DATABASE SYSTEM OF PUBLIC LIBRARY

APRIL 26, 2017

CS 6360.002

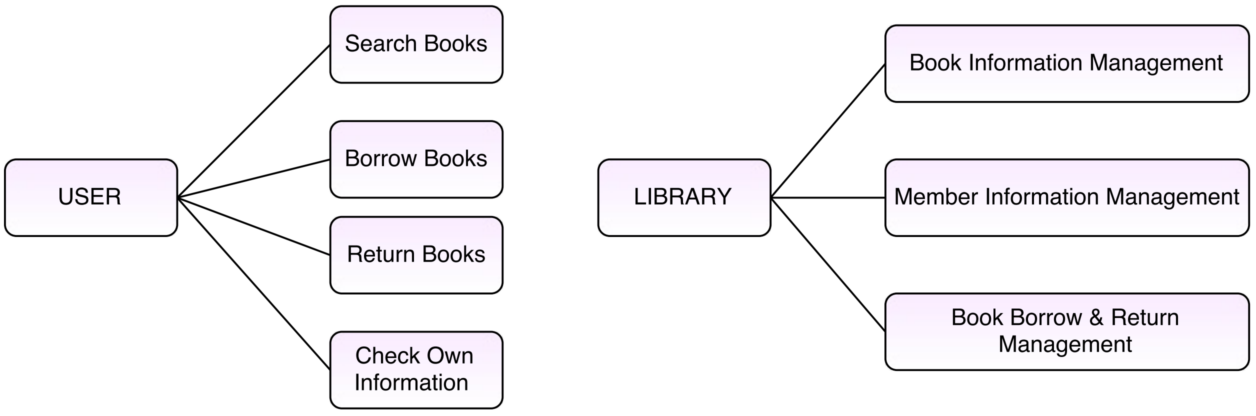
AN LUO
CHUNHUI LIN
JIANGYUE XI

1. Project Description

In this project, we design a database system for a public library, which provides services to its members and librarians. We implement this system through the following steps:

- Analyze public library database system requirements.
- Create ER diagram.
- Create relational schema.
- Discuss database normalization on the database tables and update relational schema.
- Create tables using SQL commands.
- Define two different stored procedures and two triggers.
- Implement CRUD operations for AUTHOR table using PHP.

2. Library Database System Requirements


USER
Search Books
Borrow Books
Return Books
Check Own Information
LIBRARY
Book Information Management
Member Information Management
Book Borrow & Return Management

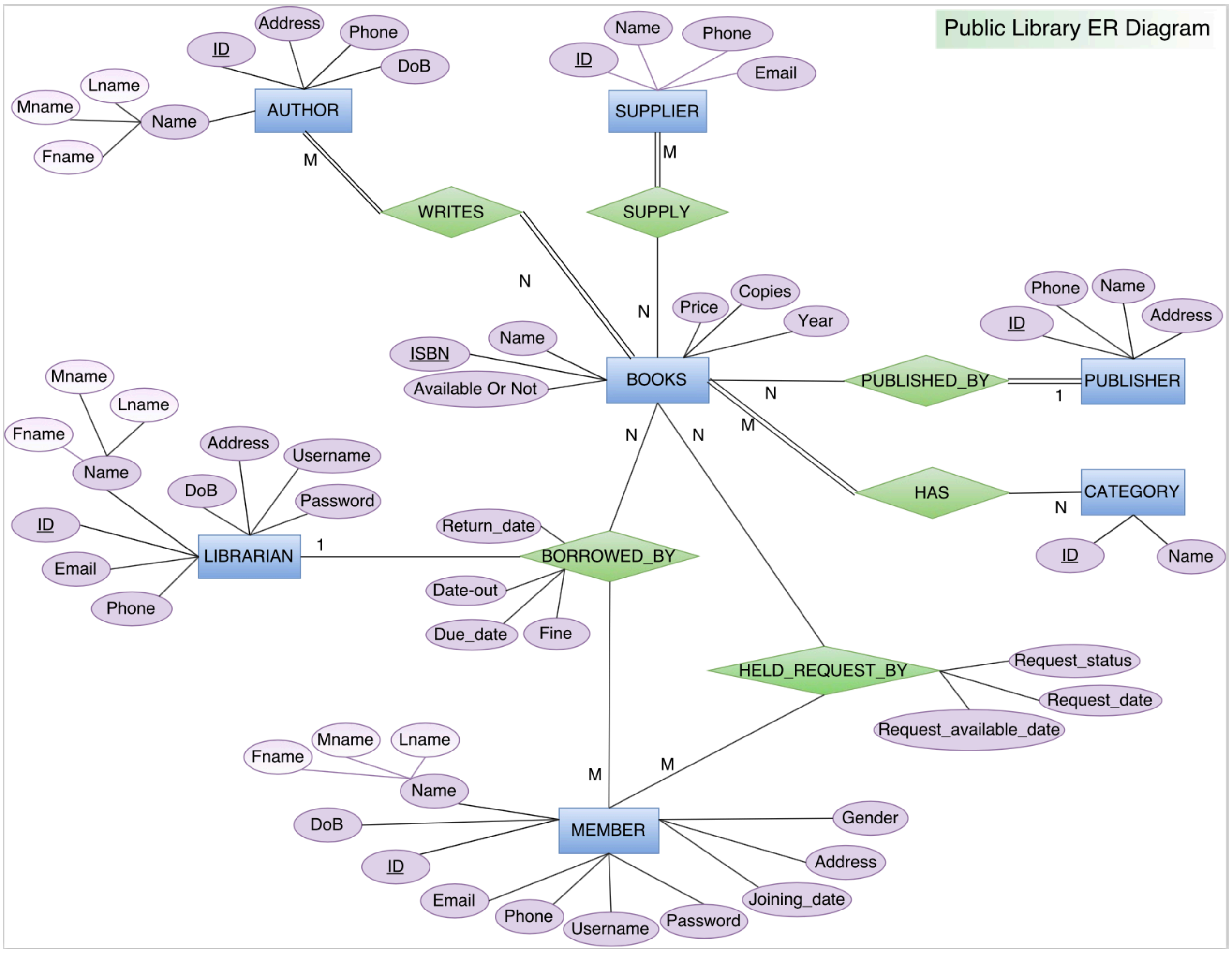
Public System Requirements

According to the library system requirements, we design the following data item and data structure:

- BOOKS(<u>ISBN</u>, Name, Price, Copies, Year, Available Or Not)
- AUTHOR(<u>ID</u>, Fname, Mname, Lname, Address, Phone, DoB)
- PUBLISHER(<u>ID</u>, Name, Address, Phone)
- CATEGORY(<u>ID</u>, Name)

- SUPPLIER(<u>ID</u>, Name, Address, Phone)
- MEMBER(<u>ID</u>, Fname, Mname, Lname, Address, Phone, DoB, Email, Gender, Joining_date, Username, Password)
- LIBRARIAN(<u>ID</u>, Fname, Mname, Lname, Address, Phone, DoB, Email, Username, Password)

3. Library ER Diagram



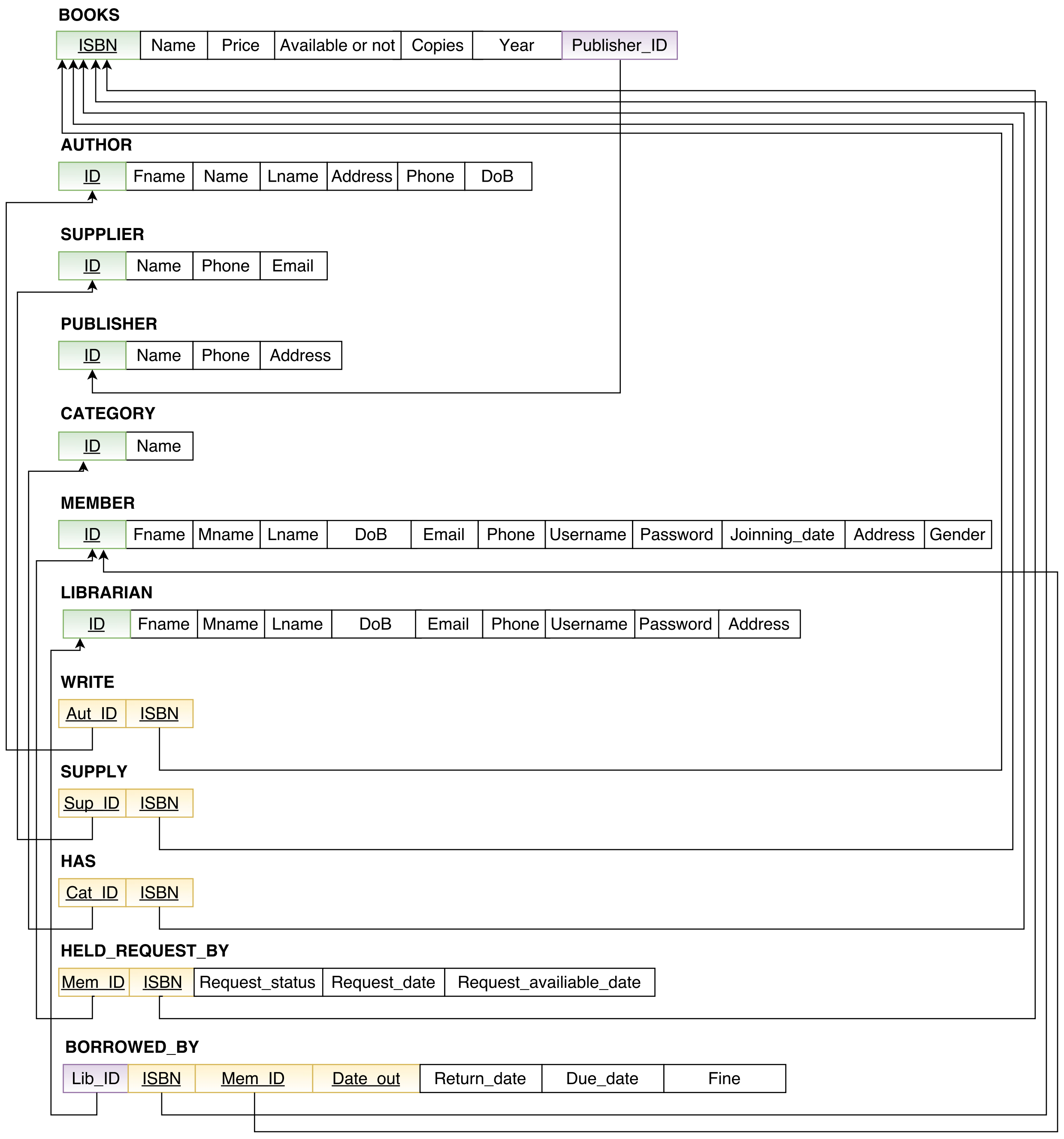
4. Library Relational schema

From the ER diagram, we can conclude:

Relation			Method
Binary relationships	M:N	WRITES	1. Create new tables: WRITES, SUPPLY, HAS, HELD_REQUEST_BY. 2. Include as foreign key attributes in new relations the primary keys of the relations that represent the participating entity types; the combination will form the primary key of the new relation. 3. Include any simple attributes of the M:N relationship type as attributes of the new relation.
		SUPPLY	
		HAS	
		HELD_REQUEST_BY	
	1:N	PUBLISHED_BY	1. Identify N-side of the relationship type which is BOOKS. 2. Include as foreign key in BOOKS the primary key of the relation PUBLISHER.
Ternary relationships	BORROWED_BY		1. Create a new relationship BORROWED_BY. 2. Include as foreign key attributes in BORROWED_BY the primary keys of the relations that represent the participating entity types. 3. Include any simple attributes of the 3-ary relationship type as attributes of BORROWED_BY.

Table 1: Relationship Type in ER Diagram

Using the methods in the Table 1, we create the relational schema below:


BOOKS
ISBN | Name | Price | Available or not | Copies | Year | Publisher_ID
AUTHOR
ID | Fname | Name | Lname | Address | Phone | DoB
SUPPLIER
ID | Name | Phone | Email
PUBLISHER
ID | Name | Phone | Address
CATEGORY
ID | Name
MEMBER
ID | Fname | Mname | Lname | DoB | Email | Phone | Username | Password | Joinning_date | Address | Gender
LIBRARIAN
ID | Fname | Mname | Lname | DoB | Email | Phone | Username | Password | Address
WRITE
Aut_ID | ISBN
SUPPLY
Sup_ID | ISBN
HAS
Cat_ID | ISBN
HELD_REQUEST_BY
Mem_ID | ISBN | Request_status | Request_date | Request_availiable_date
BORROWED_BY
Lib_ID | ISBN | Mem_ID | Date_out | Return_date | Due_date | Fine

Library Relational Schema 🟩 Primary Key 🟪 Foreign Key 🟨 Combination Primary Key

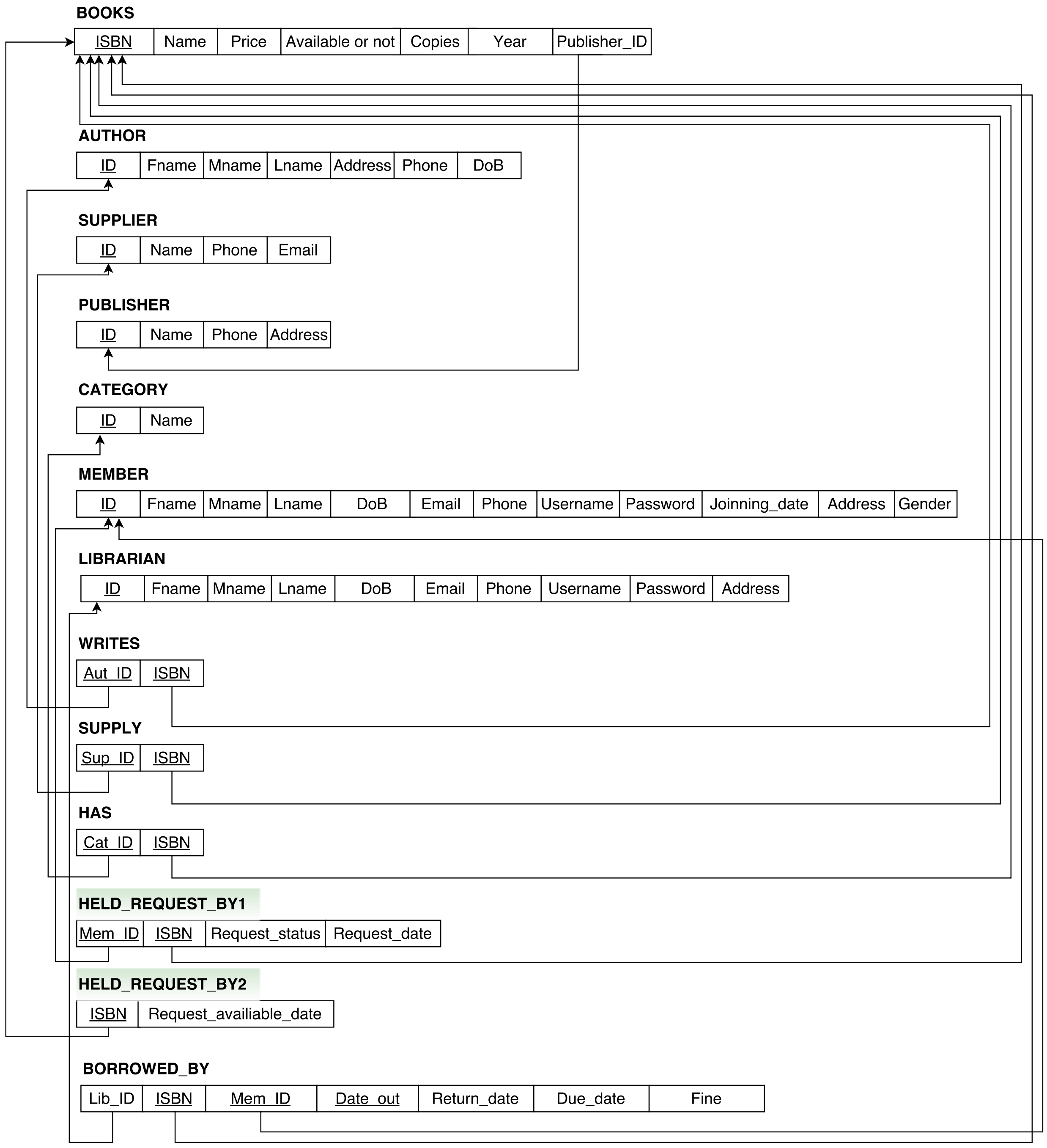
5. Library Database Normalization

According to the following functional dependency:

Mem_ID, ISBN → Request_status, Request_date

ISBN → Request_availiable_date

We normalize our tables into 3NF:


BOOKS
ISBN | Name | Price | Available or not | Copies | Year | Publisher_ID
AUTHOR
ID | Fname | Mname | Lname | Address | Phone | DoB
SUPPLIER
ID | Name | Phone | Email
PUBLISHER
ID | Name | Phone | Address
CATEGORY
ID | Name
MEMBER
ID | Fname | Mname | Lname | DoB | Email | Phone | Username | Password | Joinning_date | Address | Gender
LIBRARIAN
ID | Fname | Mname | Lname | DoB | Email | Phone | Username | Password | Address
WRITES
Aut_ID | ISBN
SUPPLY
Sup_ID | ISBN
HAS
Cat_ID | ISBN
HELD_REQUEST_BY1
Mem_ID | ISBN | Request_status | Request_date
HELD_REQUEST_BY2
ISBN | Request_availiable_date
BORROWED_BY
Lib_ID | ISBN | Mem_ID | Date_out | Return_date | Due_date | Fine

Final Relation Schema

6. Create Tables Using SQL Commands

Create tables in MAMP with the following SQL commands:

```
-- Host: localhost:8889
-- Generation Time: Apr 19, 2017 at 07:51 AM
-- Server version: 5.6.35
-- PHP Version: 7.0.15
-- Database: `Public Library`
```

6.1 Create Tables

```
--Table structure for table `AUTHOR`
CREATE TABLE AUTHOR (
  ID                        int(10)      NOT NULL,
  Fname                     varchar(15) NOT NULL,
  Mname                     varchar(15) DEFAULT NULL,
  Lname                     varchar(15) NOT NULL,
  Address                   varchar(30) DEFAULT NULL,
  Phone                     int(10)      DEFAULT NULL,
  DoB                       date         DEFAULT NULL
);

--Table structure for table `BOOKS`
CREATE TABLE BOOKS (
  ISBN                      int(13)      NOT NULL,
  Name                      varchar(15) NOT NULL,
  Price                     float        DEFAULT NULL,
  Available or not          varchar(10) NOT NULL,
  Copies                    int(2)       DEFAULT NULL,
  Year                      year(4)      DEFAULT NULL,
  Publisher_ID              int(10)      DEFAULT NULL
);

--Table structure for table `BORROWED_BY`
CREATE TABLE BORROWED_BY (
  Lib_ID                    int(10)      NOT NULL,
  ISBN                      int(13)      NOT NULL,
  Mem_ID                    int(10)      NOT NULL,
  Date_out                  date         NOT NULL,
  Return_date               date         NOT NULL,
  Due_date                  date         NOT NULL,
  Fine                      float        DEFAULT NULL
);

--Table structure for table `CATEGORY`
CREATE TABLE CATEGORY (
  ID                        int(10)      NOT NULL,
  Name                      varchar(30) NOT NULL
);

--Table structure for table `HAS`
CREATE TABLE HAS (
  Cat_ID                    int(10)      NOT NULL,
  ISBN                      int(13)      NOT NULL
);

--Table structure for table `HELD_REQUEST_BY1`
CREATE TABLE HELD_REQUEST_BY1 (
  Mem_ID                int(10)     NOT NULL,
  ISBN                  int(13)     NOT NULL,
  Request_status        varchar(10) NOT NULL,
  Request_date          date        NOT NULL
);

--Table structure for table `HELD_REQUEST_BY2`
CREATE TABLE HELD_REQUEST_BY2 (
  ISBN                        int(13)     NOT NULL,
  Request_availiable_date     date        NOT NULL
);

--Table structure for table `LIBRARIAN`
CREATE TABLE LIBRARIAN (
  ID                int(10)            NOT NULL,
  Fname             varchar(15)        NOT NULL,
  Mname             varchar(15)        DEFAULT NULL,
  Lname             varchar(15)        NOT NULL,
  DoB               date               NOT NULL,
  Email             varchar(30)        NOT NULL,
  Phone             int(10)            NOT NULL,
  Username          varchar(30)        NOT NULL,
  Password          varchar(15)        NOT NULL,
  Address           varchar(30)        NOT NULL
);

--Table structure for table `MEMBER`
CREATE TABLE MEMBER (
  ID                int(10)            NOT NULL,
  Fname             varchar(15)        NOT NULL,
  Mname             varchar(15)        DEFAULT NULL,
  Lname             varchar(15)        NOT NULL,
  DoB               date               NOT NULL,
  Email             varchar(30)        DEFAULT NULL,
  Phone             int(10)            DEFAULT NULL,
  Username          varchar(30)        NOT NULL,
  Password          varchar(15)        NOT NULL,
  Joinning_date     date               NOT NULL,
  Address           varchar(30)        NOT NULL,
  Gender            varchar(1)         DEFAULT NULL,
);

--Table structure for table `PUBLISHER`
CREATE TABLE PUBLISHER (
  ID                int(10)            NOT NULL,
  Name              varchar(30)        NOT NULL,
  Phone             int(10)            DEFAULT NULL,
  Address           varchar(30)        DEFAULT NULL
);

--Table structure for table `SUPPLIER`
```

```
CREATE TABLE SUPPLIER (
  ID                int(10)           NOT NULL,
  Name              varchar(30)       NOT NULL,
  Phone             int(10)           DEFAULT NULL,
  Email             varchar(30)       DEFAULT NULL
);

--Table structure for table `SUPPLY`
CREATE TABLE SUPPLY (
  Sup_ID            int(10)           NOT NULL,
  ISBN              int(13)           NOT NULL
);

--Table structure for table `WRITES`
CREATE TABLE WRITES (
  Aut_ID            int(10)           NOT NULL,
  ISBN              int(13)           NOT NULL
);
```

6.2 Add Keys

```
-- Indexes for table `AUTHOR`
ALTER TABLE AUTHOR
  ADD PRIMARY KEY (ID);

-- Indexes for table `BOOKS`
ALTER TABLE BOOKS
  ADD PRIMARY KEY (ISBN),
  ADD KEY Books_Publisher (Publisher_ID);

-- Indexes for table `BORROWED_BY`
ALTER TABLE BORROWED_BY
  ADD PRIMARY KEY (ISBN, Mem_ID, Date_out),
  ADD KEY Borrowed_Member (Mem_ID),
  ADD KEY Borrowed_Librarian (Lib_ID);

-- Indexes for table `CATEGORY`
ALTER TABLE CATEGORY
  ADD PRIMARY KEY (ID);

-- Indexes for table `HAS`
ALTER TABLE HAS
  ADD PRIMARY KEY (Cat_ID, ISBN),
  ADD KEY Has_Books (ISBN`);

-- Indexes for table `HELD_REQUEST_BY1`
ALTER TABLE HELD_REQUEST_BY1
  ADD PRIMARY KEY (Mem_ID, ISBN),
  ADD KEY Request1_Books (ISBN);

-- Indexes for table `HELD_REQUEST_BY2`
ALTER TABLE HELD_REQUEST_BY2
  ADD PRIMARY KEY (ISBN);
```

```
-- Indexes for table `LIBRARIAN`
ALTER TABLE LIBRARIAN
  ADD PRIMARY KEY (ID);

-- Indexes for table `MEMBER`
ALTER TABLE MEMBER
  ADD PRIMARY KEY (ID);

-- Indexes for table `PUBLISHER`
ALTER TABLE PUBLISHER
  ADD PRIMARY KEY (ID);
-- Indexes for table `SUPPLIER`
ALTER TABLE SUPPLIER
  ADD PRIMARY KEY (ID);

-- Indexes for table `SUPPLY`
ALTER TABLE SUPPLY
  ADD PRIMARY KEY (Sup_ID, ISBN),
  ADD KEY Supply_Books (ISBN);

-- Indexes for table `WRITES`
ALTER TABLE WRITES
  ADD PRIMARY KEY (Aut_ID, ISBN),
  ADD KEY Writes_BOOKS (ISBN);
```

6.3 Create Constraints

```
-- Constraints for table `BOOKS`
ALTER TABLE BOOKS
  ADD CONSTRAINT Books_Publisher FOREIGN KEY (Publisher_ID) REFERENCES
PUBLISHER (ID);

-- Constraints for table `BORROWED_BY`
ALTER TABLE BORROWED_BY
  ADD CONSTRAINT Borrowed_Books FOREIGN KEY (ISBN) REFERENCES BOOKS (ISBN),
  ADD CONSTRAINT Borrowed_Librarian FOREIGN KEY (Lib_ID) REFERENCES LIBRARIAN
(ID),
  ADD CONSTRAINT Borrowed_Member FOREIGN KEY (Mem_ID) REFERENCES MEMBER (ID);

-- Constraints for table `HAS`
ALTER TABLE HAS
  ADD CONSTRAINT Has_Books FOREIGN KEY (ISBN) REFERENCES BOOKS (ISBN),
  ADD CONSTRAINT Has_Category FOREIGN KEY (Cat_ID) REFERENCES CATEGORY (ID);

-- Constraints for table `HELD_REQUEST_BY1`
ALTER TABLE HELD_REQUEST_BY1
  ADD CONSTRAINT Request1_Books FOREIGN KEY (ISBN) REFERENCES BOOKS (ISBN),
  ADD CONSTRAINT Request1_Member FOREIGN KEY (Mem_ID) REFERENCES MEMBER (ID);

-- Constraints for table `HELD_REQUEST_BY2`
ALTER TABLE HELD_REQUEST_BY2
  ADD CONSTRAINT Request2_Books FOREIGN KEY (ISBN) REFERENCES BOOKS (ISBN);
```

```
-- Constraints for table `SUPPLY`
ALTER TABLE SUPPLY
  ADD CONSTRAINT Supply_Books FOREIGN KEY (ISBN) REFERENCES BOOKS (ISBN),
  ADD CONSTRAINT Supply_Supplier FOREIGN KEY (Sup_ID) REFERENCES SUPPLIER
(ID);

-- Constraints for table `WRITES`
ALTER TABLE WRITES
  ADD CONSTRAINT Writes_Author FOREIGN KEY (Aut_ID) REFERENCES AUTHOR (ID),
  ADD CONSTRAINT Writes_BOOKS FOREIGN KEY (ISBN) REFERENCES BOOKS (ISBN);
```

Tables of the public library database:

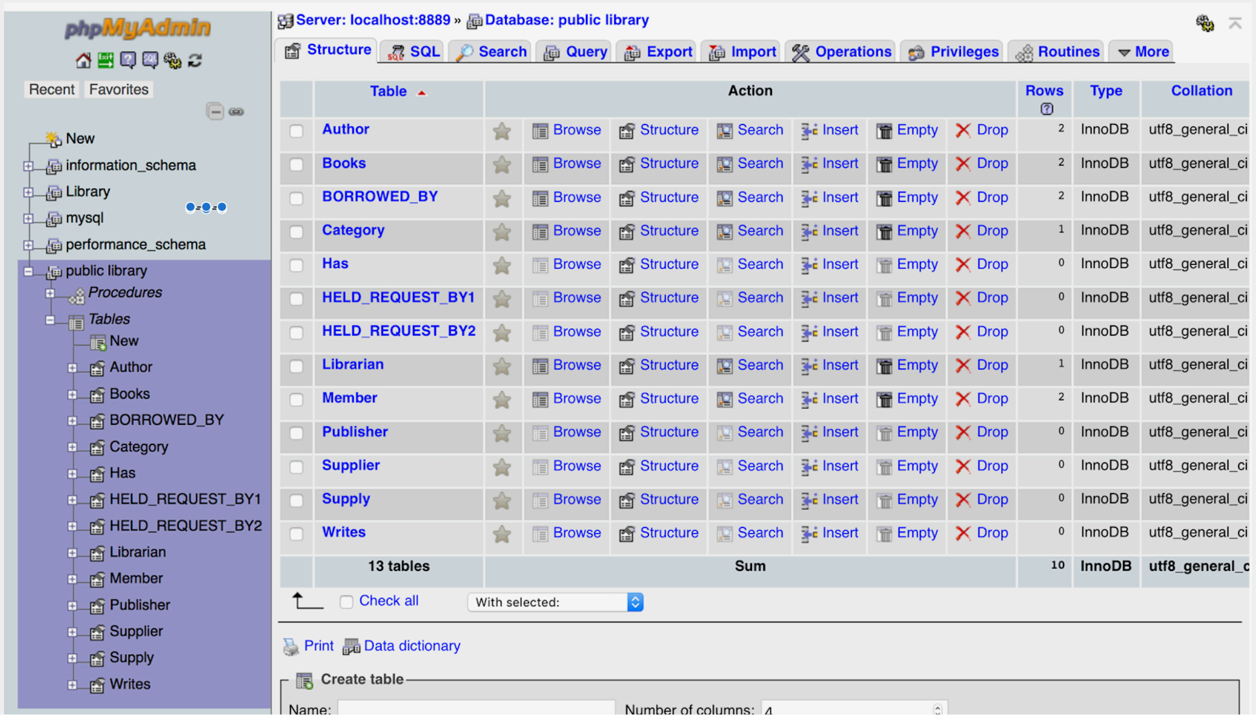


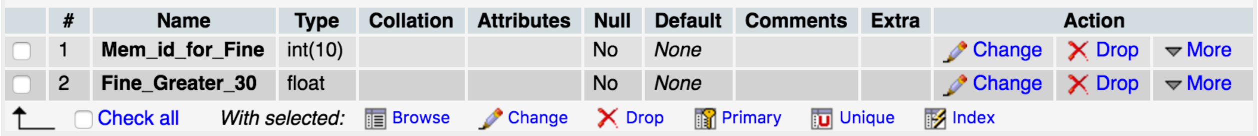
Table	Action							Rows	Type	Collation
Author	☆	Browse	Structure	Search	Insert	Empty	Drop	2	InnoDB	utf8_general_ci
Books	☆	Browse	Structure	Search	Insert	Empty	Drop	2	InnoDB	utf8_general_ci
BORROWED_BY	☆	Browse	Structure	Search	Insert	Empty	Drop	2	InnoDB	utf8_general_ci
Category	☆	Browse	Structure	Search	Insert	Empty	Drop	1	InnoDB	utf8_general_ci
Has	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8_general_ci
HELD_REQUEST_BY1	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8_general_ci
HELD_REQUEST_BY2	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8_general_ci
Librarian	☆	Browse	Structure	Search	Insert	Empty	Drop	1	InnoDB	utf8_general_ci
Member	☆	Browse	Structure	Search	Insert	Empty	Drop	2	InnoDB	utf8_general_ci
Publisher	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8_general_ci
Supplier	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8_general_ci
Supply	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8_general_ci
Writes	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8_general_ci
13 tables	Sum							10	InnoDB	utf8_general_c

7. PL/SQL: Define two Different Stored Procedures and Two Triggers

7.1 Create Triggers

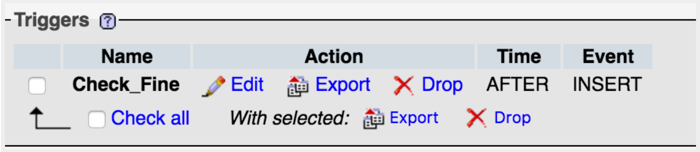
```
TRIGGER1:
We will insert member's id and the fine value into MEMBER_FINE table after
inserting a record into BORROWED_BY table whose Fine > 30.

-- Table structure for table `MEMBER_FINE`
CREATE TABLE MEMBER_FINE (
  Mem_id_for_Fine        int(10)            NOT NULL,
  Fine_Greater_30        float              NOT NULL
);
```



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	**Mem_id_for_Fine**	int(10)			No	*None*			Change	Drop	More
☐	2	**Fine_Greater_30**	float			No	*None*			Change	Drop	More

```
-----------------------------------------------------------------------------
DELIMITER $$
CREATE TRIGGER Check_Fine
      AFTER INSERT ON BORROWED_BY
      FOR EACH ROW
IF NEW.FINE > 30 THEN
      INSERT INTO MEMBER_FINE(Fine_Greater_30, Mem_id_for_Fine)
      VALUES (NEW.Fine, NEW.Mem_ID);
END IF
$$
DELIMITER;
-----------------------------------------------------------------------------
```



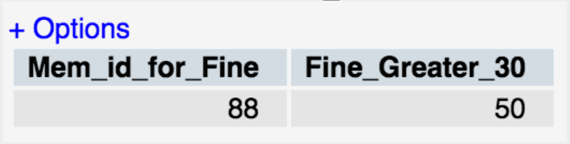
Triggers

	Name	Action			Time	Event
☐	**Check_Fine**	Edit	Export	Drop	AFTER	INSERT

```
-- Insert a new record into BORROWED_BY table
-- The values of Lib_ID, ISBN, and Mem_ID are already in their own tables

INSERT INTO BORROWED_BY (Lib_ID, ISBN, Mem_ID, Date_out, Return_date,
Due_date, Fine)
VALUES ('100000888', '1000000001', '88', '2016-01-01', '2016-01-21', '2016-
06-16', '50');
```

Result of the MEMBER_FINE table:



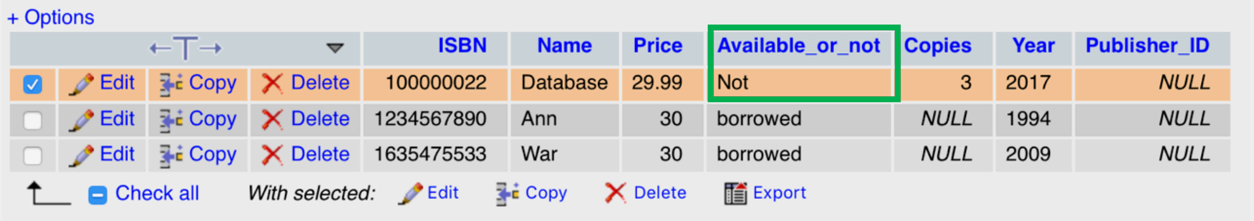
Mem_id_for_Fine	Fine_Greater_30
88	50

TRIGGER2:

We will update the attribute 'available_or_not' in BOOKS table to 'borrowed' after inserting a record into BORROWED_BY whose 'Date_out'!= 0.

```
-----------------------------------------------------------------------------
DELIMITER $$
CREATE TRIGGER borrowed
      AFTER INSERT ON BORROWED_BY
      FOR EACH ROW BEGIN
IF new.Date_out != 0 THEN
      UPDATE BOOKS SET available_or_not = 'borrowed'
      WHERE ISBN = new.ISBN;
END if;
$$
```

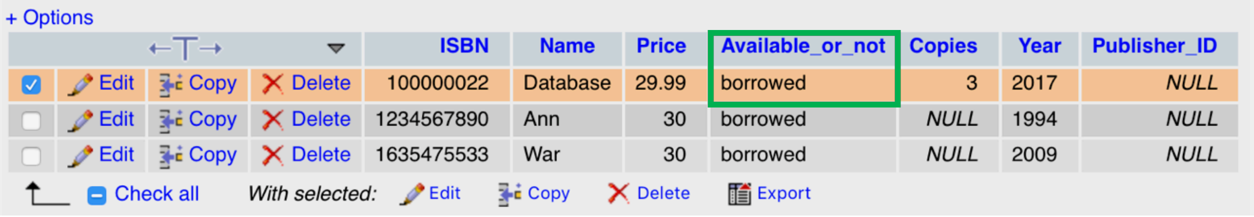
```
DELIMITER ;
------------------------------------------------------------------------------
Original BOOKS table:
```



				ISBN	Name	Price	Available_or_not	Copies	Year	Publisher_ID
✓	Edit	Copy	Delete	100000022	Database	29.99	Not	3	2017	NULL
	Edit	Copy	Delete	1234567890	Ann	30	borrowed	NULL	1994	NULL
	Edit	Copy	Delete	1635475533	War	30	borrowed	NULL	2009	NULL

```
-- Insert a new record into BORROWED_BY table which Date_out != 0
-- The values of Lib_ID, ISBN, and Mem_ID are already in their own tables
INSERT INTO BORROWED_BY (Lib_ID, ISBN, Mem_ID, Date_out, Return_date,
Due_date, Fine) VALUES ('1010111', '100000022', '111', '2017-04-09', NULL,
'2017-05-09', '0');
```

Result of the BOOKS table:

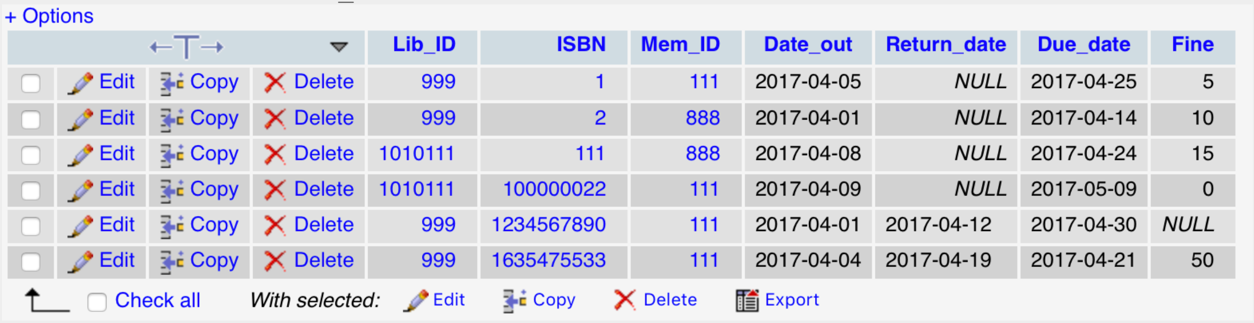


				ISBN	Name	Price	Available_or_not	Copies	Year	Publisher_ID
✓	Edit	Copy	Delete	100000022	Database	29.99	borrowed	3	2017	NULL
	Edit	Copy	Delete	1234567890	Ann	30	borrowed	NULL	1994	NULL
	Edit	Copy	Delete	1635475533	War	30	borrowed	NULL	2009	NULL

7.2 Create Procedure

PROCEDURE1:

```
We select the members who do not return books before the due date (2017-04-
28) from the BORROWED_BY table below:
```



				Lib_ID	ISBN	Mem_ID	Date_out	Return_date	Due_date	Fine
	Edit	Copy	Delete	999	1	111	2017-04-05	NULL	2017-04-25	5
	Edit	Copy	Delete	999	2	888	2017-04-01	NULL	2017-04-14	10
	Edit	Copy	Delete	1010111	111	888	2017-04-08	NULL	2017-04-24	15
	Edit	Copy	Delete	1010111	100000022	111	2017-04-09	NULL	2017-05-09	0
	Edit	Copy	Delete	999	1234567890	111	2017-04-01	2017-04-12	2017-04-30	NULL
	Edit	Copy	Delete	999	1635475533	111	2017-04-04	2017-04-19	2017-04-21	50

```
--------------------------------------------------------------------------
CREATE TABLE BOOKS_DUE (
  BOOK_ID int(13),
  BOOK_NAME varchar(15),
  BOOK_DUE_DATE date,
  MEM_FNAME varchar(15),
  MEM_EMAIL varchar(30),
  BOOK_RETURN_DATE DATE
);
```

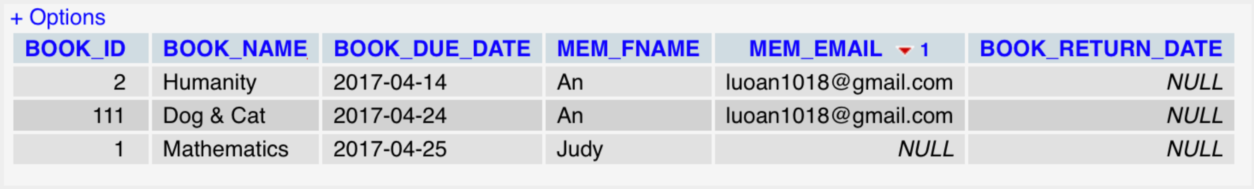
```
DELIMITER //
BEGIN
      DECLARE done INT DEFAULT FALSE;
      DECLARE BOOK_ID INT(13);
      DECLARE BOOK_NAME VARCHAR(15);
      DECLARE BOOK_RETURN_DATE DATE;
      DECLARE BOOK_DUE_DATE DATE;
      DECLARE MEM_FNAME VARCHAR(15);
      DECLARE MEM_EMAIL VARCHAR(30);

      DECLARE BOOKDUE CURSOR FOR
      SELECT BOOKS.ISBN, BOOKS.Name, BORROWED_BY.Return_date,
             BORROWED_BY.Due_date,MEMBER.FNAME, MEMBER.Email
      FROM BOOKS, MEMBER, BORROWED_BY
      WHERE BOOKS.ISBN = BORROWED_BY.ISBN AND MEMBER.ID = BORROWED_BY.MEM_ID;
      DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

      OPEN BOOKDUE;
      READ_LOOP: LOOP
      FETCH BOOKDUE INTO
        BOOK_ID,BOOK_NAME,BOOK_RETURN_DATE,BOOK_DUE_DATE,MEM_FNAME,MEM_EMAIL;
        IF done THEN
            LEAVE READ_LOOP;
        END IF;
        IF BOOK_DUE_DATE < '2017-04-28' AND BOOK_RETURN_DATE IS NULL THEN
            INSERT INTO BOOKS_DUE
                 (BOOK_ID,BOOK_NAME,BOOK_RETURN_DATE,BOOK_DUE_DATE,MEM_FNAME,
                  MEM_EMAIL)
           VALUES(BOOK_ID,BOOK_NAME,BOOK_RETURN_DATE,BOOK_DUE_DATE,MEM_FNAME,
                  MEM_EMAIL);
        END IF;
      END LOOP;
      CLOSE BOOKDUE;
END//
-------------------------------------------------------------------------

CALL BORROW_INFO();
```

Result:

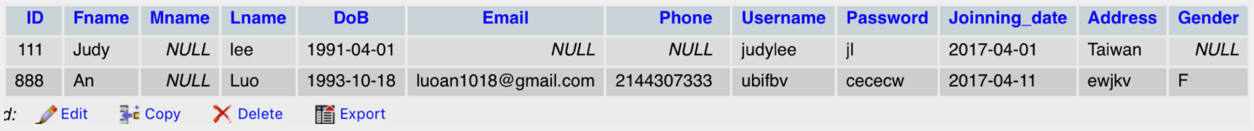


+ Options

BOOK_ID	BOOK_NAME	BOOK_DUE_DATE	MEM_FNAME	MEM_EMAIL ▼ 1	BOOK_RETURN_DATE
2	Humanity	2017-04-14	An	luoan1018@gmail.com	*NULL*
111	Dog & Cat	2017-04-24	An	luoan1018@gmail.com	*NULL*
1	Mathematics	2017-04-25	Judy	*NULL*	*NULL*

PROCEDURE2:

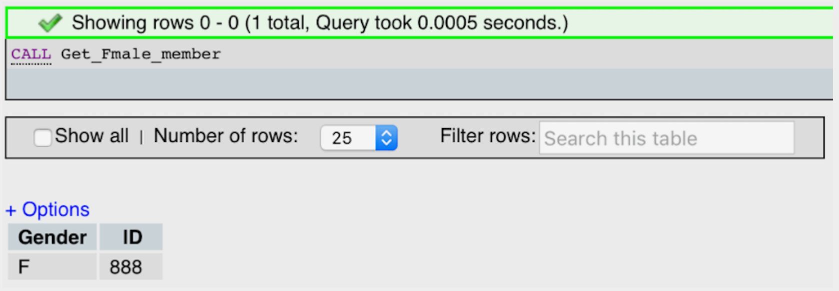
In order to know about the female readers in our library, we select female readers' member id from MEMBER table below:



ID	Fname	Mname	Lname	DoB	Email	Phone	Username	Password	Joinning_date	Address	Gender
111	Judy	NULL	lee	1991-04-01	NULL	NULL	judylee	jl	2017-04-01	Taiwan	NULL
888	An	NULL	Luo	1993-10-18	luoan1018@gmail.com	2144307333	ubifbv	cececw	2017-04-11	ewjkv	F

```
-------------------------------------------------------------------------
DELIMITER //
CREATE PROCEDURE Get_Fmale_member()
BEGIN
  SELECT Gender, ID
  FROM Member
  WHERE Gender='F';
END//
-------------------------------------------------------------------------
CALL Get_Fmale_member();
```

Result:



Showing rows 0 - 0 (1 total, Query took 0.0005 seconds.)

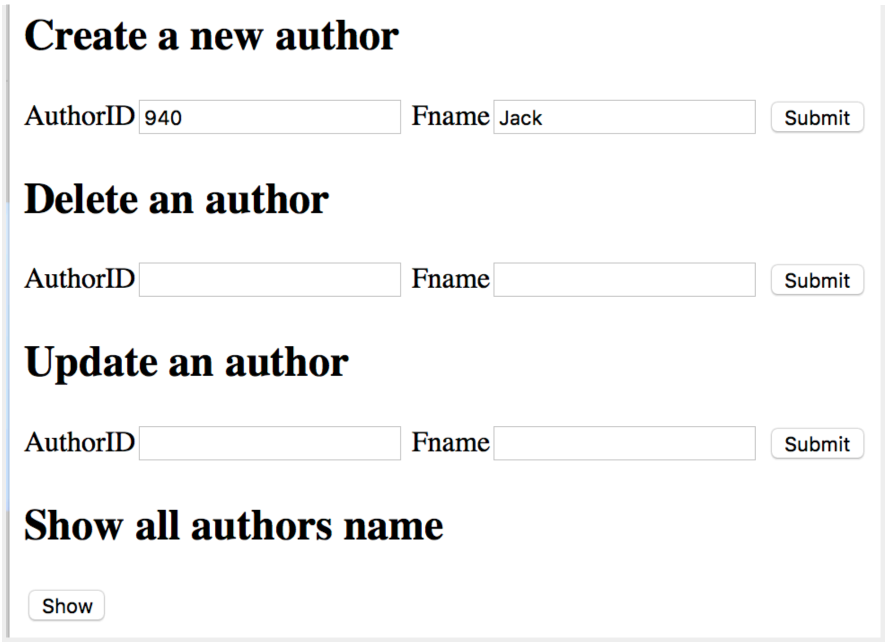
```
CALL Get_Fmale_member
```

Gender	ID
F	888

8. Implement CRUD operations Using PHP

8.1 Create new record in AUTHOR table

```
<html>
<h1> Author</h1>
<?php
$con=mysqli_connect("localhost","root","root","public library");
$ID = $_POST["ID"];
$Fname = $_POST["Fname"];
$sql = "INSERT INTO Author(ID, Fname) VALUES ('{$ID}','{$Fname}')";
echo "Insert a Author ID='{$ID}' Fname='{$Fname}'";
mysqli_query($con,$sql);
mysqli_close($con);
?>
<?
$con1=mysqli_connect("localhost","root","root","public library");
$result = mysqli_query($con1,"SELECT * FROM AUTHOR");
echo "<table border='1'>
<tr>
<th>Author ID</th>
<th>First Name</th>
</tr>";
while($row = mysqli_fetch_array($result))
{
echo "<tr>";
echo "<td>" . $row['ID'] . "</td>";
echo "<td>" . $row['Fname'] . "</td>";
echo "</tr>";
}
echo "</table>";
mysqli_query($con1,$sql);
mysqli_close($con1);
?>
</html>
```



Create a new author

AuthorID 940 Fname Jack Submit

Delete an author

AuthorID Fname Submit

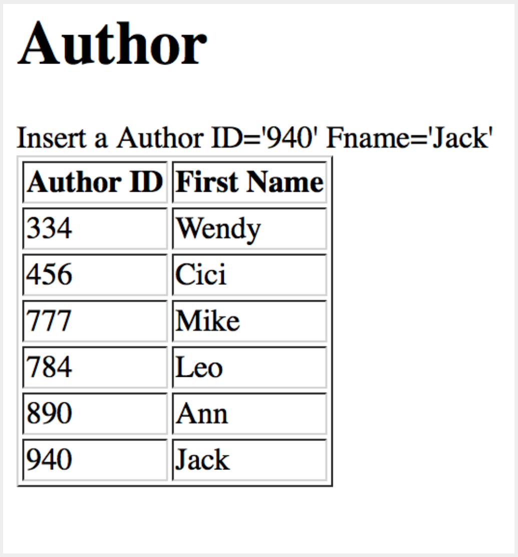
Update an author

AuthorID Fname Submit

Show all authors name

Show

```
Result:
```



Author

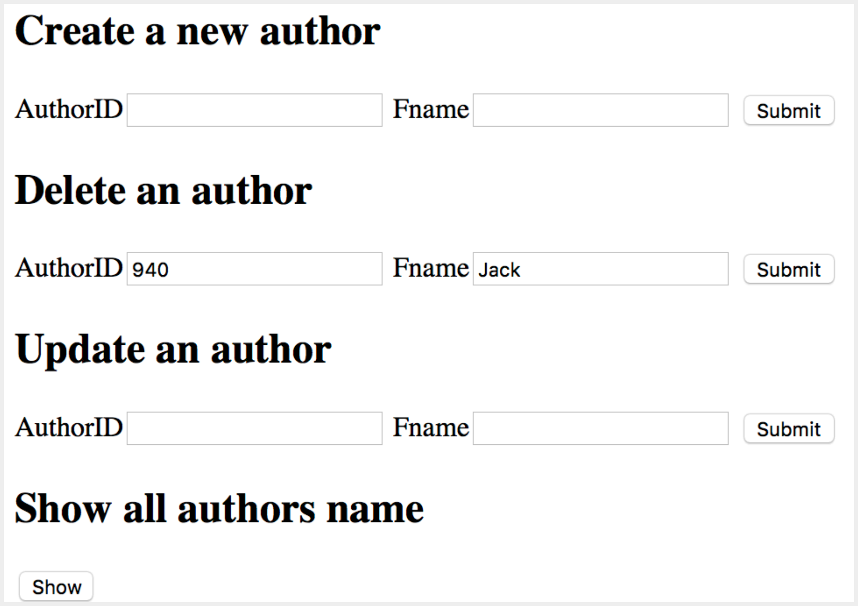
Insert a Author ID='940' Fname='Jack'

Author ID	First Name
334	Wendy
456	Cici
777	Mike
784	Leo
890	Ann
940	Jack

8.2 Delete a record in AUTHOR Table

```
<html>
<h1> Author</h1>
<?php
$ID = $_POST["ID"];
$Fname = $_POST["Fname"];
$con=mysqli_connect("localhost","root","root","public library");
$sql = "DELETE from Author WHERE ID='{$ID}' AND Fname='{$Fname}'";
echo "Delete a Author ID='{$ID}' Fname='{$Fname}'";
mysqli_query($con,$sql);
```

```
mysqli_close($con);
?>
<?
$con1=mysqli_connect("localhost","root","root","public library");
$result = mysqli_query($con1,"SELECT * FROM AUTHOR");
echo "<table border='1'>
<tr>
<th>Author ID</th>
<th>First Name</th>
</tr>";
while($row = mysqli_fetch_array($result))
{
echo "<tr>";
echo "<td>" . $row['ID'] . "</td>";
echo "<td>" . $row['Fname'] . "</td>";
echo "</tr>";
}
echo "</table>";
mysqli_query($con1,$sql);
mysqli_close($con1);
?>
</html>
```



Create a new author

AuthorID Fname

Delete an author

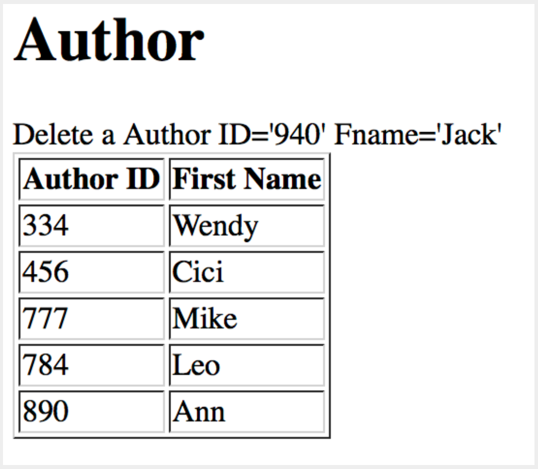
AuthorID 940 Fname Jack

Update an author

AuthorID Fname

Show all authors name

Result:



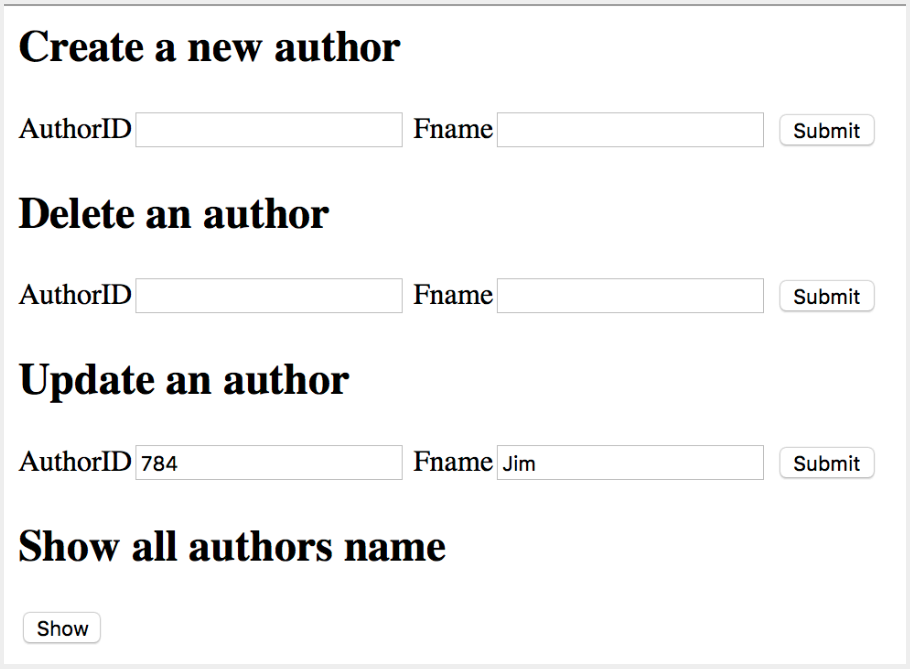
Author

Delete a Author ID='940' Fname='Jack'

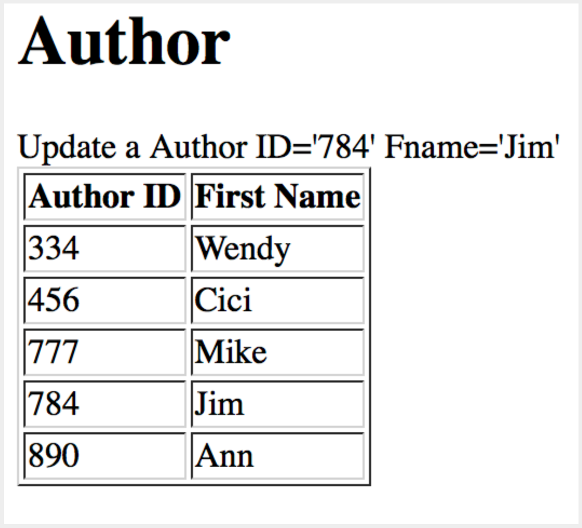
Author ID	First Name
334	Wendy
456	Cici
777	Mike
784	Leo
890	Ann

8.3 Update AUTHOR Table

```
<html>
<h1> Author</h1>
<?php
$ID = $_POST["ID"];
$Fname = $_POST["Fname"];
$con=mysqli_connect("localhost","root","root","public library");
$sql = "UPDATE Author SET Fname = '{$Fname}' WHERE ID = '{$ID}'";
echo "Update a Author ID='{$ID}' Fname='{$Fname}'";
mysqli_query($con,$sql);
mysqli_close($con);
?>
<?
$con1=mysqli_connect("localhost","root","root","public library");
$result = mysqli_query($con1,"SELECT * FROM AUTHOR");
echo "<table border='1'>
<tr>
<th>Author ID</th>
<th>First Name</th>
</tr>";
while($row = mysqli_fetch_array($result))
{
echo "<tr>";
echo "<td>" . $row['ID'] . "</td>";
echo "<td>" . $row['Fname'] . "</td>";
echo "</tr>";
}
echo "</table>";
mysqli_query($con1,$sql);
mysqli_close($con1);
?>
</html>
```



```
Result:
```



Author

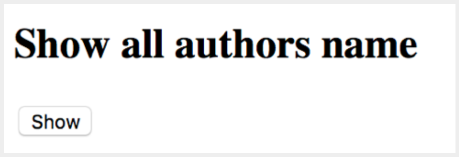
Update a Author ID='784' Fname='Jim'

Author ID	First Name
334	Wendy
456	Cici
777	Mike
784	Jim
890	Ann

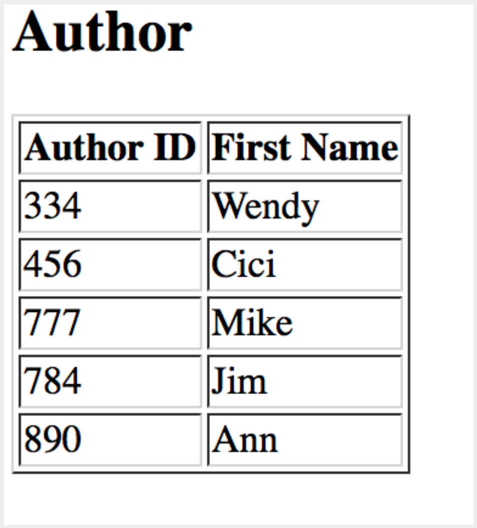
8.4 Read the records in AUTHOR Table

```
<html>
<h2> Author</h2>
<?php
$Fname = $_GET["Fname"];
$con=mysqli_connect("localhost","root","root","public library");
$sql = "SELECT ID, Fname, Lname from Author";
```

```
$result = $con -> query ($sql);
echo "<table border='1'>
<tr>
<th>Author ID</th>
<th>First Name</th>
</tr>";
while($row = mysqli_fetch_array($result))
{
echo "<tr>";
echo "<td>" . $row['ID'] . "</td>";
echo "<td>" . $row['Fname'] . "</td>";
echo "</tr>";
}
echo "</table>";
mysqli_query($con,$sql);
mysqli_close($con);
?>
</html>
```


Show all authors name

Result:


Author

Author ID	First Name
334	Wendy
456	Cici
777	Mike
784	Jim
890	Ann

8.5 The HTML file using to implement CRUD operations

```
<html>
<form action="createauthor.php" method="POST">
<h2> Create a new author </h2>
      AuthorID<input type="text" name="ID">
      Fname<input type="text" name="Fname">
<input type="submit">
</form>
```

```
<form action="deleteauthor.php" method="POST">
<h2> Delete an author </h2>
      AuthorID<input type="text" name="ID">
      Fname<input type="text" name="Fname">
<input type="submit">
</form>


<form action="updateauthor.php" method="POST">
<h2> Update an author </h2>
      AuthorID<input type="text" name="ID">
      Fname<input type="text" name="Fname">
<input type="submit">
</form>


<form action="readauthor.php" method="GET">
<h2> Show all authors name </h2>
<button  name= "Fname "type="submit">Show</button>
</form>
</html>
```