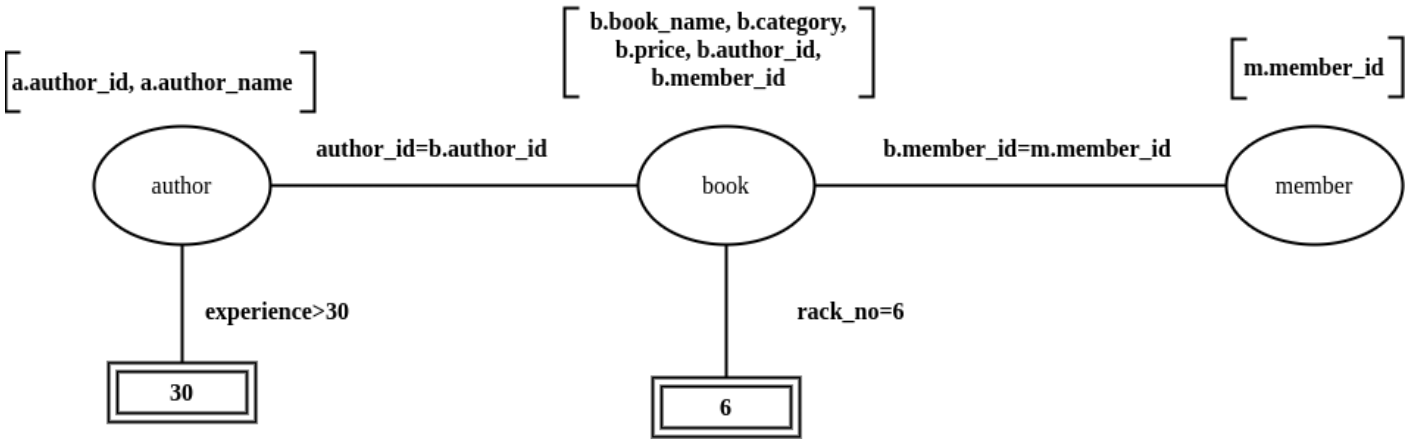
Step 4: Replace cartesian product and select with join operation.



Step 5: Moving project operation down the query tree.



Query graph:



Indexing

Indexing is a way to optimize the performance of a database by minimizing the number of disk accesses required when a query is processed.

An index is an access structure that allows us to find the location of records with given values in certain fields which are called index fields.

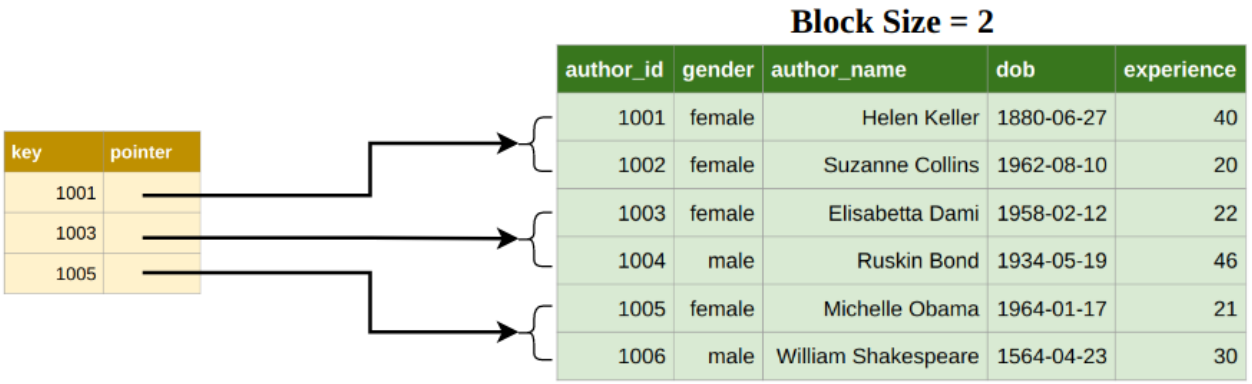
Primary indexing

Primary indexing is applied on the primary key (has a unique value of each record). There are two types of primary indexing, sparse and dense. In sparse primary indexing, the number of rows in the index table is less than the number of rows in the main table whereas in dense primary indexing, the number of rows in the index table is equal to the number of rows in the main table.

- **Sparse indexing**

  AUTHOR (author_id, gender, author_name, dob, experience)

  author_id is a sorted key attribute, so we can apply sparse indexing on this column.



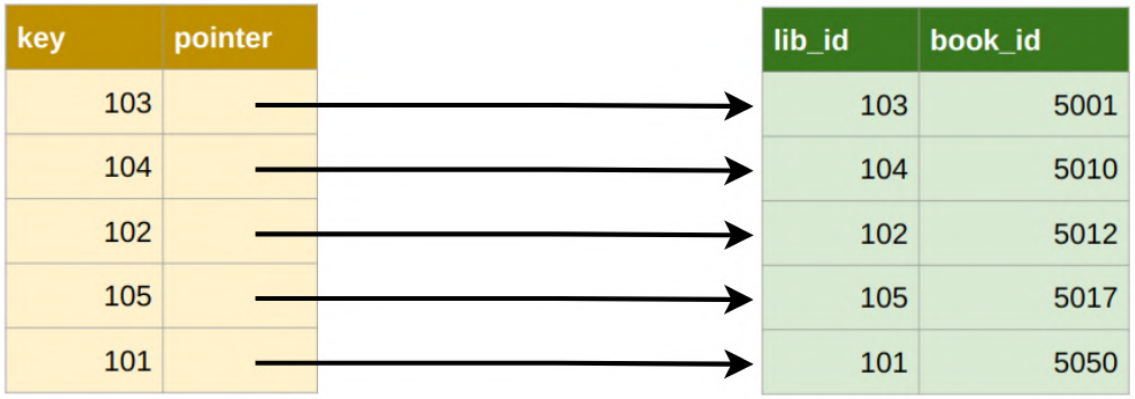
Primary indexing applied on the primary key column of the table.

The values in *author_id* are unique and ordered.

- **Dense indexing**

  MANAGES (lib_id, book_id)

  lib_id is an unsorted key attribute, so we can apply dense indexing on this column.

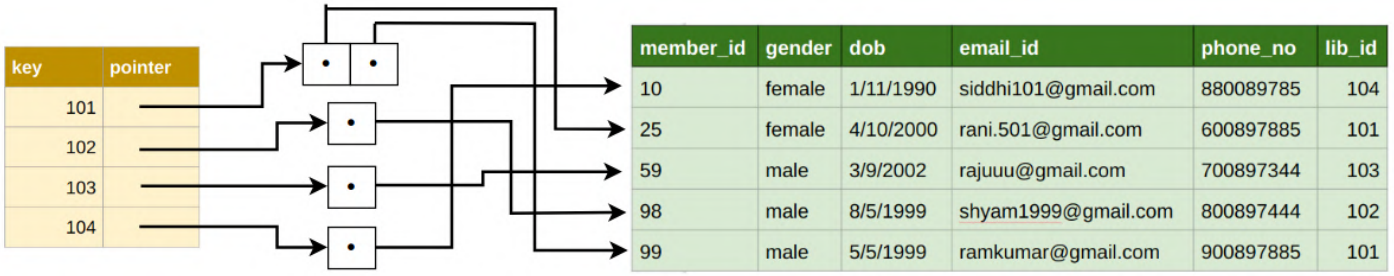


Primary indexing applied on the primary key column of the table.

The values in *lib_id* are unique and unordered.

Secondary indexing

Secondary indexing can be applied on a key attribute that has unique values for all records or also on a non-key value that has repeating values for different values. In case of a key-attribute, each record will point to exactly one secondary key, so the indexing becomes dense. In case of a non-key attribute, the index table consists of a secondary key and a block pointer and these block pointers further point to the records that point at that secondary index value.



key	pointer
101	
102	
103	
104	

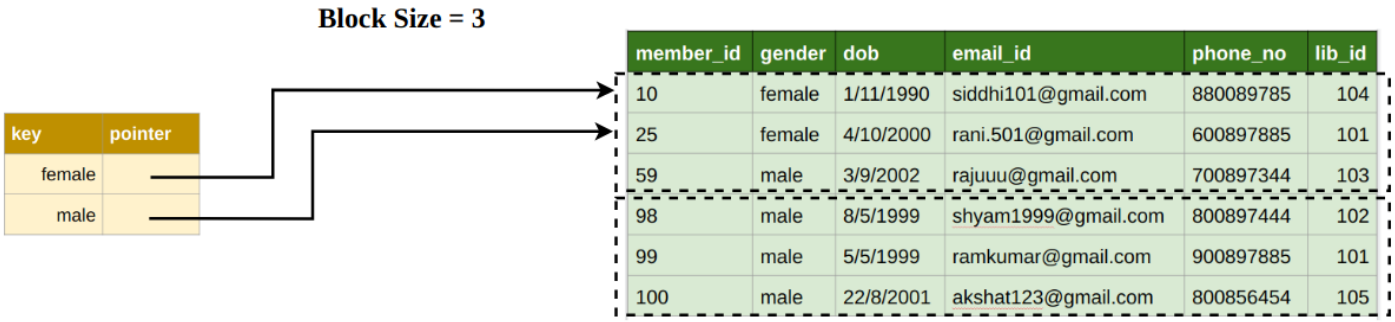
member_id	gender	dob	email_id	phone_no	lib_id
10	female	1/11/1990	siddhi101@gmail.com	880089785	104
25	female	4/10/2000	rani.501@gmail.com	600897885	101
59	male	3/9/2002	rajuuu@gmail.com	700897344	103
98	male	8/5/1999	shyam1999@gmail.com	800897444	102
99	male	5/5/1999	ramkumar@gmail.com	900897885	101

Secondary indexing applied on non-key attributes of the table.

The values in *lib_id* are redundant and unordered.

Cluster indexing

Cluster indexing is applied to a non-key attribute. Here, the attribute (also called the clustering index) may have repeating values, the pointer points to the block where the first entry of the key is present.



key	pointer
female	
male	

member_id	gender	dob	email_id	phone_no	lib_id
10	female	1/11/1990	siddhi101@gmail.com	880089785	104
25	female	4/10/2000	rani.501@gmail.com	600897885	101
59	male	3/9/2002	rajuuu@gmail.com	700897344	103
98	male	8/5/1999	shyam1999@gmail.com	800897444	102
99	male	5/5/1999	ramkumar@gmail.com	900897885	101
100	male	22/8/2001	akshat123@gmail.com	800856454	105

Cluster indexing applied on non-key attributes of the table.

The values in *gender* are redundant and ordered.

Conclusion

This project was designed to make the organisation and functioning of a small scale library easy and efficient. It ensures that the library is well planned and running. Since the routes are to be extremely scalable, they are very easy to handle in the case that there is a need of making changes or implementing new features.

Firstly, the various requirements of the project were discussed through a problem statement. Then the problem statements were broken down and the multiple aspects of the system were identified. Next, a pictorial representation of the system was created through which, the various relations/tables were made. Then, the tables were further simplified/broken down to make the system more flexible and quicker through normalization, after which, the queries were executed leading to the creation of the database. Finally, indexing was done on all the relations to make the process even more efficient.