DBMS - Mini Project

Library Management System

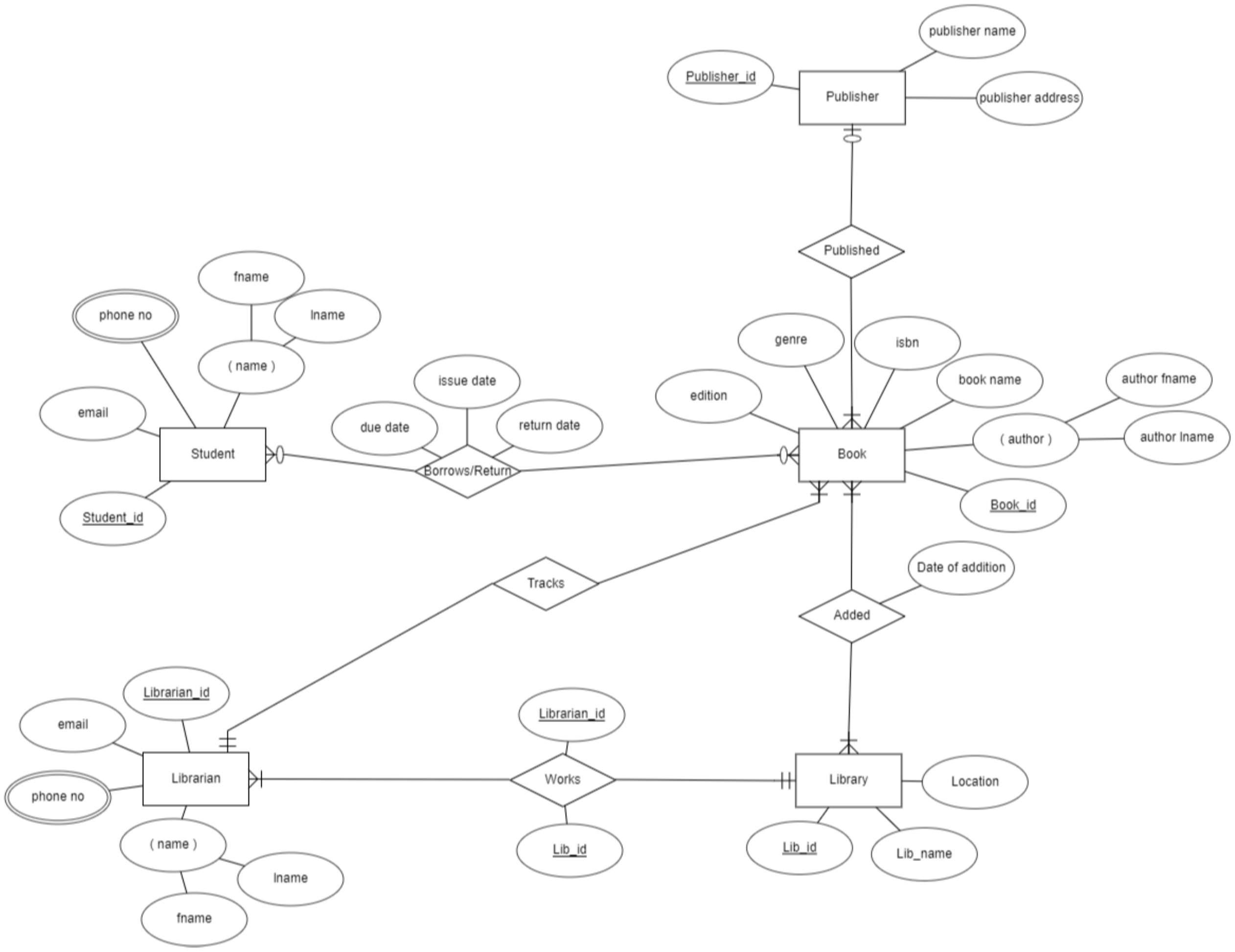
Submitted By:
Name: Abishek Deivam
SRN: PES1UG20CS012
V Semester Section: A

Short Description and Scope of the Project

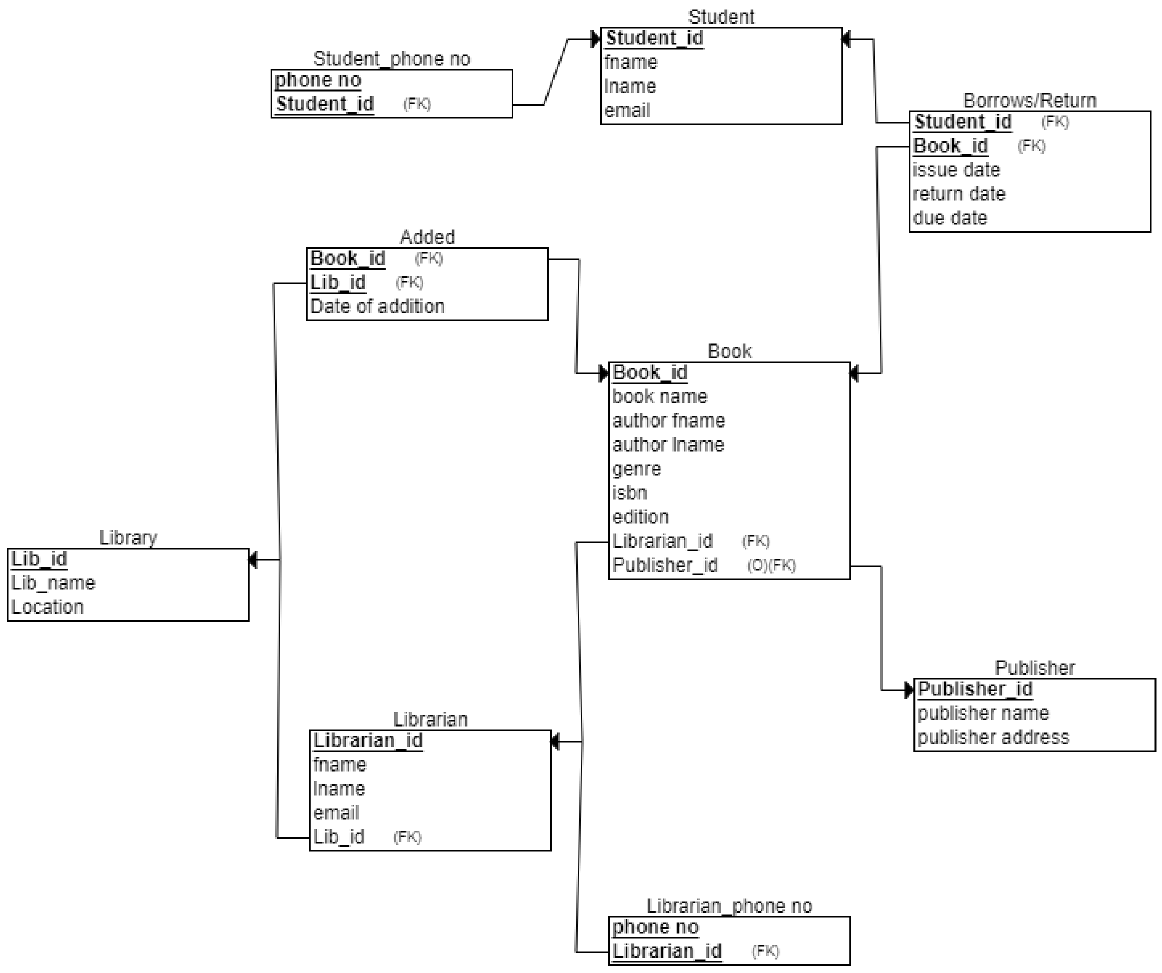
This project is about a library management system , for borrowing and returning books by students. A student can borrow and return a book , while a record is maintained on when he/she has borrowed it and when he/she has returned it. There are librarians to facilitate these actions and maintain books.There are multiple libraries to enhance the scope of the project to multiple libraries.

A user can choose to perform crud operations or execute manual queries from the frontend.

ER Diagram


publisher name
Publisher_id
Publisher
publisher address
Published
fname
phone no
lname
(name)
genre
isbn
issue date
edition
book name
author fname
email
due date
return date
Student
Borrows/Return
Book
(author)
author lname
Student_id
Book_id
Date of addition
Tracks
Added
Librarian_id
email
Librarian_id
Librarian
Works
Library
Location
phone no
(name)
Lib_id
Lib_id
Lib_name
lname
fname

Relational Schema



DDL statements - Building the database

```
CREATE TABLE Student
(
  fname VARCHAR(20) NOT NULL,
  lname VARCHAR(20) NOT NULL,
  email VARCHAR(30) NOT NULL,
  Student_id VARCHAR(20) NOT NULL UNIQUE,
  PRIMARY KEY (Student_id)
);

CREATE TABLE Library
(
  Lib_id VARCHAR(20) NOT NULL UNIQUE,
  Lib_name VARCHAR(20) NOT NULL,
  Location VARCHAR(20) NOT NULL,
  PRIMARY KEY (Lib_id)
);

CREATE TABLE Librarian
(
  Librarian_id VARCHAR(20) NOT NULL UNIQUE,
  fname VARCHAR(20) NOT NULL,
  lname VARCHAR(20) NOT NULL,
  email VARCHAR(30) NOT NULL,
  Lib_id VARCHAR(20) NOT NULL,
  PRIMARY KEY (Librarian_id),
  FOREIGN KEY (Lib_id) REFERENCES Library(Lib_id)
  ON DELETE CASCADE
);

CREATE TABLE Publisher
(
  Publisher_id VARCHAR(20) NOT NULL UNIQUE,
  publisher_name VARCHAR(20) NOT NULL,
  publisher_address VARCHAR(40) NOT NULL,
  PRIMARY KEY (Publisher_id)
);

CREATE TABLE Student_phone_no
(
  phone_no NUMERIC(10) NOT NULL,
  Student_id VARCHAR(20) NOT NULL,
  PRIMARY KEY (phone_no, Student_id),
  FOREIGN KEY (Student_id) REFERENCES Student(Student_id)
  ON DELETE CASCADE
);

CREATE TABLE Librarian_phone_no
(
  phone_no NUMERIC(10) NOT NULL,
  Librarian_id VARCHAR(20) NOT NULL UNIQUE,
  PRIMARY KEY (phone_no, Librarian_id),
  FOREIGN KEY (Librarian_id) REFERENCES Librarian(Librarian_id)
  ON DELETE CASCADE
);

CREATE TABLE Book
(
  book_name VARCHAR(40) NOT NULL,
  Book_id VARCHAR(20) NOT NULL UNIQUE,
  author_fname VARCHAR(20) NOT NULL,
  author_lname VARCHAR(20) NOT NULL,
  genre VARCHAR(20) NOT NULL,
  isbn VARCHAR(20) NOT NULL,
  edition INT NOT NULL,
  Librarian_id VARCHAR(20),
  Publisher_id VARCHAR(20),
  PRIMARY KEY (Book_id),
  FOREIGN KEY (Librarian_id) REFERENCES Librarian(Librarian_id)
  ON DELETE SET NULL,
  FOREIGN KEY (Publisher_id) REFERENCES Publisher(Publisher_id)
  ON DELETE SET NULL
```

```
);

CREATE TABLE Added
(
  Date_of_addition DATE NOT NULL,
  Book_id VARCHAR(20) NOT NULL UNIQUE,
  Lib_id VARCHAR(20),
  PRIMARY KEY (Book_id, Lib_id),
  FOREIGN KEY (Book_id) REFERENCES Book(Book_id)
  ON DELETE CASCADE,
  FOREIGN KEY (Lib_id) REFERENCES Library(Lib_id)
  ON DELETE CASCADE
);

CREATE TABLE Borrows_Return
(
  issue_date DATE NOT NULL,
  return_date DATE ,
  due_date DATE NOT NULL,
  Student_id VARCHAR(20),
  Book_id VARCHAR(20),
  PRIMARY KEY (Student_id, Book_id),
  FOREIGN KEY (Student_id) REFERENCES Student(Student_id)
  ON DELETE CASCADE,
  FOREIGN KEY (Book_id) REFERENCES Book(Book_id)
  ON DELETE CASCADE
);
```

Populating the Database

```
INSERT INTO library VALUES
('LIB001','Library 1','Palm street'),
('LIB002','Library 2','Star street'),
('LIB003','Library 3','Central street'),
('LIB004','Library 4','Metro street'),
('LIB005','Library 5','Park street');

INSERT INTO publisher VALUES
("PUBL_001","Harper Collins","New York , United States"),
("PUBL_002","Simon & Schuster","New York , United States"),
("PUBL_003","Macmillan","London , United Kingdom"),
("PUBL_004","Hachette","Paris , France"),
("PUBL_005","Penguin Random House","New York , United States");

INSERT INTO student VALUES
("Philip","Mckinley","Philip.Mckinley@gmail.com","STD_001"),
("Lucilio","Bonney","Lucilio.Bonney@gmail.com","STD_002"),
("Louise","Triggs","Louise.Triggs@gmail.com","STD_003"),
("Celyn","Panza","Celyn.Panza@gmail.com","STD_004"),
("Brice","Adair","Brice.Adair@gmail.com","STD_005"),
("Blanca","Lim","Blanca.Lim@gmail.com","STD_006"),
("Nia","Beck","Nia.Beck@gmail.com","STD_007"),
("Davey","Blue","Davey.Blue@gmail.com","STD_008"),
("Larisa","Troy","Larisa.Troy@gmail.com","STD_009"),
("Rajesh","Sands","Rajesh.Sands@gmail.com","STD_0010");

INSERT INTO librarian VALUES
("LR_001","Sheila","Joel","Sheila.Joel@gmail.com","LIB001"),
("LR_002","Roshan","Shiva","Roshan.Shiva@gmail.com","LIB001"),
("LR_003","Ganesh","Malcom","Ganesh.Malcom@gmail.com","LIB002"),
("LR_004","Daria","Marina","Daria.Marina@gmail.com","LIB004"),
("LR_005","Eino","Ervin","Eino.Ervin@gmail.com","LIB004");

INSERT INTO book VALUES
```

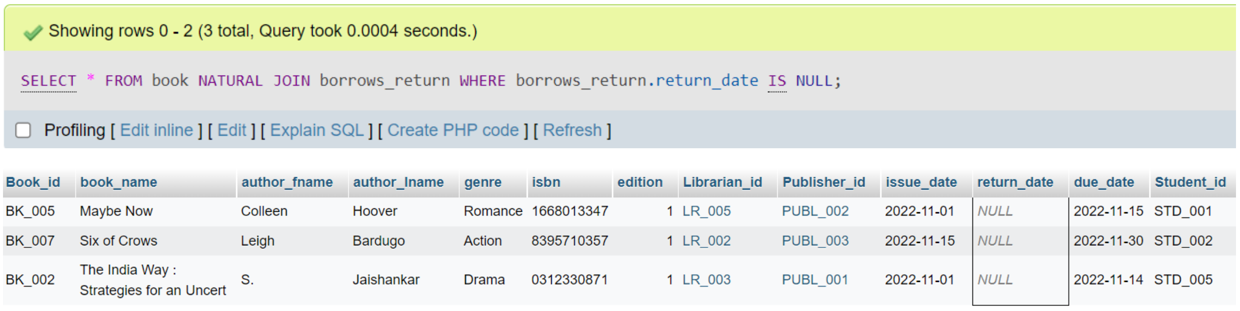
```
("And Then There Were
None","BK_001","Agatha","Christie","Mystery","0312330871",1,"LR_001","PUBL_0
01"),
("The India Way : Strategies for an Uncertain
World","BK_002","S.","Jaishankar","Drama","0312330871",1,"LR_003","PUBL_001"
),
("Red
Queen","BK_003","Victoria","Aveyard","Fantasy","006231064X",2,"LR_004","PUBL
_001"),
("City of
Bones","BK_004","Cassandra","Claire","Fantasy","1442472065",1,"LR_002","PUBL
_002"),
("Maybe
Now","BK_005","Colleen","Hoover","Romance","1668013347",1,"LR_005","PUBL_0
02"),
("Chainsaw
Man","BK_006","Tatsuki","Fujimoto","Comic","1974709930",1,"LR_003","PUBL_002
"),
("Six of
Crows","BK_007","Leigh","Bardugo","Action","8395710357",1,"LR_002","PUBL_00
3"),
("The Fall of Boris Johnson: The Full Story
","BK_008","Sebastian","Payne","Biography","4259235733",1,"LR_001","PUBL_003
"),
("The Hitchhiker's Guide to the
Galaxy","BK_009","Douglas","Adams","Humour","1248703523",1,"LR_005","PUBL_
003"),
("I Wouldn't Do That If I Were Me: Modern Blunders and Modest
Triumphs","BK_0010","Jason","Gay","Humour","9756327323",1,"LR_001","PUBL_0
04"),
("Limca Book of Records 2020–22
","BK_0011","Hachette","India","Educational","7325225230",1,"LR_002","PUBL_00
4"),
("The Boys From
Biloxi","BK_0012","John","Grisham","Thriller","0812487643",1,"LR_004","PUBL_00
4"),
("What's for Dessert
","BK_0013","Claire","Saffitz","Food","8452301223",1,"LR_003","PUBL_005"),
("Distant
Thunder","BK_0014","Stuart","Woods","Thriller","9021392024",1,"LR_005","PUBL_
005"),
("Peril in
Paris","BK_0015","Rhys","Bowen","Mystery","1023122944",1,"LR_004","PUBL_005
");

INSERT INTO borrows_return VALUES
("2022-11-01",null,"2022-11-15","STD_001","BK_005"),
("2022-10-28","2022-11-01","2022-11-05","STD_002","BK_0010"),
("2022-10-14","2022-11-05","2022-10-28","STD_003","BK_008"),
("2022-11-15",null,"2022-11-30","STD_002","BK_007"),
("2022-11-01",null,"2022-11-14","STD_005","BK_002");

INSERT INTO added VALUES
("2022-09-14","BK_001","LIB001"),
("2022-10-15","BK_002","LIB002"),
("2022-08-16","BK_003","LIB004"),
("2022-07-17","BK_004","LIB005"),
("2022-09-18","BK_005","LIB002"),
("2022-06-19","BK_006","LIB003"),
("2022-01-20","BK_007","LIB005"),
("2022-04-21","BK_008","LIB001"),
("2022-05-22","BK_009","LIB003"),
("2022-10-23","BK_0010","LIB002"),
("2022-10-12","BK_0011","LIB004"),
("2022-09-15","BK_0012","LIB004"),
("2022-11-19","BK_0013","LIB002"),
("2022-04-29","BK_0014","LIB001"),
("2022-02-15","BK_0015","LIB001");

INSERT INTO librarian_phone_no VALUES
(8684596278,"LR_001"),
(9240943280,"LR_002"),
(0872854029,"LR_003"),
(7879032164,"LR_004"),
(7601156977,"LR_005");
```

```
INSERT INTO student_phone_no VALUES
(3485916693 ,"STD_001"),
(4442157181 ,"STD_002"),
(6386298426 ,"STD_003"),
(8986481841 ,"STD_004"),
(1933145623 ,"STD_005"),
(9466404558 ,"STD_006"),
(2033490286 ,"STD_007"),
(4141868514 ,"STD_008"),
(2869097310 ,"STD_009"),
(4857721399 ,"STD_0010");
```

Join Queries

1. Show which books have currently been borrowed from the library

   SELECT * FROM book NATURAL JOIN borrows_return WHERE borrows_return.return_date IS NULL;



Showing rows 0 - 2 (3 total, Query took 0.0004 seconds.)

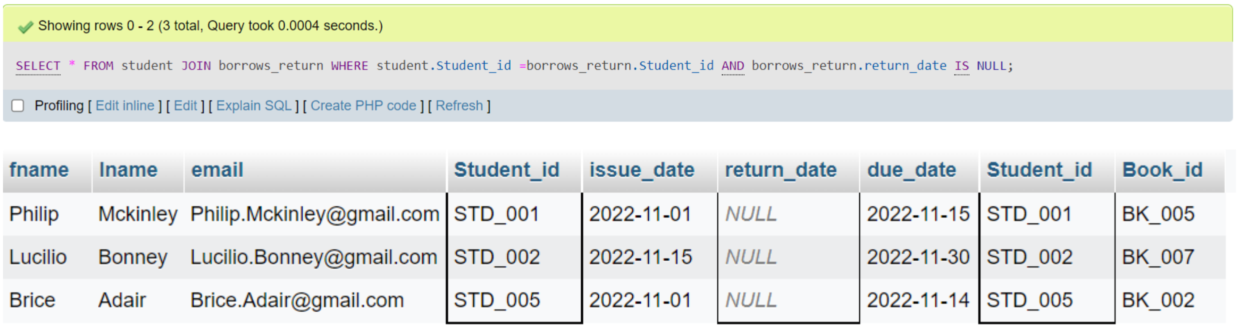
```
SELECT * FROM book NATURAL JOIN borrows_return WHERE borrows_return.return_date IS NULL;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Book_id	book_name	author_fname	author_lname	genre	isbn	edition	Librarian_id	Publisher_id	issue_date	return_date	due_date	Student_id
BK_005	Maybe Now	Colleen	Hoover	Romance	1668013347	1	LR_005	PUBL_002	2022-11-01	NULL	2022-11-15	STD_001
BK_007	Six of Crows	Leigh	Bardugo	Action	8395710357	1	LR_002	PUBL_003	2022-11-15	NULL	2022-11-30	STD_002
BK_002	The India Way : Strategies for an Uncert	S.	Jaishankar	Drama	0312330871	1	LR_003	PUBL_001	2022-11-01	NULL	2022-11-14	STD_005

2. Show which students currently have library books

   SELECT * FROM student JOIN borrows_return WHERE student.Student_id =borrows_return.Student_id AND borrows_return.return_date IS NULL;



Showing rows 0 - 2 (3 total, Query took 0.0004 seconds.)

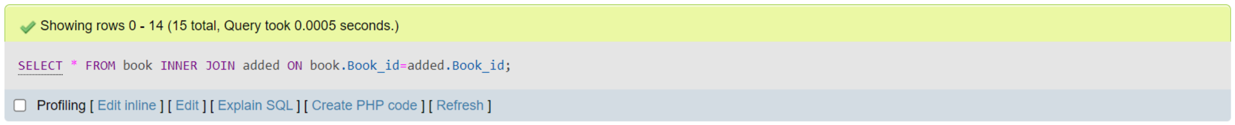
```
SELECT * FROM student JOIN borrows_return WHERE student.Student_id =borrows_return.Student_id AND borrows_return.return_date IS NULL;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

fname	lname	email	Student_id	issue_date	return_date	due_date	Student_id	Book_id
Philip	Mckinley	Philip.Mckinley@gmail.com	STD_001	2022-11-01	NULL	2022-11-15	STD_001	BK_005
Lucilio	Bonney	Lucilio.Bonney@gmail.com	STD_002	2022-11-15	NULL	2022-11-30	STD_002	BK_007
Brice	Adair	Brice.Adair@gmail.com	STD_005	2022-11-01	NULL	2022-11-14	STD_005	BK_002

3. Show which library has which book

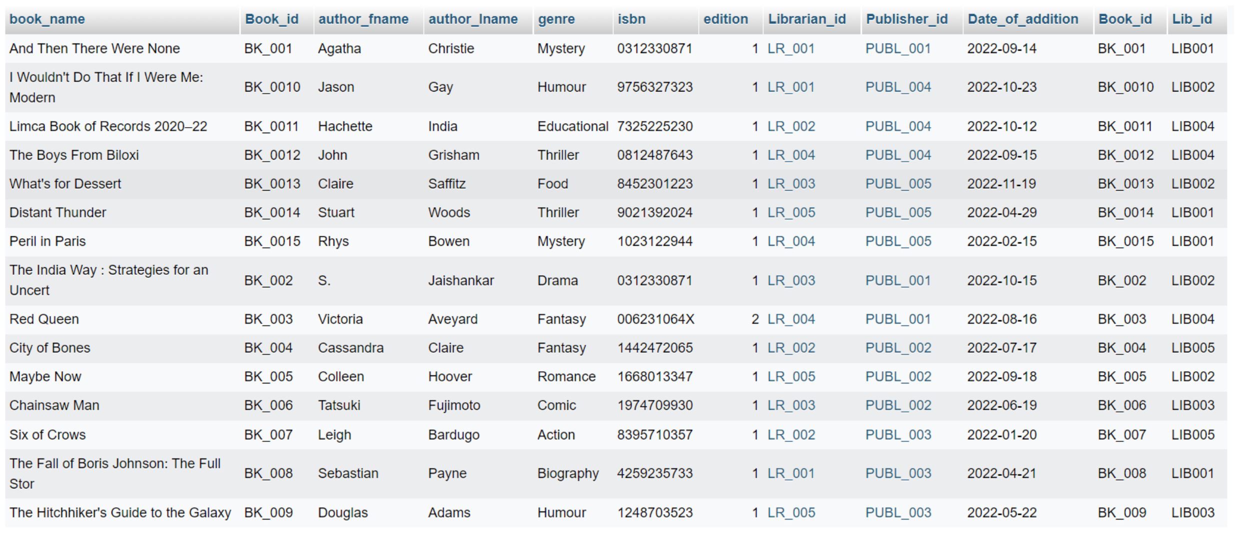
   SELECT * FROM book INNER JOIN added ON book.Book_id=added.Book_id;



Showing rows 0 - 14 (15 total, Query took 0.0005 seconds.)

```
SELECT * FROM book INNER JOIN added ON book.Book_id=added.Book_id;
```

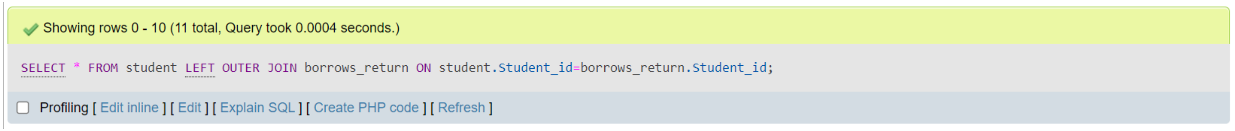
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]



book_name	Book_id	author_fname	author_lname	genre	isbn	edition	Librarian_id	Publisher_id	Date_of_addition	Book_id	Lib_id
And Then There Were None	BK_001	Agatha	Christie	Mystery	0312330871	1	LR_001	PUBL_001	2022-09-14	BK_001	LIB001
I Wouldn't Do That If I Were Me: Modern	BK_0010	Jason	Gay	Humour	9756327323	1	LR_001	PUBL_004	2022-10-23	BK_0010	LIB002
Limca Book of Records 2020–22	BK_0011	Hachette	India	Educational	7325225230	1	LR_002	PUBL_004	2022-10-12	BK_0011	LIB004
The Boys From Biloxi	BK_0012	John	Grisham	Thriller	0812487643	1	LR_004	PUBL_004	2022-09-15	BK_0012	LIB004
What's for Dessert	BK_0013	Claire	Saffitz	Food	8452301223	1	LR_003	PUBL_005	2022-11-19	BK_0013	LIB002
Distant Thunder	BK_0014	Stuart	Woods	Thriller	9021392024	1	LR_005	PUBL_005	2022-04-29	BK_0014	LIB001
Peril in Paris	BK_0015	Rhys	Bowen	Mystery	1023122944	1	LR_004	PUBL_005	2022-02-15	BK_0015	LIB001
The India Way : Strategies for an Uncert	BK_002	S.	Jaishankar	Drama	0312330871	1	LR_003	PUBL_001	2022-10-15	BK_002	LIB002
Red Queen	BK_003	Victoria	Aveyard	Fantasy	006231064X	2	LR_004	PUBL_001	2022-08-16	BK_003	LIB004
City of Bones	BK_004	Cassandra	Claire	Fantasy	1442472065	1	LR_002	PUBL_002	2022-07-17	BK_004	LIB005
Maybe Now	BK_005	Colleen	Hoover	Romance	1668013347	1	LR_005	PUBL_002	2022-09-18	BK_005	LIB002
Chainsaw Man	BK_006	Tatsuki	Fujimoto	Comic	1974709930	1	LR_003	PUBL_002	2022-06-19	BK_006	LIB003
Six of Crows	BK_007	Leigh	Bardugo	Action	8395710357	1	LR_002	PUBL_003	2022-01-20	BK_007	LIB005
The Fall of Boris Johnson: The Full Stor	BK_008	Sebastian	Payne	Biography	4259235733	1	LR_001	PUBL_003	2022-04-21	BK_008	LIB001
The Hitchhiker's Guide to the Galaxy	BK_009	Douglas	Adams	Humour	1248703523	1	LR_005	PUBL_003	2022-05-22	BK_009	LIB003

4. Show which students have borrowed books and which have not

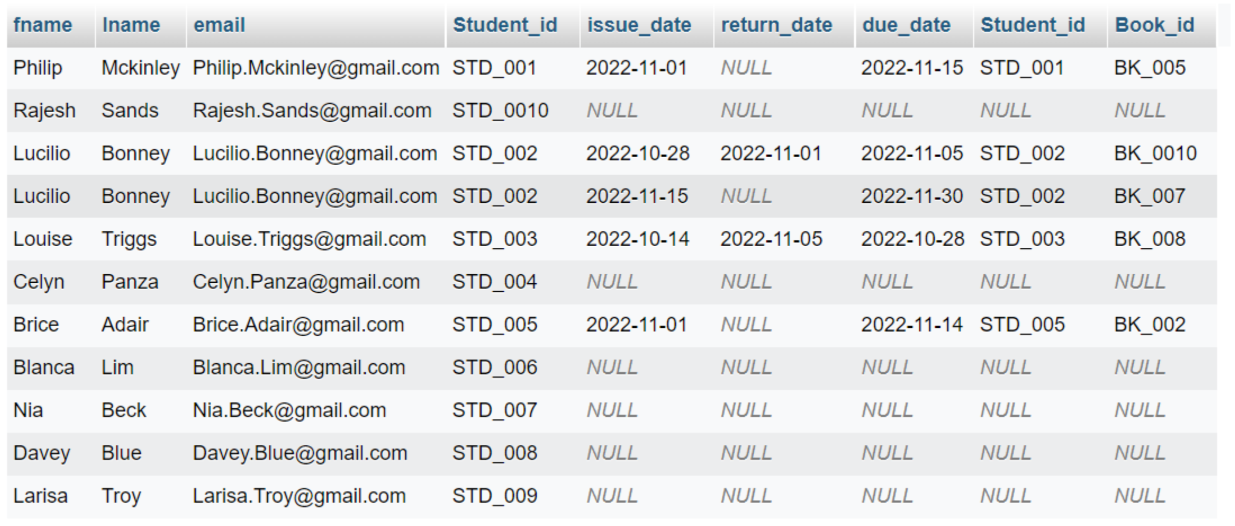
SELECT * FROM student LEFT OUTER JOIN borrows_return ON student.Student_id=borrows_return.Student_id;



✓ Showing rows 0 - 10 (11 total, Query took 0.0004 seconds.)

```
SELECT * FROM student LEFT OUTER JOIN borrows_return ON student.Student_id=borrows_return.Student_id;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

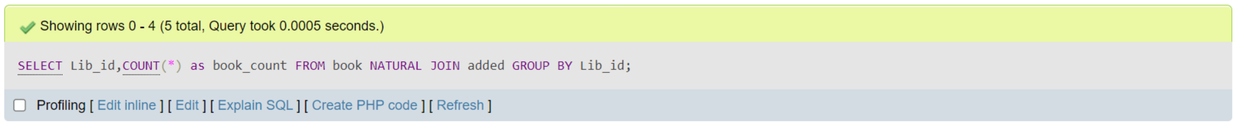


fname	lname	email	Student_id	issue_date	return_date	due_date	Student_id	Book_id
Philip	Mckinley	Philip.Mckinley@gmail.com	STD_001	2022-11-01	NULL	2022-11-15	STD_001	BK_005
Rajesh	Sands	Rajesh.Sands@gmail.com	STD_0010	NULL	NULL	NULL	NULL	NULL
Lucilio	Bonney	Lucilio.Bonney@gmail.com	STD_002	2022-10-28	2022-11-01	2022-11-05	STD_002	BK_0010
Lucilio	Bonney	Lucilio.Bonney@gmail.com	STD_002	2022-11-15	NULL	2022-11-30	STD_002	BK_007
Louise	Triggs	Louise.Triggs@gmail.com	STD_003	2022-10-14	2022-11-05	2022-10-28	STD_003	BK_008
Celyn	Panza	Celyn.Panza@gmail.com	STD_004	NULL	NULL	NULL	NULL	NULL
Brice	Adair	Brice.Adair@gmail.com	STD_005	2022-11-01	NULL	2022-11-14	STD_005	BK_002
Blanca	Lim	Blanca.Lim@gmail.com	STD_006	NULL	NULL	NULL	NULL	NULL
Nia	Beck	Nia.Beck@gmail.com	STD_007	NULL	NULL	NULL	NULL	NULL
Davey	Blue	Davey.Blue@gmail.com	STD_008	NULL	NULL	NULL	NULL	NULL
Larisa	Troy	Larisa.Troy@gmail.com	STD_009	NULL	NULL	NULL	NULL	NULL

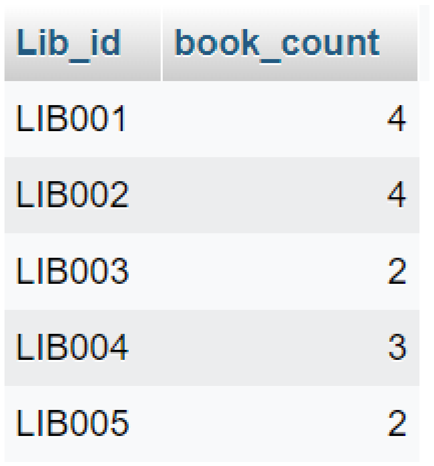
Aggregate Functions

1. Number of books library wise

   SELECT Lib_id,COUNT(*) as book_count FROM book NATURAL JOIN added GROUP BY Lib_id;



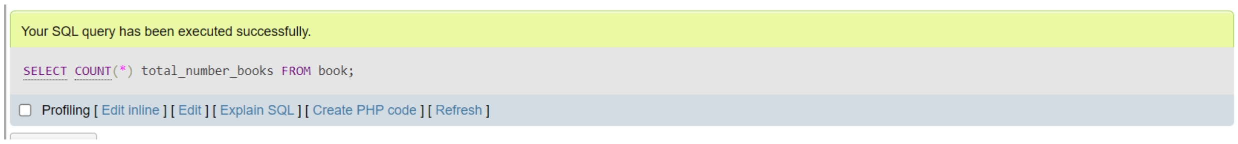
```
Showing rows 0 - 4 (5 total, Query took 0.0005 seconds.)
SELECT Lib_id,COUNT(*) as book_count FROM book NATURAL JOIN added GROUP BY Lib_id;
Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]
```



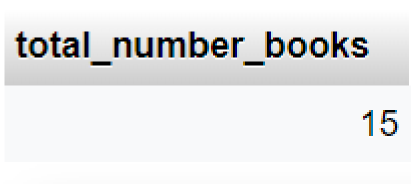
Lib_id	book_count
LIB001	4
LIB002	4
LIB003	2
LIB004	3
LIB005	2

2. Total number of books

   SELECT COUNT(*) total_number_books FROM book;



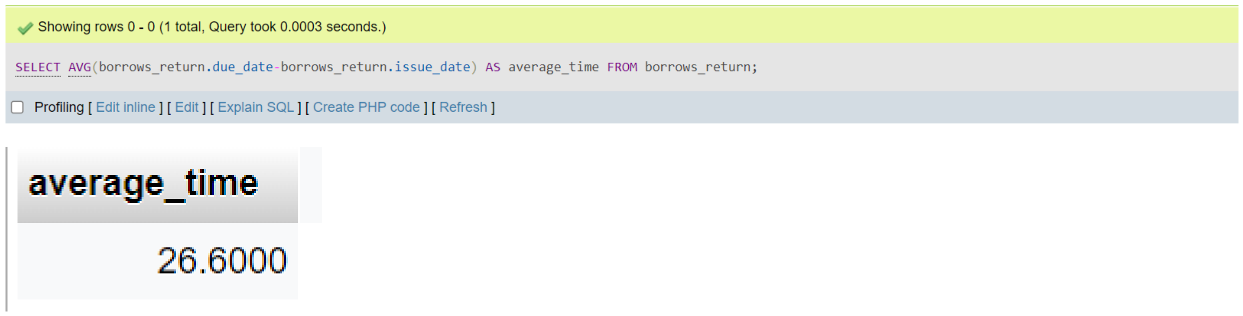
```
Your SQL query has been executed successfully.
SELECT COUNT(*) total_number_books FROM book;
Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]
```



total_number_books
15

3. Average duration of books getting borrowed

SELECT AVG(borrows_return.due_date-borrows_return.issue_date) AS average_time FROM borrows_return;

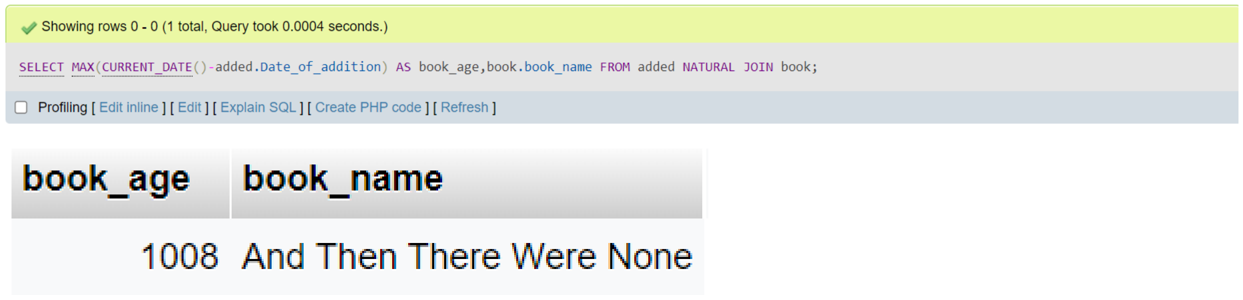


```
Showing rows 0 - 0 (1 total, Query took 0.0003 seconds.)

SELECT AVG(borrows_return.due_date-borrows_return.issue_date) AS average_time FROM borrows_return;

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]
```

average_time
26.6000

4. Oldest book in the library

   SELECT MAX(CURRENT_DATE()-added.Date_of_addition) AS book_age,book.book_name FROM added NATURAL JOIN book;



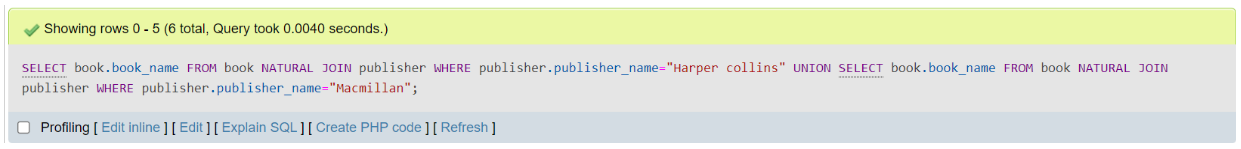
```
Showing rows 0 - 0 (1 total, Query took 0.0004 seconds.)

SELECT MAX(CURRENT_DATE()-added.Date_of_addition) AS book_age,book.book_name FROM added NATURAL JOIN book;

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]
```

book_age	book_name
1008	And Then There Were None

Set Operations

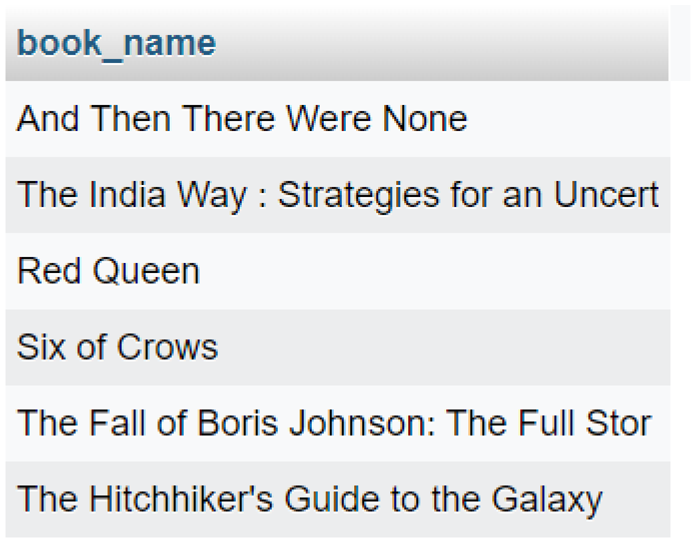
1. Find books published by harper collins and macmillan

   SELECT book.book_name FROM book NATURAL JOIN publisher WHERE publisher.publisher_name="Harper collins"
   UNION
   SELECT book.book_name FROM book NATURAL JOIN publisher WHERE publisher.publisher_name="Macmillan";



```
Showing rows 0 - 5 (6 total, Query took 0.0040 seconds.)

SELECT book.book_name FROM book NATURAL JOIN publisher WHERE publisher.publisher_name="Harper collins" UNION SELECT book.book_name FROM book NATURAL JOIN
publisher WHERE publisher.publisher_name="Macmillan";
```

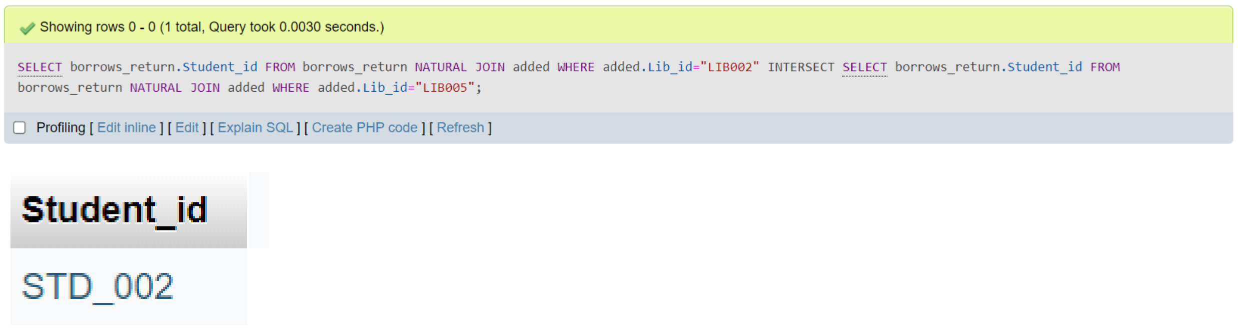
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]



book_name
And Then There Were None
The India Way : Strategies for an Uncert
Red Queen
Six of Crows
The Fall of Boris Johnson: The Full Stor
The Hitchhiker's Guide to the Galaxy

2. Find students who have borrowed books from library 2 and library 5

   SELECT borrows_return.Student_id FROM borrows_return NATURAL JOIN added WHERE added.Lib_id="LIB002"
   INTERSECT
   SELECT borrows_return.Student_id FROM borrows_return NATURAL JOIN added WHERE added.Lib_id="LIB005";



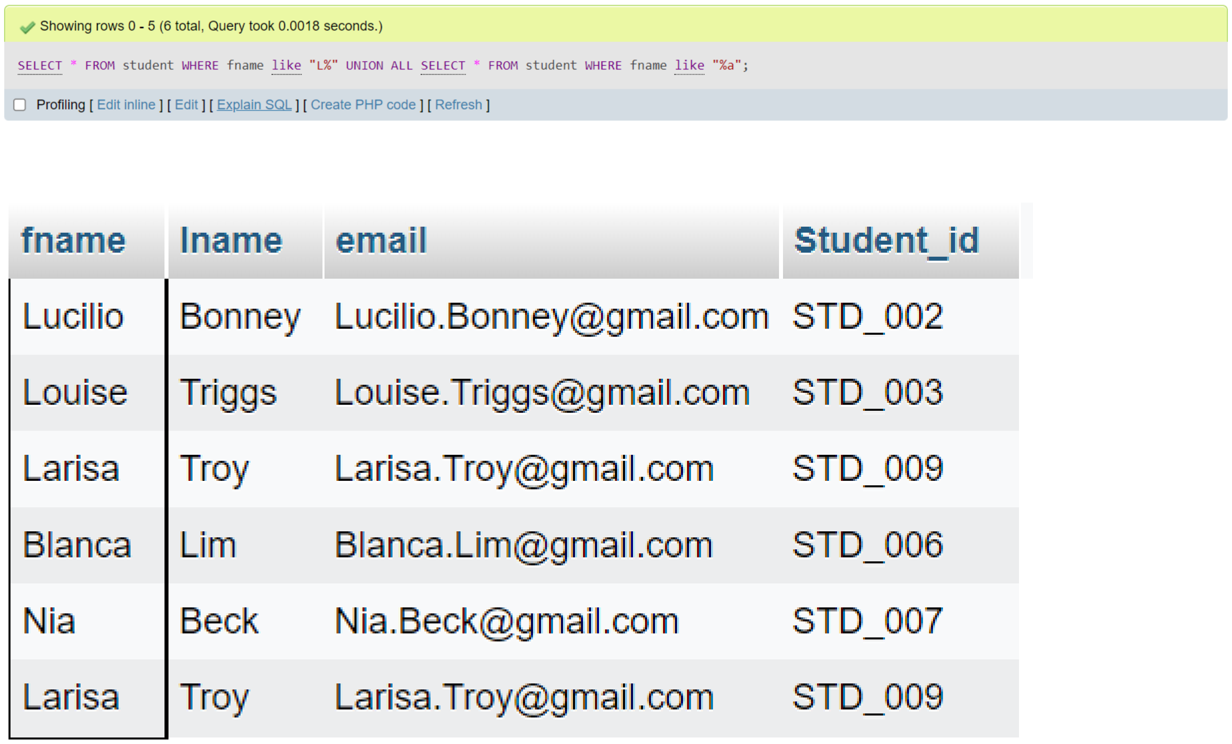
Showing rows 0 - 0 (1 total, Query took 0.0030 seconds.)

```
SELECT borrows_return.Student_id FROM borrows_return NATURAL JOIN added WHERE added.Lib_id="LIB002" INTERSECT SELECT borrows_return.Student_id FROM
borrows_return NATURAL JOIN added WHERE added.Lib_id="LIB005";
```

Student_id
STD_002

3. Find students whose first names begin with L or end with A

   SELECT * FROM student WHERE fname like "L%"
   UNION ALL
   SELECT * FROM student WHERE fname like "%a";



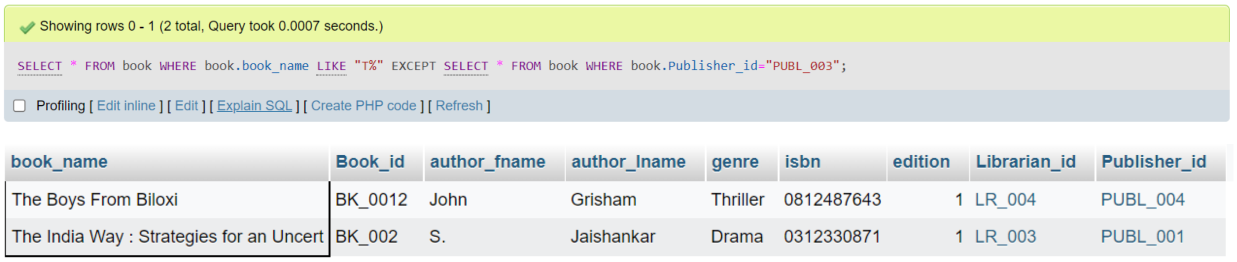
Showing rows 0 - 5 (6 total, Query took 0.0018 seconds.)

```
SELECT * FROM student WHERE fname like "L%" UNION ALL SELECT * FROM student WHERE fname like "%a";
```

fname	lname	email	Student_id
Lucilio	Bonney	Lucilio.Bonney@gmail.com	STD_002
Louise	Triggs	Louise.Triggs@gmail.com	STD_003
Larisa	Troy	Larisa.Troy@gmail.com	STD_009
Blanca	Lim	Blanca.Lim@gmail.com	STD_006
Nia	Beck	Nia.Beck@gmail.com	STD_007
Larisa	Troy	Larisa.Troy@gmail.com	STD_009

4. Find books which start with letter T but not those published by PUBL_003

   SELECT * FROM book WHERE book.book_name LIKE "T%"
   EXCEPT
   SELECT * FROM book WHERE book.Publisher_id="PUBL_003";


Showing rows 0 - 1 (2 total, Query took 0.0007 seconds.)

```
SELECT * FROM book WHERE book.book_name LIKE "T%" EXCEPT SELECT * FROM book WHERE book.Publisher_id="PUBL_003";
```

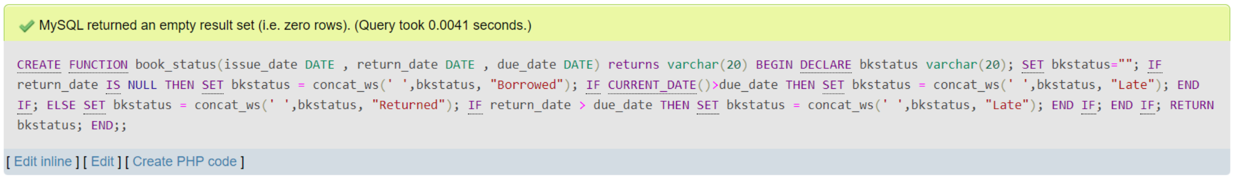
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

book_name	Book_id	author_fname	author_lname	genre	isbn	edition	Librarian_id	Publisher_id
The Boys From Biloxi	BK_0012	John	Grisham	Thriller	0812487643	1	LR_004	PUBL_004
The India Way : Strategies for an Uncert	BK_002	S.	Jaishankar	Drama	0312330871	1	LR_003	PUBL_001

Functions and Procedures

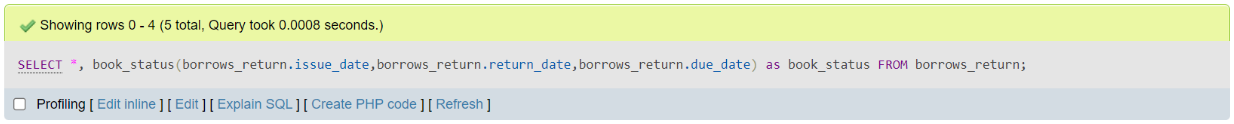
1. Function to check status off book , if its borrowed , returned or is overdue

```
DELIMITER $$
CREATE FUNCTION book_status(issue_date DATE , return_date DATE ,
due_date DATE)
returns varchar(20)
BEGIN
  DECLARE bkstatus varchar(20);
    SET bkstatus="";
    IF return_date IS NULL THEN
      SET bkstatus = concat_ws(' ',bkstatus, "Borrowed");
        IF CURRENT_DATE()>due_date THEN
          SET bkstatus = concat_ws(' ',bkstatus, "Late");
        END IF;
    ELSE
       SET bkstatus = concat_ws(' ',bkstatus, "Returned");
       IF return_date > due_date THEN
          SET bkstatus = concat_ws(' ',bkstatus, "Late");
       END IF;
    END IF;
   RETURN bkstatus;
END; $$
```



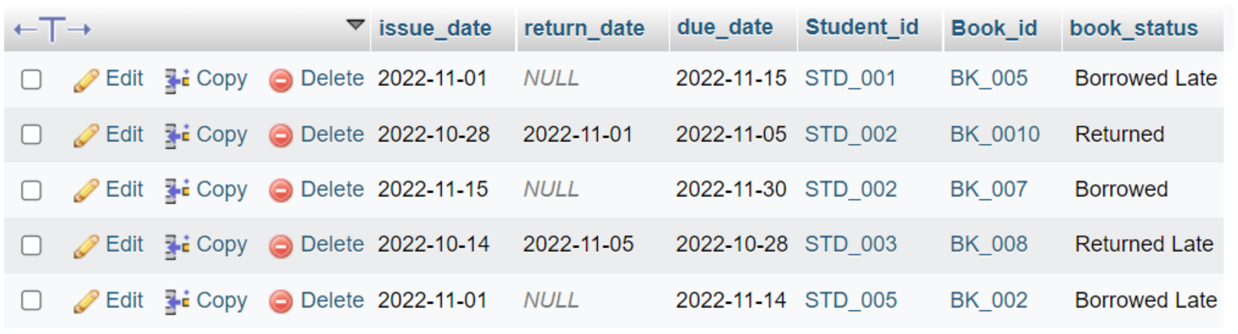
```
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0041 seconds.)

CREATE FUNCTION book_status(issue_date DATE , return_date DATE , due_date DATE) returns varchar(20) BEGIN DECLARE bkstatus varchar(20); SET bkstatus=""; IF
return_date IS NULL THEN SET bkstatus = concat_ws(' ',bkstatus, "Borrowed"); IF CURRENT_DATE()>due_date THEN SET bkstatus = concat_ws(' ',bkstatus, "Late"); END
IF; ELSE SET bkstatus = concat_ws(' ',bkstatus, "Returned"); IF return_date > due_date THEN SET bkstatus = concat_ws(' ',bkstatus, "Late"); END IF; END IF; RETURN
bkstatus; END;;
```

```
SELECT *,
book_status(borrows_return.issue_date,borrows_return.return_date,borrows_re
turn.due_date) as book_status FROM borrows_return;
```



```
Showing rows 0 - 4 (5 total, Query took 0.0008 seconds.)

SELECT *, book_status(borrows_return.issue_date,borrows_return.return_date,borrows_return.due_date) as book_status FROM borrows_return;
```

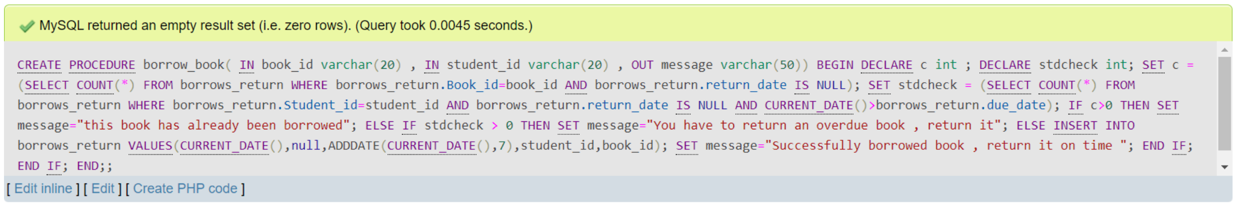
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]



	issue_date	return_date	due_date	Student_id	Book_id	book_status
Edit Copy Delete	2022-11-01	NULL	2022-11-15	STD_001	BK_005	Borrowed Late
Edit Copy Delete	2022-10-28	2022-11-01	2022-11-05	STD_002	BK_0010	Returned
Edit Copy Delete	2022-11-15	NULL	2022-11-30	STD_002	BK_007	Borrowed
Edit Copy Delete	2022-10-14	2022-11-05	2022-10-28	STD_003	BK_008	Returned Late
Edit Copy Delete	2022-11-01	NULL	2022-11-14	STD_005	BK_002	Borrowed Late

2. Procedure to borrow book

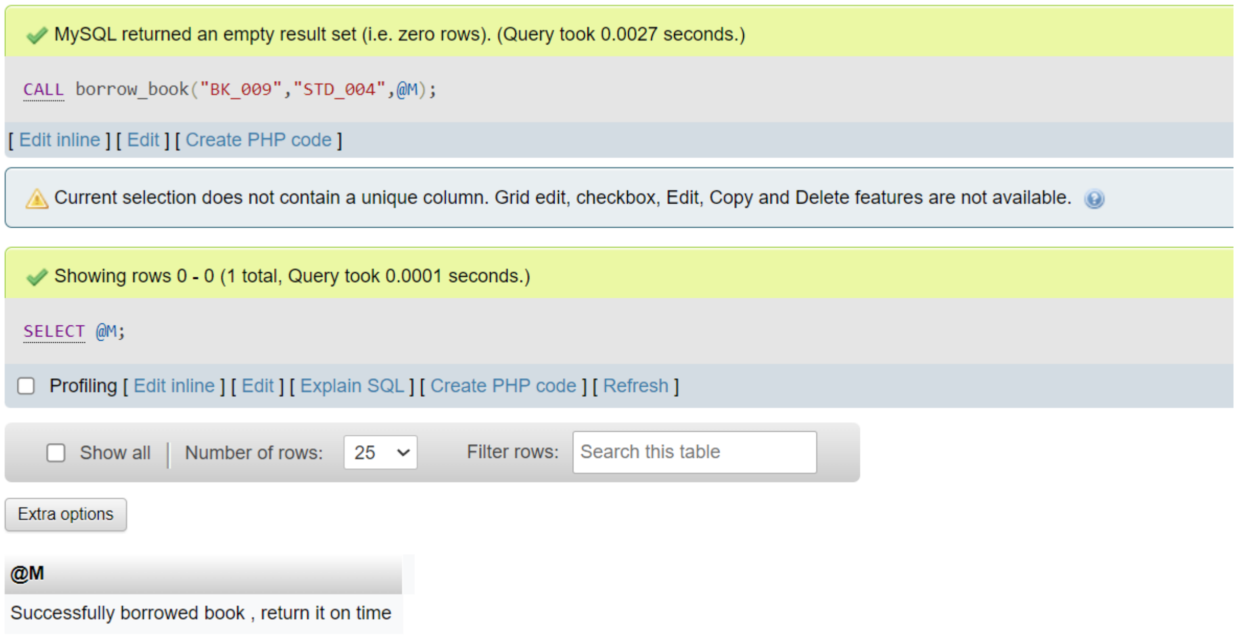
```
DELIMITER $$
CREATE PROCEDURE borrow_book(
   IN book_id varchar(20) , IN student_id varchar(20) , OUT message
varchar(50))
BEGIN
DECLARE c int ;
DECLARE stdcheck int;
SET c =(SELECT COUNT(*) FROM borrows_return WHERE
borrows_return.Book_id=book_id AND borrows_return.return_date IS NULL);
SET stdcheck = (SELECT COUNT(*) FROM borrows_return WHERE
borrows_return.Student_id=student_id AND borrows_return.return_date IS
NULL AND CURRENT_DATE()>borrows_return.due_date);
IF c>0 THEN
 SET message="this book has already been borrowed";
ELSE
 IF stdcheck > 0 THEN
 SET message="You have to return an overdue book , return it";
 ELSE
 INSERT INTO borrows_return
VALUES(CURRENT_DATE(),null,ADDDATE(CURRENT_DATE(),7),student_id,
book_id);
 SET message="Successfully borrowed book , return it on time ";
 END IF;
END IF;
END; $$
```


MySQL returned an empty result set (i.e. zero rows). (Query took 0.0045 seconds.)

```
CREATE PROCEDURE borrow_book( IN book_id varchar(20) , IN student_id varchar(20) , OUT message varchar(50)) BEGIN DECLARE c int ; DECLARE stdcheck int; SET c =
(SELECT COUNT(*) FROM borrows_return WHERE borrows_return.Book_id=book_id AND borrows_return.return_date IS NULL); SET stdcheck = (SELECT COUNT(*) FROM
borrows_return WHERE borrows_return.Student_id=student_id AND borrows_return.return_date IS NULL AND CURRENT_DATE()>borrows_return.due_date); IF c>0 THEN SET
message="this book has already been borrowed"; ELSE IF stdcheck > 0 THEN SET message="You have to return an overdue book , return it"; ELSE INSERT INTO
borrows_return VALUES(CURRENT_DATE(),null,ADDDATE(CURRENT_DATE(),7),student_id,book_id); SET message="Successfully borrowed book , return it on time "; END IF;
END IF; END;;
```

[Edit inline] [Edit] [Create PHP code]

CALL borrow_book("BK_009","STD_004",@M)
SELECT @M;


MySQL returned an empty result set (i.e. zero rows). (Query took 0.0027 seconds.)

```
CALL borrow_book("BK_009","STD_004",@M);
```

[Edit inline] [Edit] [Create PHP code]

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 0 (1 total, Query took 0.0001 seconds.)

```
SELECT @M;
```

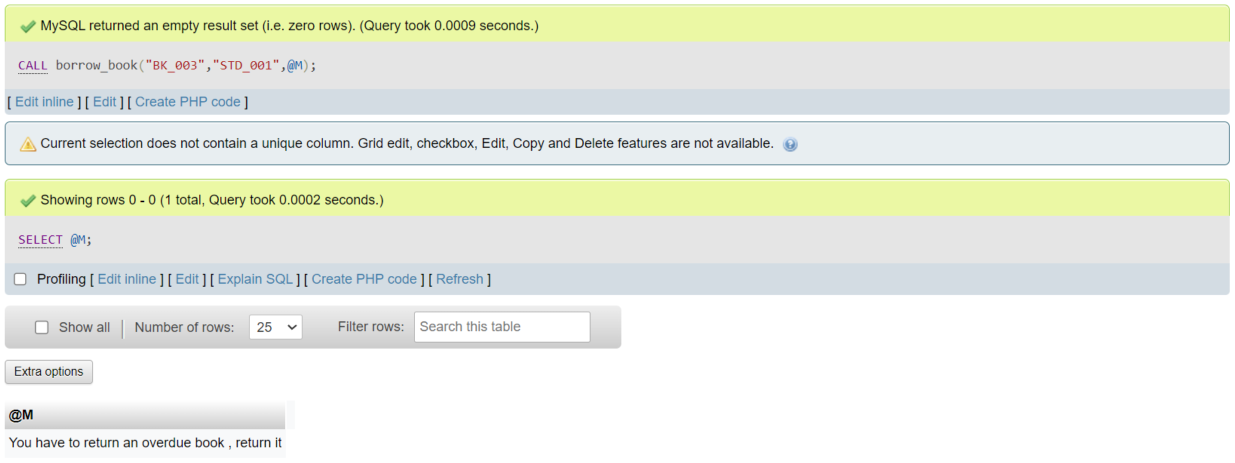
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

@M
Successfully borrowed book , return it on time

CALL borrow_book("BK_003","STD_001",@M);
SELECT @M;


MySQL returned an empty result set (i.e. zero rows). (Query took 0.0009 seconds.)

```
CALL borrow_book("BK_003","STD_001",@M);
```

[Edit inline] [Edit] [Create PHP code]

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)

```
SELECT @M;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

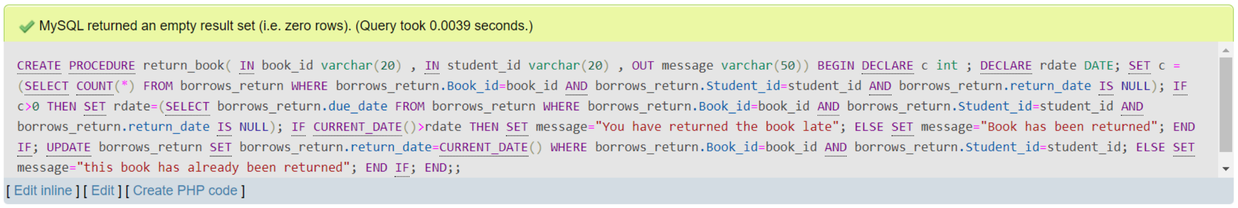
Show all | Number of rows: 25 Filter rows: Search this table

Extra options

@M
You have to return an overdue book , return it

3. Procedure to return book

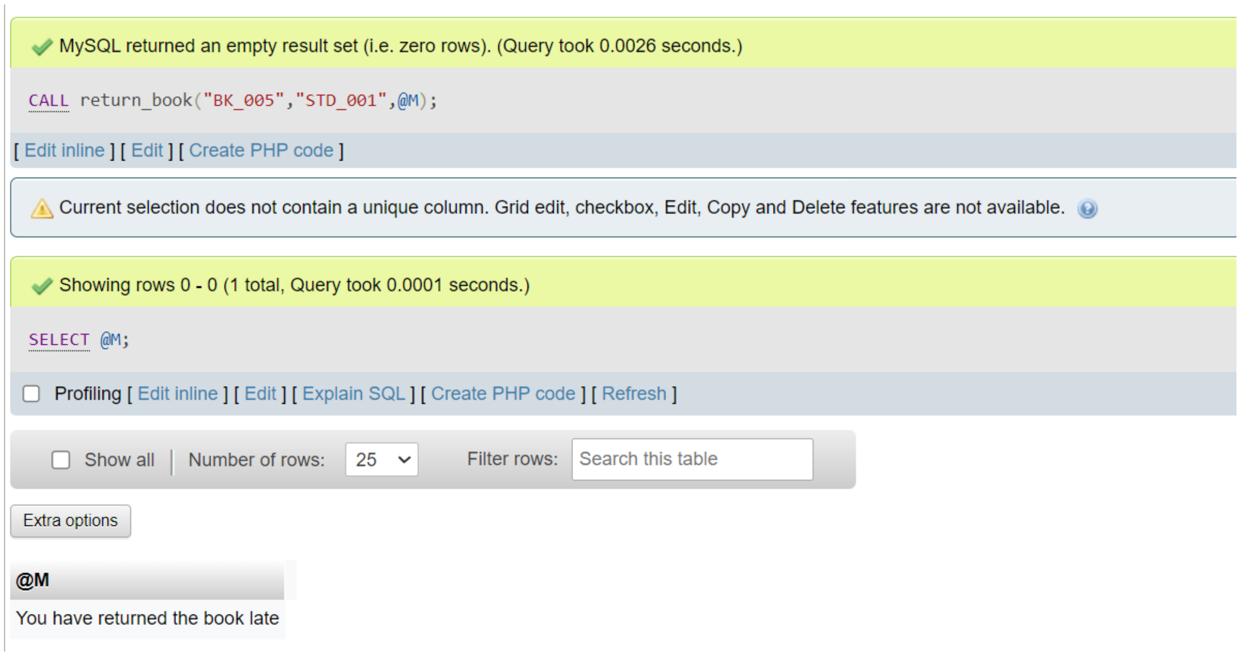
```
DELIMITER $$
CREATE PROCEDURE return_book(
   IN book_id varchar(20) , IN student_id varchar(20) , OUT message
varchar(50))
BEGIN
DECLARE c int ;
DECLARE rdate DATE;
SET c = (SELECT COUNT(*) FROM borrows_return WHERE
borrows_return.Book_id=book_id AND borrows_return.Student_id=student_id
AND borrows_return.return_date IS NULL);
IF c>0 THEN
 SET rdate=(SELECT borrows_return.due_date FROM borrows_return WHERE
borrows_return.Book_id=book_id AND borrows_return.Student_id=student_id
AND borrows_return.return_date IS NULL);
 IF CURRENT_DATE()>rdate THEN
 SET message="You have returned the book late";
 ELSE
 SET message="Book has been returned";
 END IF;
UPDATE borrows_return SET borrows_return.return_date=CURRENT_DATE()
WHERE borrows_return.Book_id=book_id AND
borrows_return.Student_id=student_id;
ELSE
 SET message="this book has already been returned";
END IF;
END; $$
```



```
✔ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0039 seconds.)

CREATE PROCEDURE return_book( IN book_id varchar(20) , IN student_id varchar(20) , OUT message varchar(50)) BEGIN DECLARE c int ; DECLARE rdate DATE; SET c =
(SELECT COUNT(*) FROM borrows_return WHERE borrows_return.Book_id=book_id AND borrows_return.Student_id=student_id AND borrows_return.return_date IS NULL); IF
c>0 THEN SET rdate=(SELECT borrows_return.due_date FROM borrows_return WHERE borrows_return.Book_id=book_id AND borrows_return.Student_id=student_id AND
borrows_return.return_date IS NULL); IF CURRENT_DATE()>rdate THEN SET message="You have returned the book late"; ELSE SET message="Book has been returned"; END
IF; UPDATE borrows_return SET borrows_return.return_date=CURRENT_DATE() WHERE borrows_return.Book_id=book_id AND borrows_return.Student_id=student_id; ELSE SET
message="this book has already been returned"; END IF; END;;
```

```
CALL return_book("BK_005","STD_001",@M);
```

SELECT @M;



MySQL returned an empty result set (i.e. zero rows). (Query took 0.0026 seconds.)

```
CALL return_book("BK_005","STD_001",@M);
```

[Edit inline] [Edit] [Create PHP code]

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 0 (1 total, Query took 0.0001 seconds.)

```
SELECT @M;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

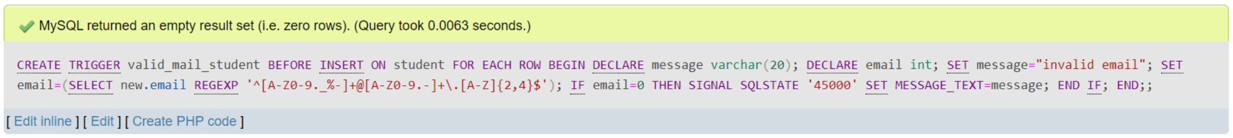
Extra options

@M
You have returned the book late

Triggers and Cursors

1. Trigger to check if email entered by a student is valid or not

```
DELIMITER $$
CREATE TRIGGER valid_mail_student
BEFORE INSERT ON student FOR EACH ROW
BEGIN
DECLARE message varchar(20);
DECLARE email int;
SET message="invalid email";
SET email=(SELECT new.email REGEXP
'^[A-Z0-9._%-]+@[A-Z0-9.-]+\.[A-Z]{2,4}$');
IF email=0 THEN
  SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT=message;
END IF;
END; $$
```

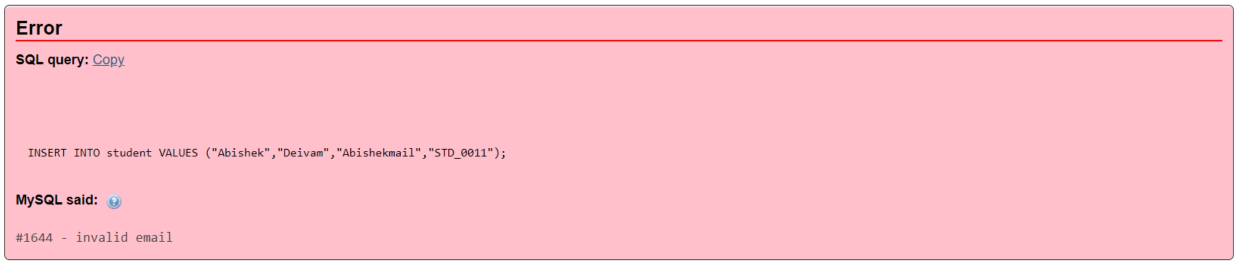


MySQL returned an empty result set (i.e. zero rows). (Query took 0.0063 seconds.)

```
CREATE TRIGGER valid_mail_student BEFORE INSERT ON student FOR EACH ROW BEGIN DECLARE message varchar(20); DECLARE email int; SET message="invalid email"; SET
email=(SELECT new.email REGEXP '^[A-Z0-9._%-]+@[A-Z0-9.-]+\.[A-Z]{2,4}$'); IF email=0 THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT=message; END IF; END;;
```

[Edit inline] [Edit] [Create PHP code]

```
INSERT INTO student VALUES
("Abishek","Deivam","Abishekmail","STD_0011");
```



Error

SQL query: Copy

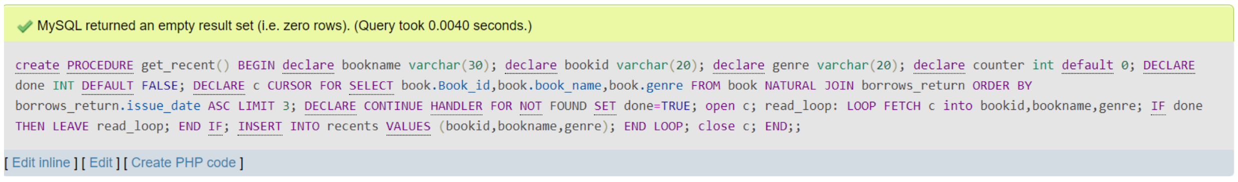
```
INSERT INTO student VALUES ("Abishek","Deivam","Abishekmail","STD_0011");
```

MySQL said:

```
#1644 - invalid email
```

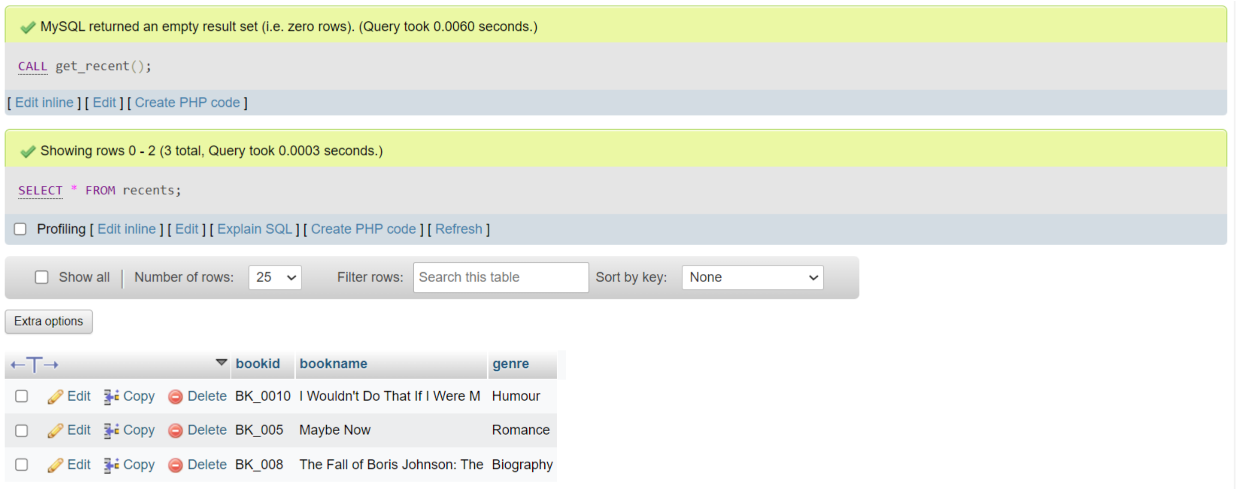
2. Cursor to copy the 3 most recently borrowed books into another table

```
DELIMITER $$
create PROCEDURE get_recent()
BEGIN
declare bookname varchar(30);
declare bookid varchar(20);
declare genre varchar(20);
declare counter int default 0;
DECLARE done INT DEFAULT FALSE;
DECLARE c CURSOR FOR SELECT
book.Book_id,book.book_name,book.genre FROM book NATURAL JOIN
borrows_return ORDER BY borrows_return.issue_date ASC LIMIT 3;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done=TRUE;
open c;
read_loop: LOOP
FETCH c into bookid,bookname,genre;
IF done THEN
LEAVE read_loop;
END IF;
INSERT INTO recents VALUES (bookid,bookname,genre);
END LOOP;
close c;
END; $$
```



```
✔ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0040 seconds.)

create PROCEDURE get_recent() BEGIN declare bookname varchar(30); declare bookid varchar(20); declare genre varchar(20); declare counter int default 0; DECLARE
done INT DEFAULT FALSE; DECLARE c CURSOR FOR SELECT book.Book_id,book.book_name,book.genre FROM book NATURAL JOIN borrows_return ORDER BY
borrows_return.issue_date ASC LIMIT 3; DECLARE CONTINUE HANDLER FOR NOT FOUND SET done=TRUE; open c; read_loop: LOOP FETCH c into bookid,bookname,genre; IF done
THEN LEAVE read_loop; END IF; INSERT INTO recents VALUES (bookid,bookname,genre); END LOOP; close c; END;;
```

```
CALL get_recent();
SELECT * FROM recents;
```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0060 seconds.)

```
CALL get_recent();
```

[Edit inline] [Edit] [Create PHP code]

✓ Showing rows 0 - 2 (3 total, Query took 0.0003 seconds.)

```
SELECT * FROM recents;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

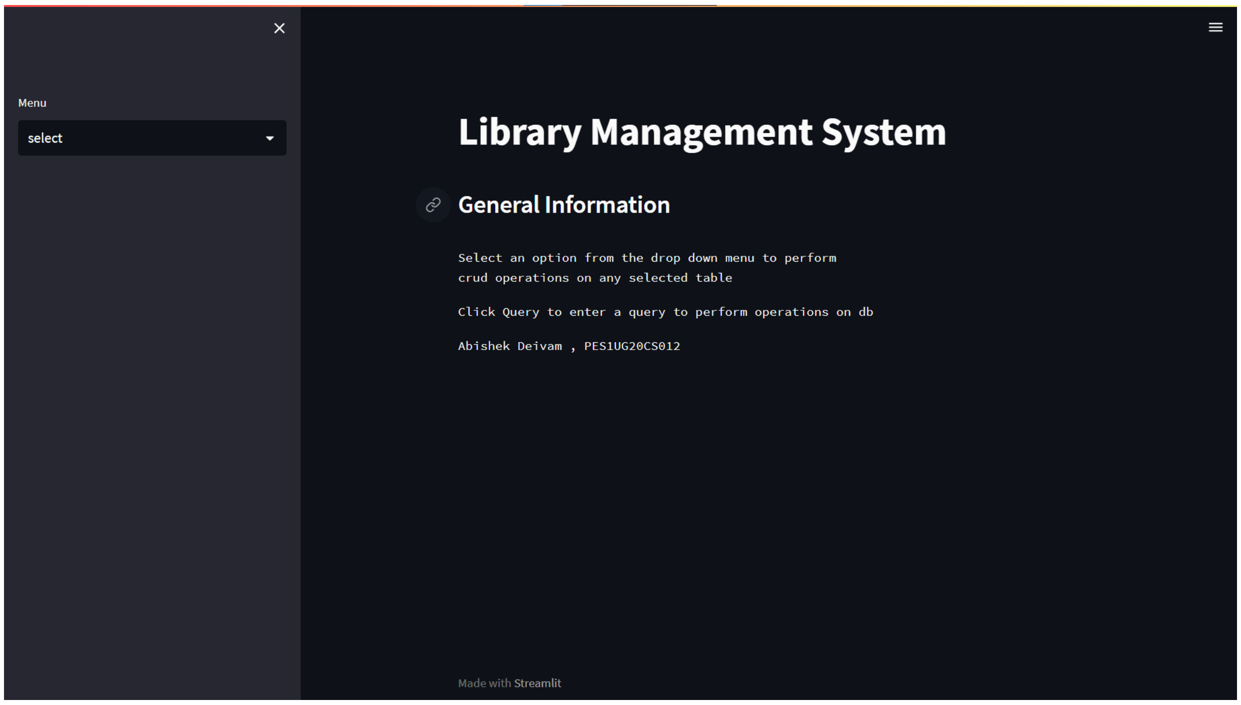
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	bookid	bookname	genre
Edit Copy Delete	BK_0010	I Wouldn't Do That If I Were M	Humour
Edit Copy Delete	BK_005	Maybe Now	Romance
Edit Copy Delete	BK_008	The Fall of Boris Johnson: The	Biography

Developing a Frontend

General

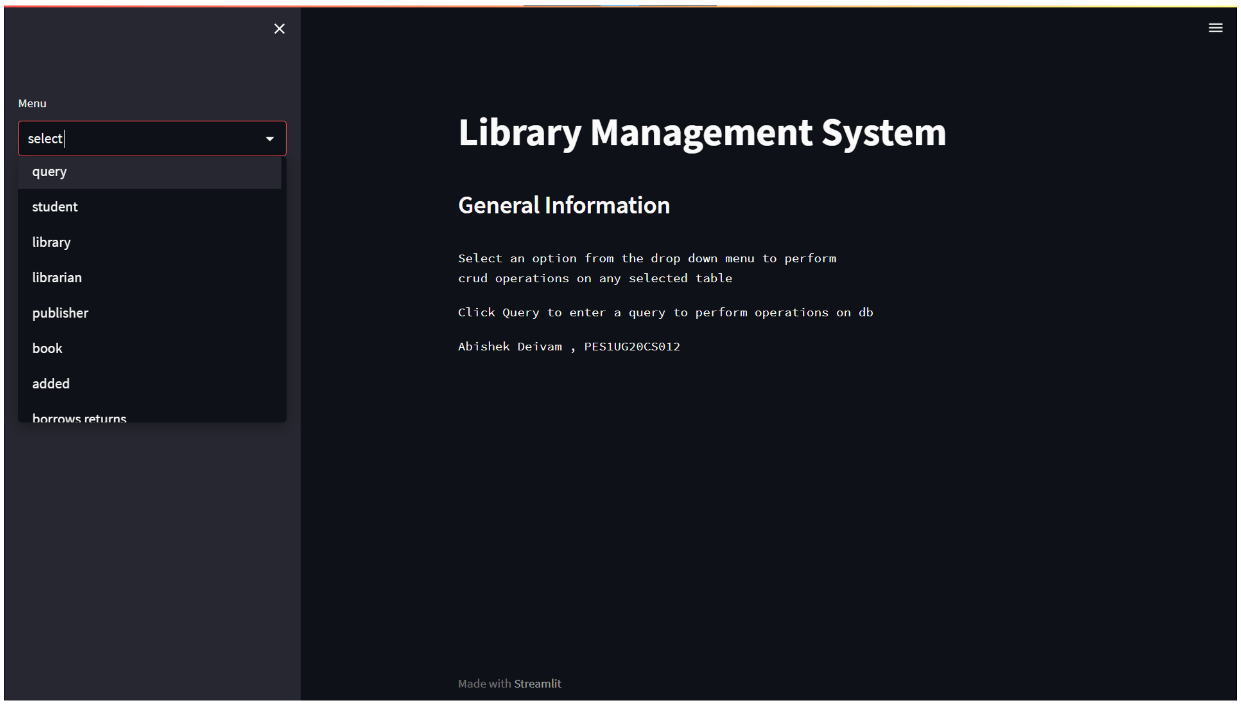


Library Management System

General Information

```
Select an option from the drop down menu to perform
crud operations on any selected table

Click Query to enter a query to perform operations on db

Abishek Deivam , PES1UG20CS012
```

List of tables

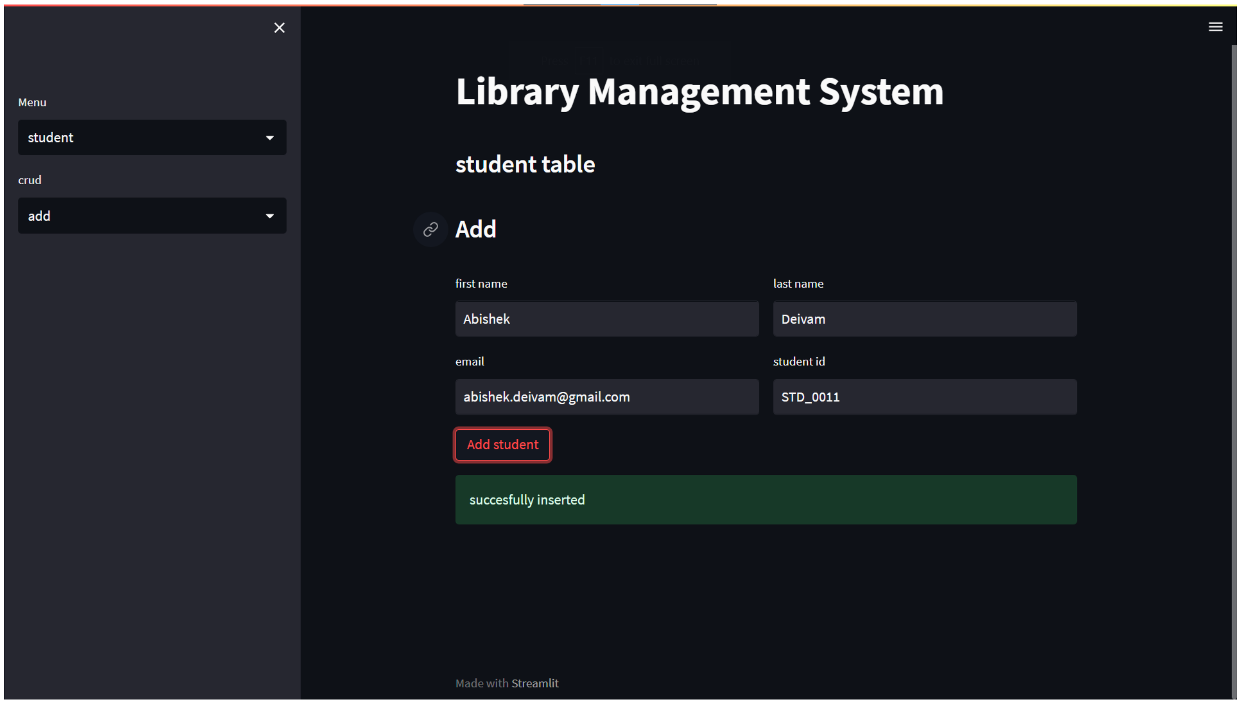


Library Management System

General Information

```
Select an option from the drop down menu to perform
crud operations on any selected table

Click Query to enter a query to perform operations on db

Abishek Deivam , PES1UG20CS012
```

Create new record in student table



Library Management System

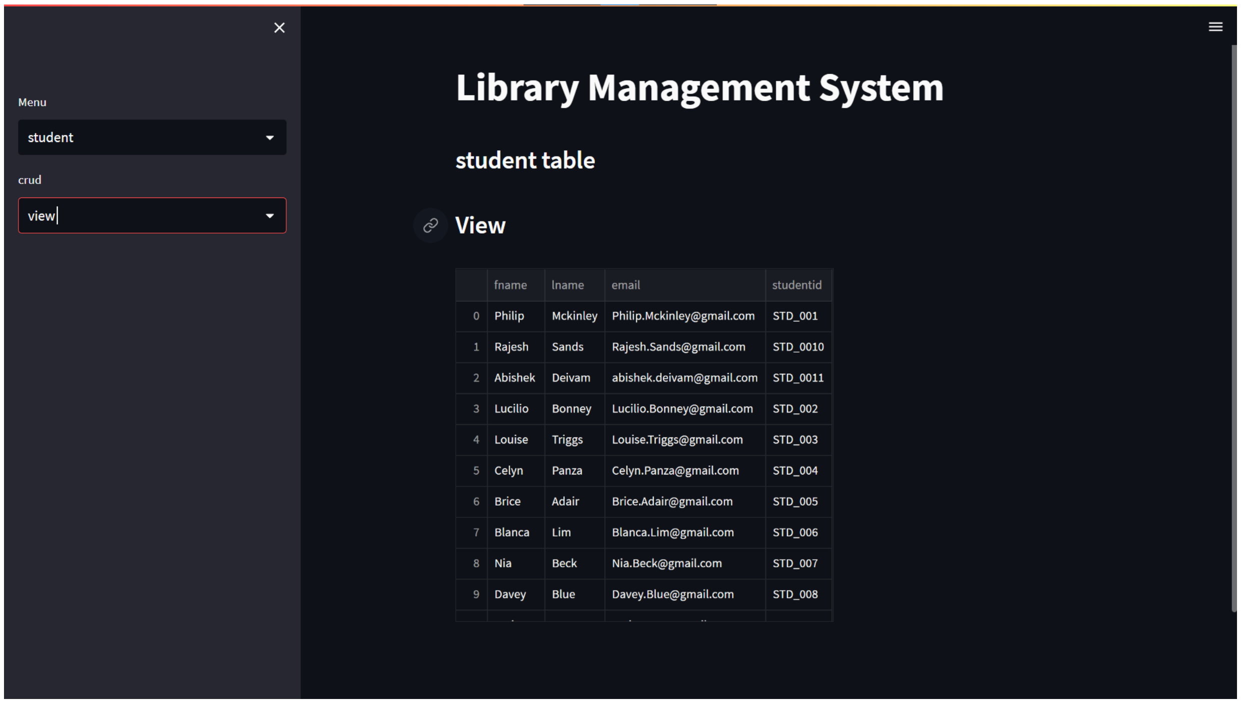
student table

Add

first name	Abishek
last name	Deivam
email	abishek.deivam@gmail.com
student id	STD_0011

succesfully inserted

Read records from student table



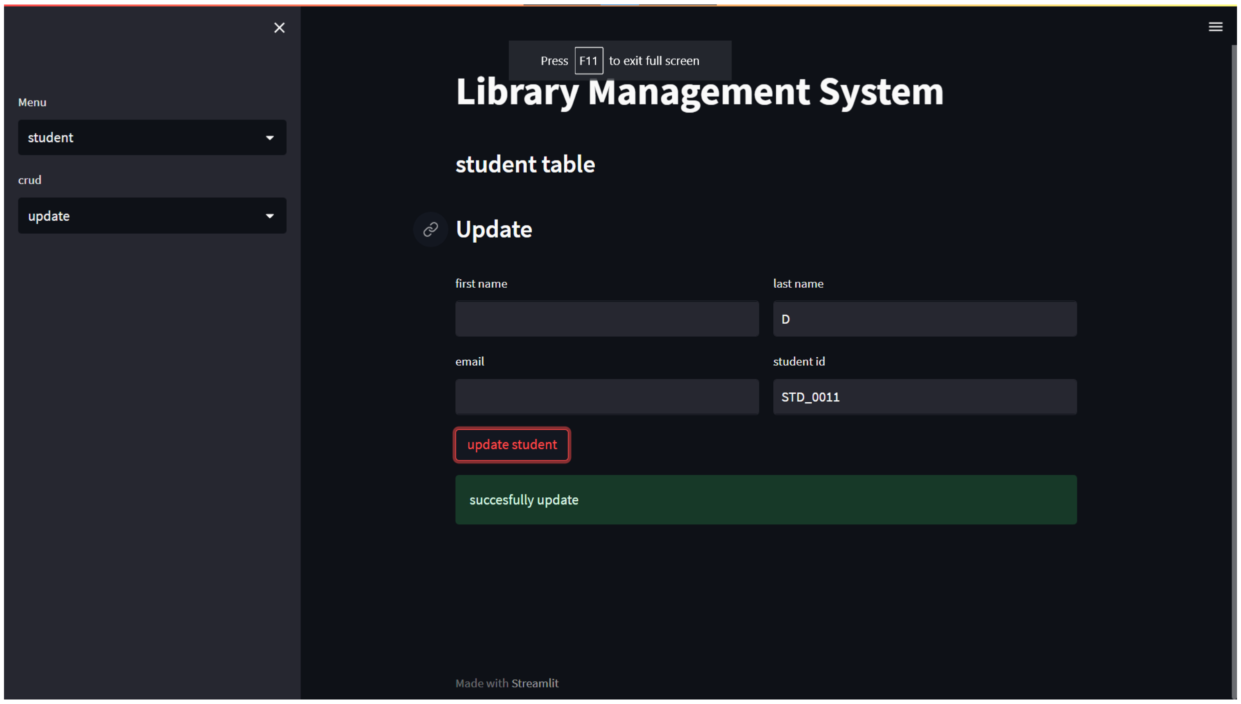
Library Management System

student table

View

	fname	lname	email	studentid
0	Philip	Mckinley	Philip.Mckinley@gmail.com	STD_001
1	Rajesh	Sands	Rajesh.Sands@gmail.com	STD_0010
2	Abishek	Deivam	abishek.deivam@gmail.com	STD_0011
3	Lucilio	Bonney	Lucilio.Bonney@gmail.com	STD_002
4	Louise	Triggs	Louise.Triggs@gmail.com	STD_003
5	Celyn	Panza	Celyn.Panza@gmail.com	STD_004
6	Brice	Adair	Brice.Adair@gmail.com	STD_005
7	Blanca	Lim	Blanca.Lim@gmail.com	STD_006
8	Nia	Beck	Nia.Beck@gmail.com	STD_007
9	Davey	Blue	Davey.Blue@gmail.com	STD_008

Update record from student table



Library Management System

student table

Update

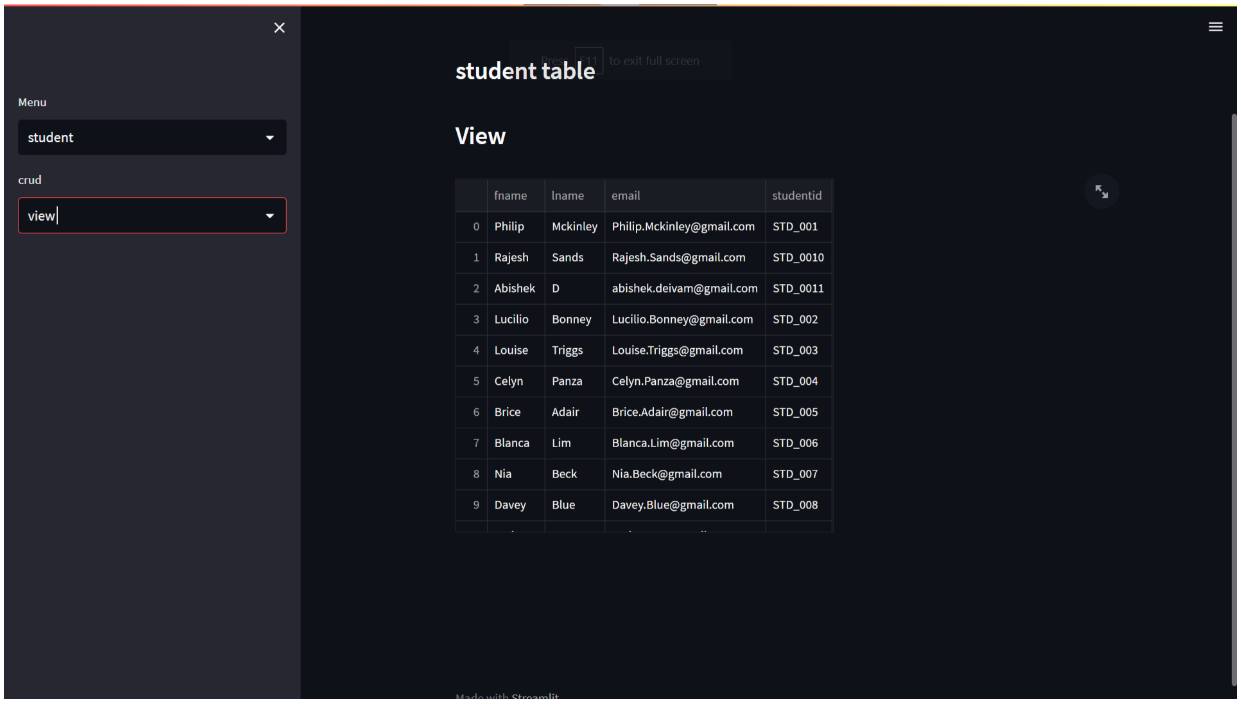
first name:

last name: D

email:

student id: STD_0011

succesfully update



Menu: student

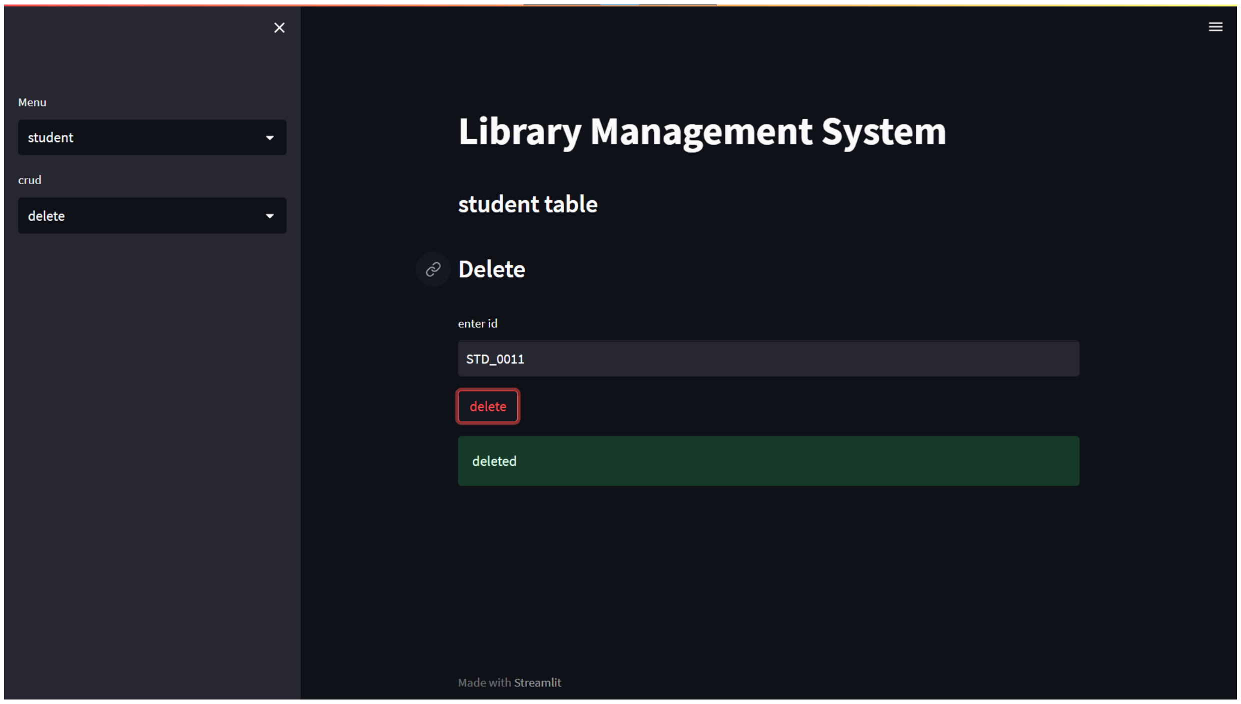
crud: view

student table

View

	fname	lname	email	studentid
0	Philip	Mckinley	Philip.Mckinley@gmail.com	STD_001
1	Rajesh	Sands	Rajesh.Sands@gmail.com	STD_0010
2	Abishek	D	abishek.deivam@gmail.com	STD_0011
3	Lucilio	Bonney	Lucilio.Bonney@gmail.com	STD_002
4	Louise	Triggs	Louise.Triggs@gmail.com	STD_003
5	Celyn	Panza	Celyn.Panza@gmail.com	STD_004
6	Brice	Adair	Brice.Adair@gmail.com	STD_005
7	Blanca	Lim	Blanca.Lim@gmail.com	STD_006
8	Nia	Beck	Nia.Beck@gmail.com	STD_007
9	Davey	Blue	Davey.Blue@gmail.com	STD_008

Delete record from student table



Library Management System

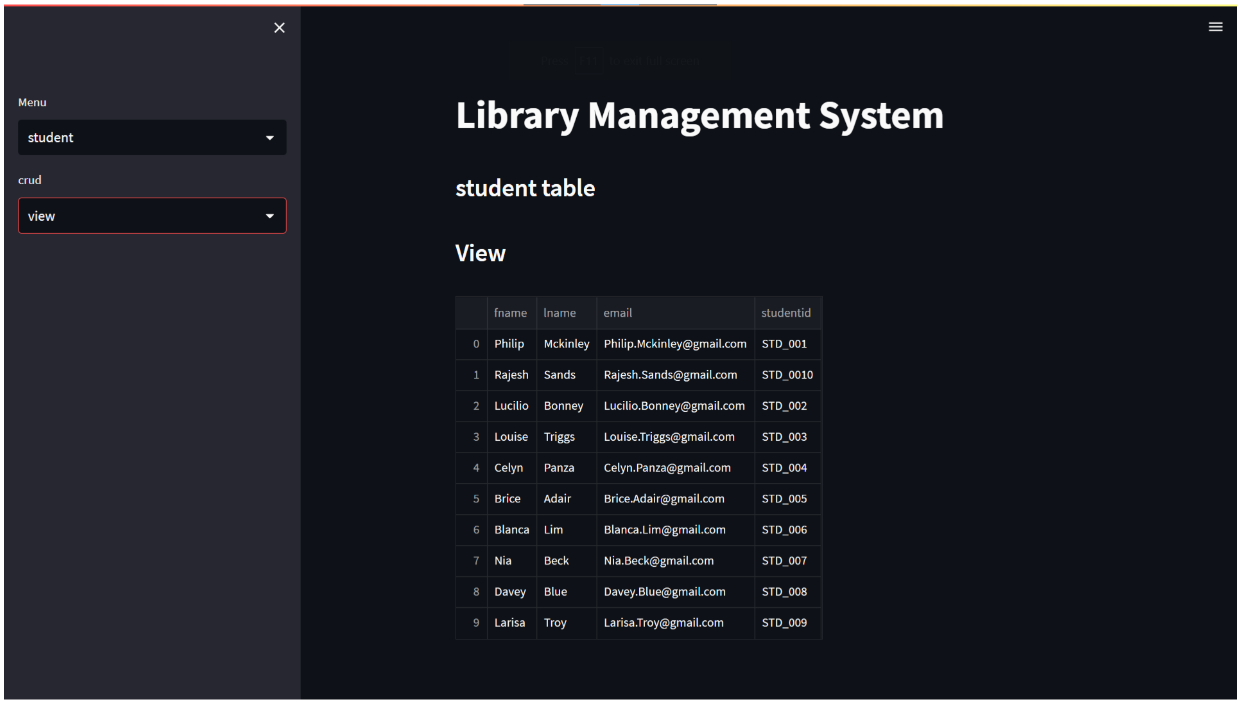
student table

Delete

enter id

STD_0011

deleted



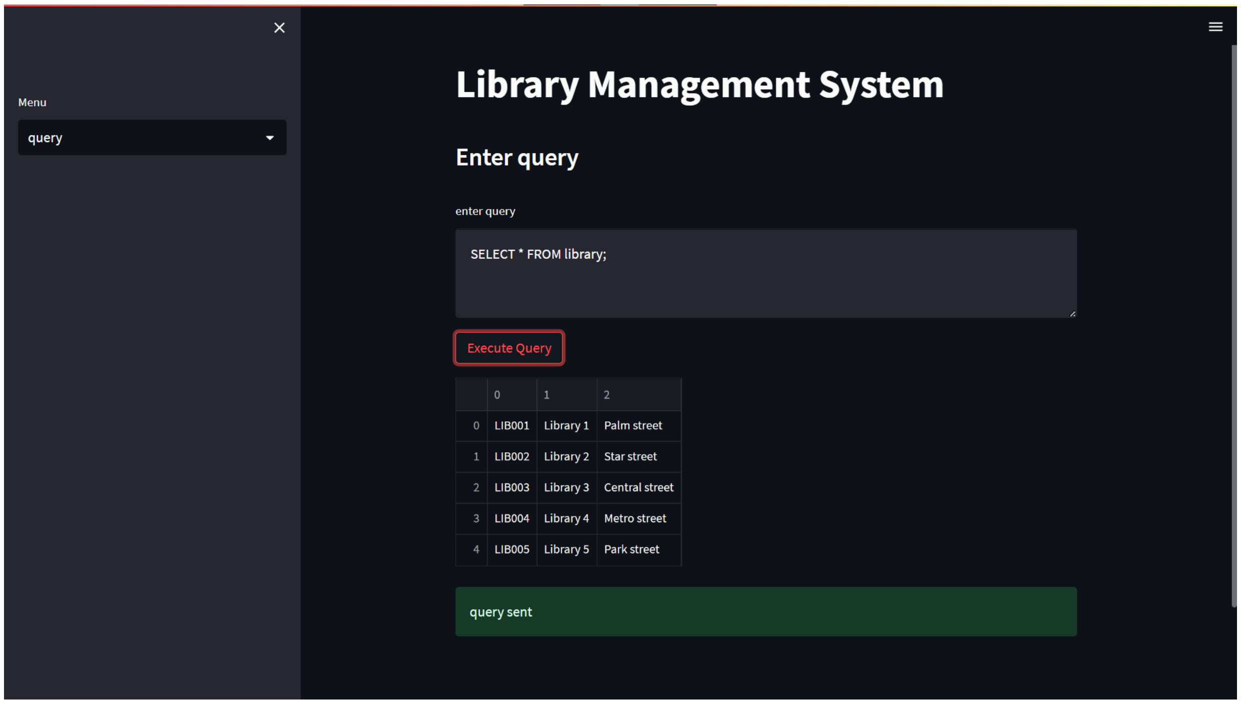
Library Management System

student table

View

	fname	lname	email	studentid
0	Philip	Mckinley	Philip.Mckinley@gmail.com	STD_001
1	Rajesh	Sands	Rajesh.Sands@gmail.com	STD_0010
2	Lucilio	Bonney	Lucilio.Bonney@gmail.com	STD_002
3	Louise	Triggs	Louise.Triggs@gmail.com	STD_003
4	Celyn	Panza	Celyn.Panza@gmail.com	STD_004
5	Brice	Adair	Brice.Adair@gmail.com	STD_005
6	Blanca	Lim	Blanca.Lim@gmail.com	STD_006
7	Nia	Beck	Nia.Beck@gmail.com	STD_007
8	Davey	Blue	Davey.Blue@gmail.com	STD_008
9	Larisa	Troy	Larisa.Troy@gmail.com	STD_009

2. Query box to enter queries



Library Management System

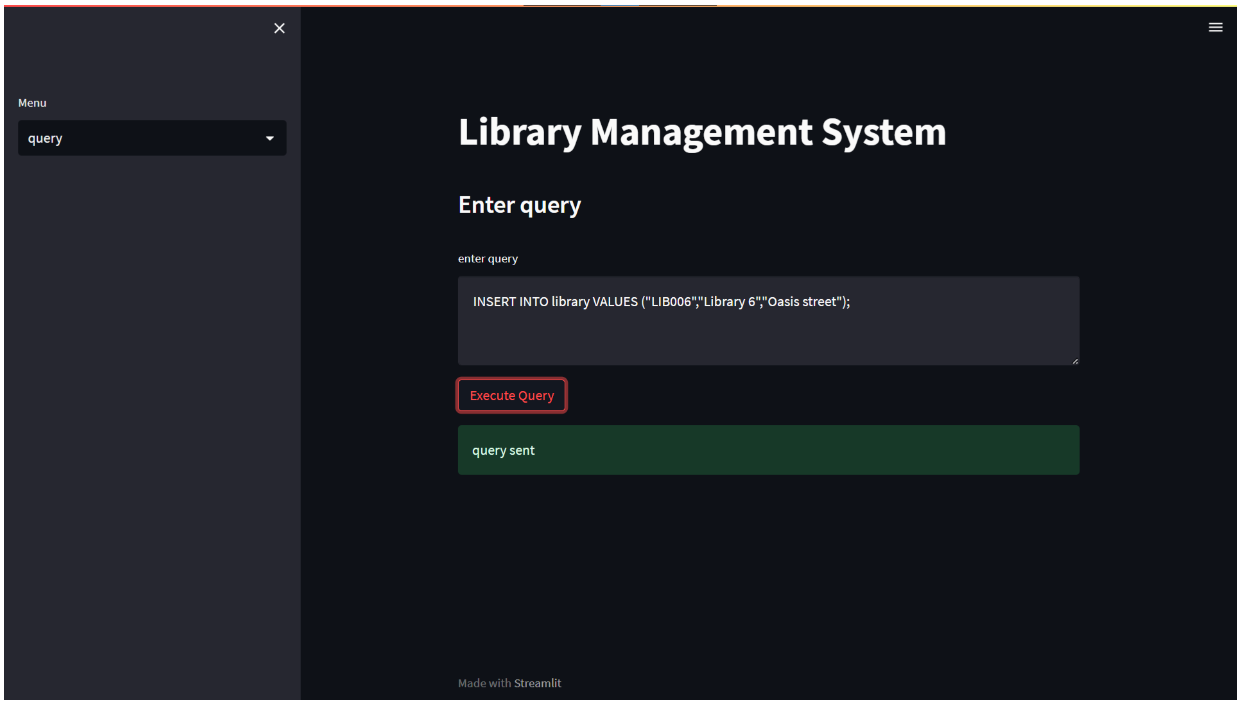
Enter query

enter query

```
SELECT * FROM library;
```

	0	1	2
0	LIB001	Library 1	Palm street
1	LIB002	Library 2	Star street
2	LIB003	Library 3	Central street
3	LIB004	Library 4	Metro street
4	LIB005	Library 5	Park street

query sent



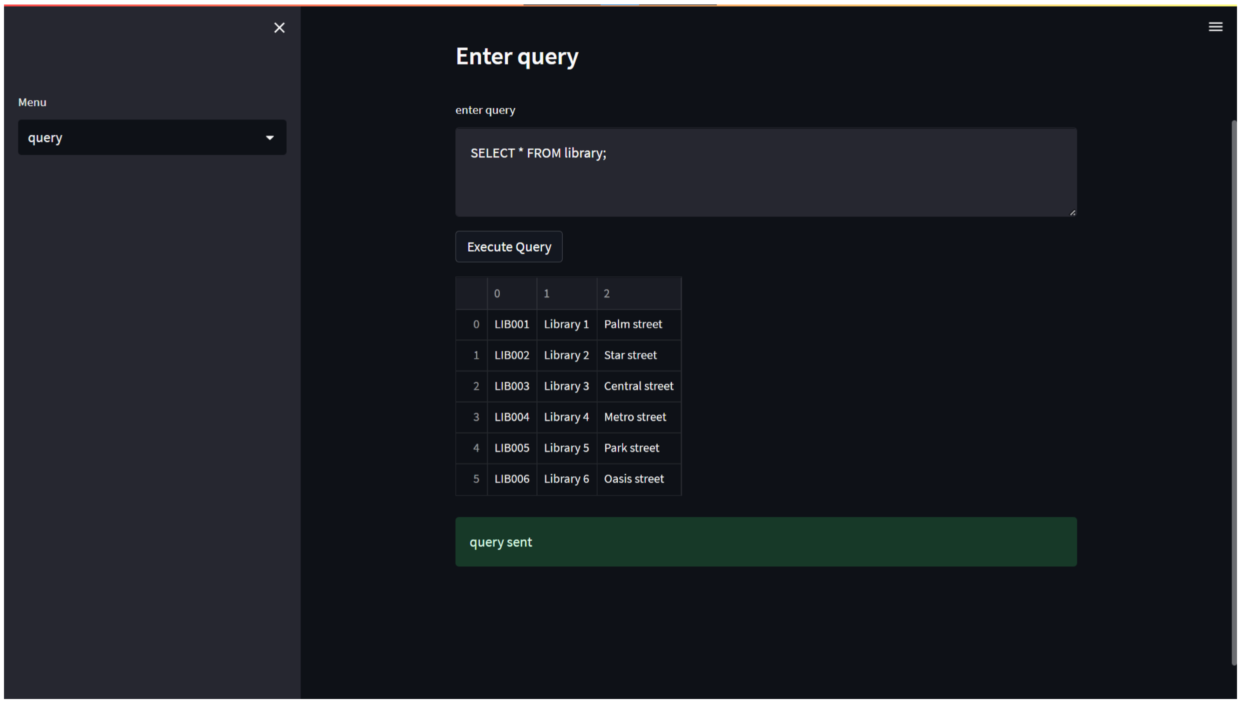
Library Management System

Enter query

enter query

```
INSERT INTO library VALUES ("LIB006","Library 6","Oasis street");
```

query sent



```
SELECT * FROM library;
```

	0	1	2
0	LIB001	Library 1	Palm street
1	LIB002	Library 2	Star street
2	LIB003	Library 3	Central street
3	LIB004	Library 4	Metro street
4	LIB005	Library 5	Park street
5	LIB006	Library 6	Oasis street

query sent