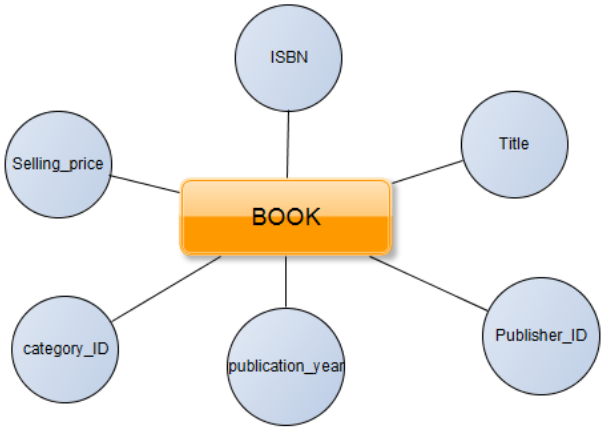
LAB PROJECT

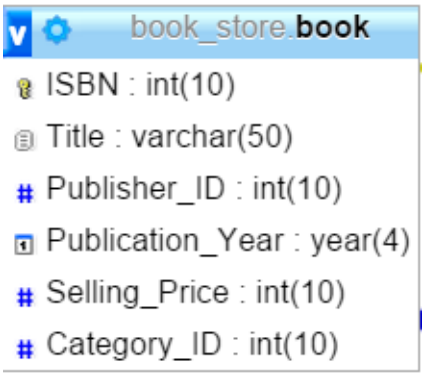
NAME(S) :

1) **Ebtehal Elaaraby Elsaid (2).**
2) **Remon Hanna Wadie (21).**
3) **Shaimaa Mousa (26).**
4) **Tarek Khaled Ragab (30).**

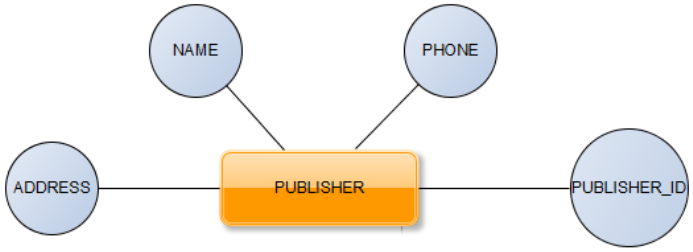
1) Database Design :

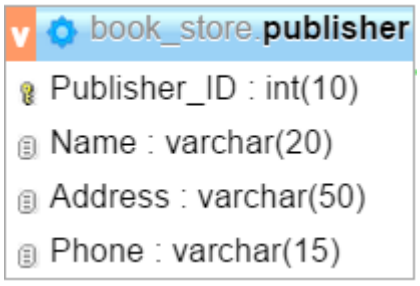
1) BOOK :



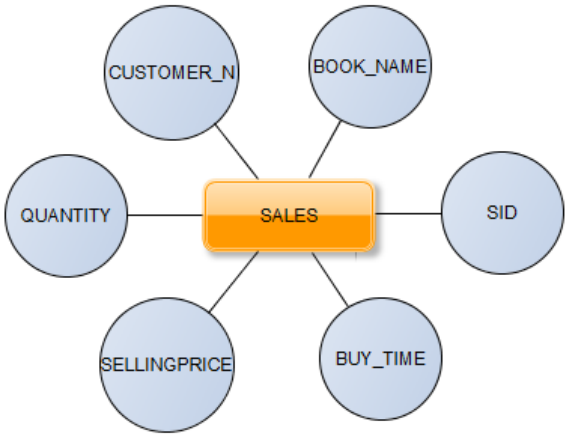


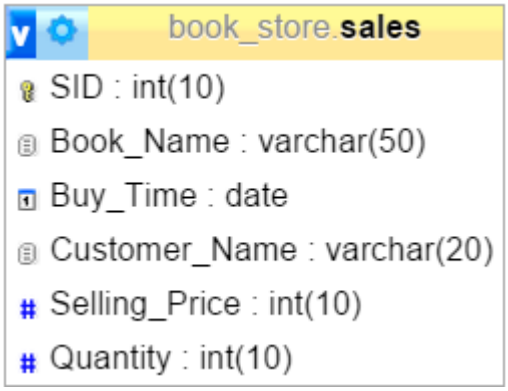
2) Publisher :



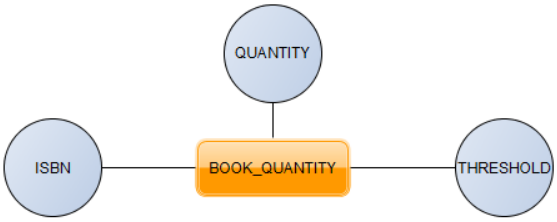


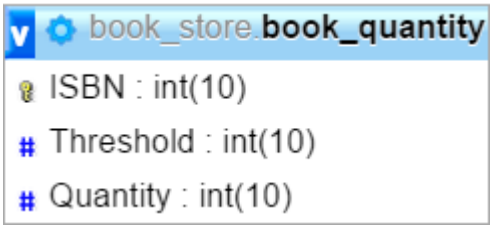
3) Sales :



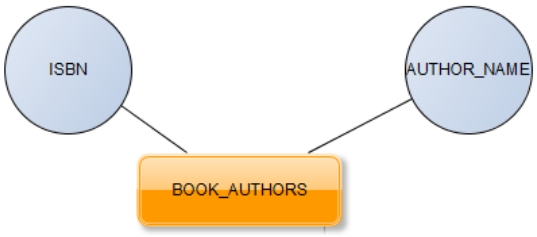


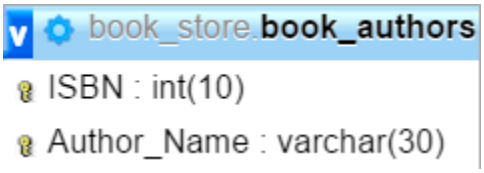
4) Book_Quantity :



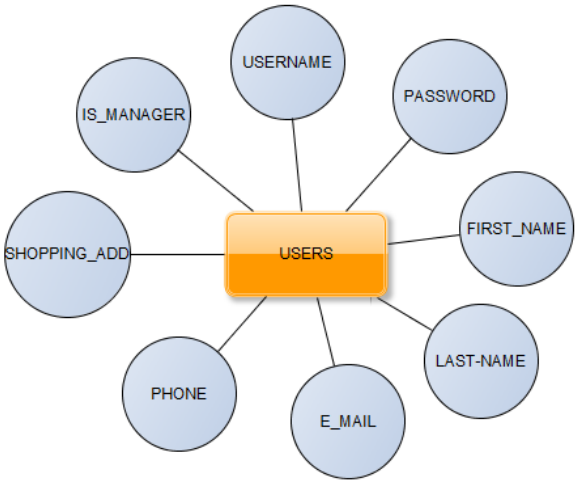


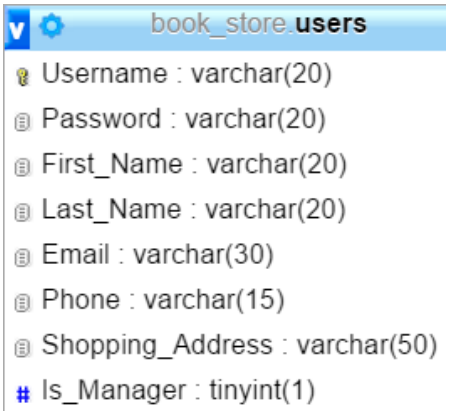
5) Book_authors :



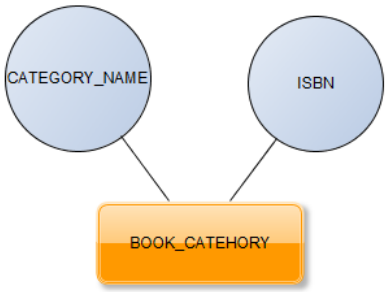


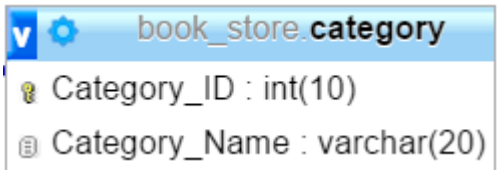
6) Users :



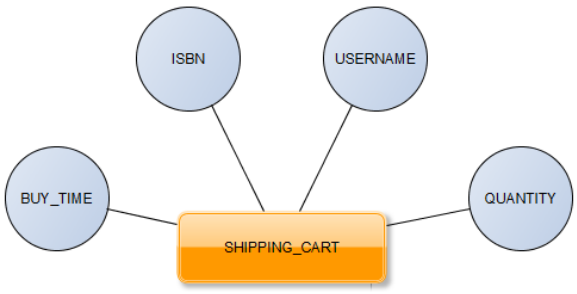


7) Category :

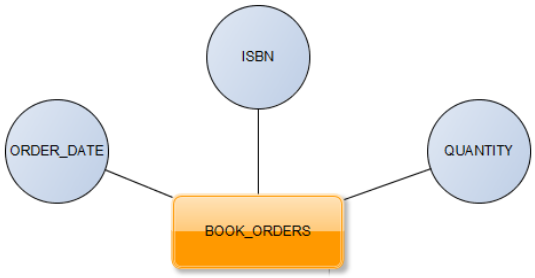


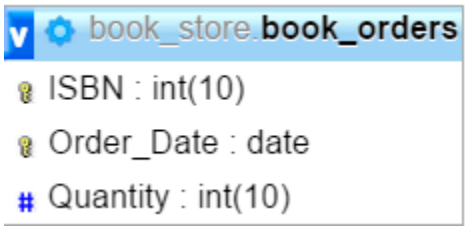


8) Shipping_cart :

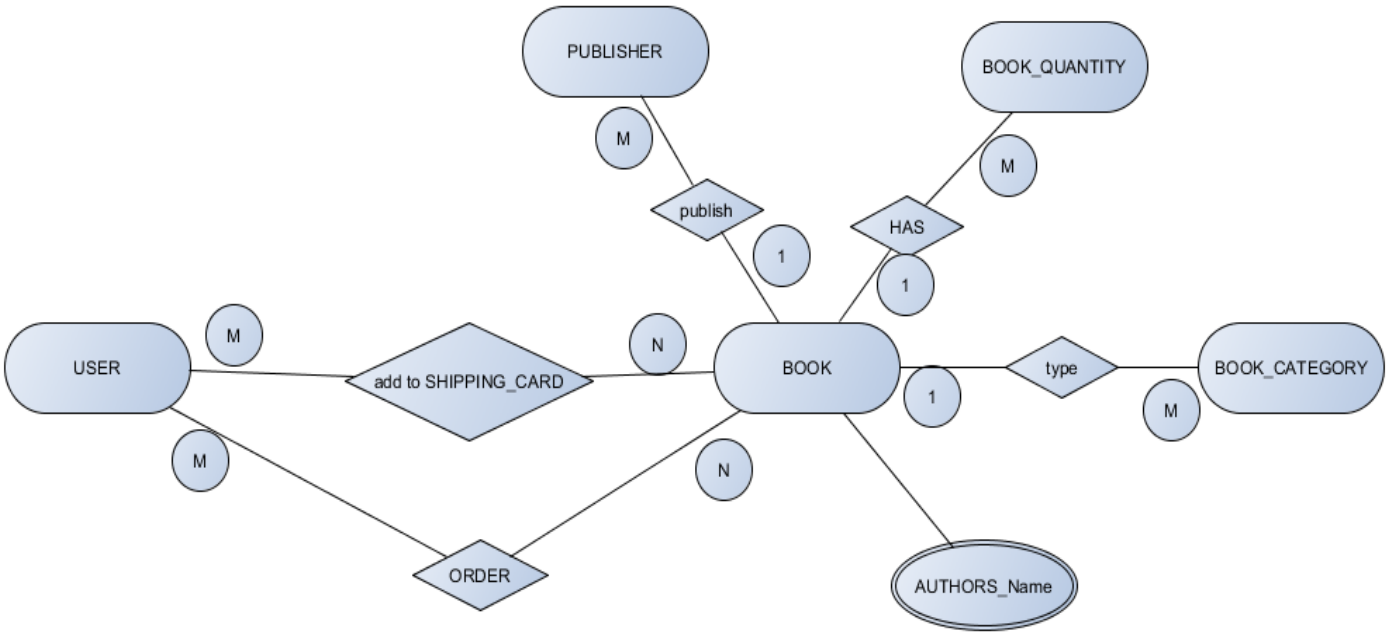


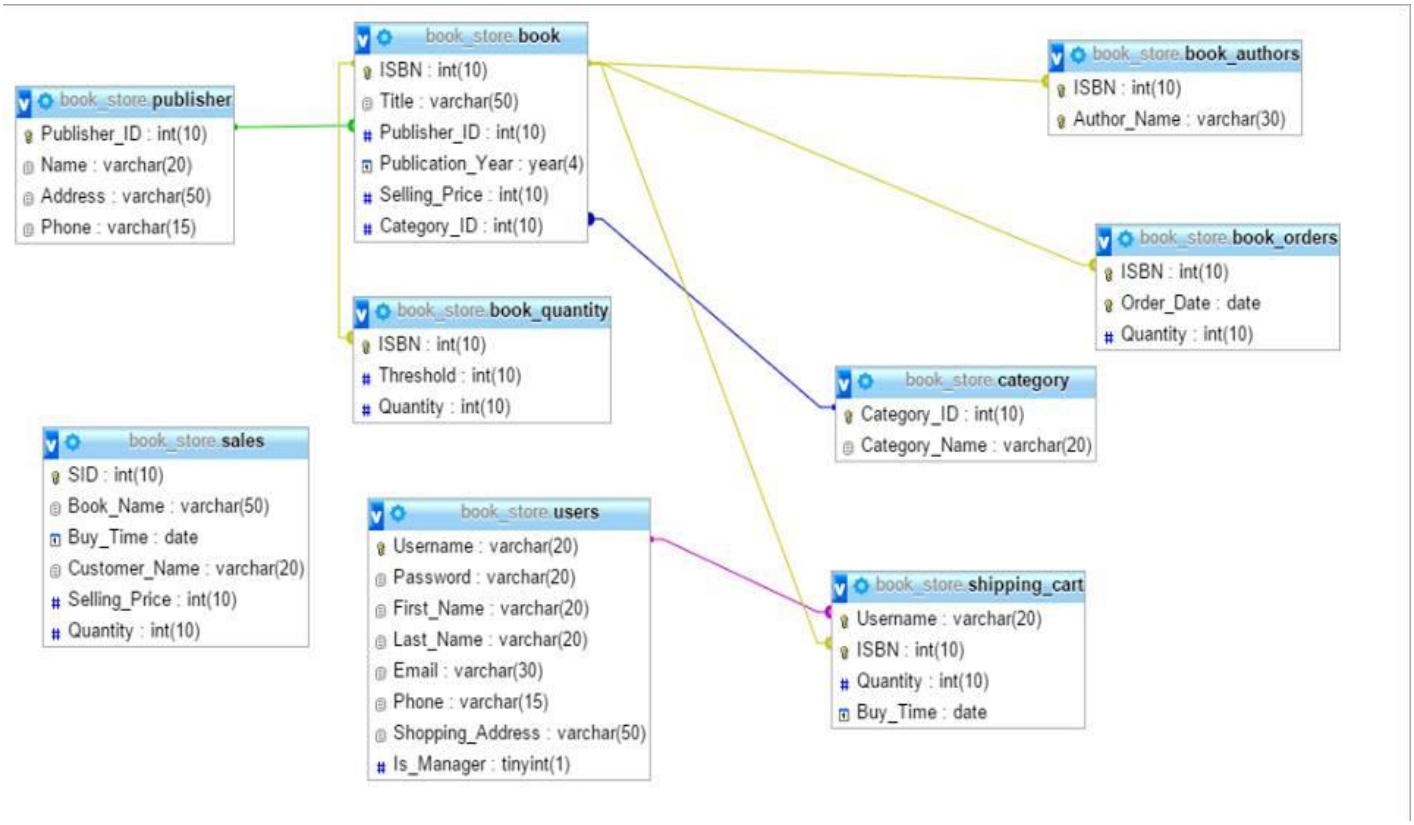
9) Book_orders :





ERD:


PUBLISHER
BOOK_QUANTITY
M
M
publish
HAS
1
1
USER
M
add to SHIPPING_CARD
N
BOOK
type
BOOK_CATEGORY
1
M
M
N
ORDER
AUTHORS_Name


book_store.publisher
Publisher_ID : int(10)
Name : varchar(20)
Address : varchar(50)
Phone : varchar(15)
book_store.book
ISBN : int(10)
Title : varchar(50)
Publisher_ID : int(10)
Publication_Year : year(4)
Selling_Price : int(10)
Category_ID : int(10)
book_store.book_authors
ISBN : int(10)
Author_Name : varchar(30)
book_store.book_orders
ISBN : int(10)
Order_Date : date
Quantity : int(10)
book_store.book_quantity
ISBN : int(10)
Threshold : int(10)
Quantity : int(10)
book_store.category
Category_ID : int(10)
Category_Name : varchar(20)
book_store.sales
SID : int(10)
Book_Name : varchar(50)
Buy_Time : date
Customer_Name : varchar(20)
Selling_Price : int(10)
Quantity : int(10)
book_store.users
Username : varchar(20)
Password : varchar(20)
First_Name : varchar(20)
Last_Name : varchar(20)
Email : varchar(30)
Phone : varchar(15)
Shopping_Address : varchar(50)
Is_Manager : tinyint(1)
book_store.shipping_cart
Username : varchar(20)
ISBN : int(10)
Quantity : int(10)
Buy_Time : date

2) Triggers:

1) This trigger updates the quantity of the book before deleting it from table book orders this means the order is confirmed

```
CREATE TRIGGER `confirm_order` BEFORE DELETE ON
`book_orders`
 FOR EACH ROW
 BEGIN
        UPDATE `book_quantity`
   SET Quantity=OLD.QUANTITY+`book_quantity`.`Quantity`
  WHERE ISBN=OLD.ISBN;
END
```

2)We create this trigger to prevent the quantity of any book to be negative and if it is negative we set it by the old quantity.

```
CREATE TRIGGER `Negative_Quantity` BEFORE UPDATE ON `book_quantity`
 FOR EACH ROW BEGIN
    IF (NEW.Quantity < 0) THEN
        set NEW.Quantity=OLD.Quantity;
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT='Stock will be negative';
    END IF;
END
```

3) we create this trigger after updating the book quantity in table book_quantity to insert new order in table book_orders and if there is old order with the same ISBN and oredr date we update the quantity only by multiplying it by 2 else we insert new order by quantity equals the threshold and the new quantity.

```
CREATE TRIGGER `PlaceOrder` AFTER UPDATE ON `book_quantity`
 FOR EACH ROW BEGIN
    IF (NEW.Quantity < OLD.Threshold) THEN
                    IF (SELECT count(*) FROM book_orders WHERE
ISBN=NEW.ISBN and Order_Date=CURDATE()) = 0 THEN
          insert into book_orders
values(NEW.ISBN,CURDATE(),NEW.Threshold+NEW.Quantity);
       else
          update book_orders set Order_Date=CURDATE(),
Quantity=Quantity*2 where ISBN=NEW.ISBN and
Order_Date=CURDATE();
           END IF;
    END IF;
END
```

3) Assumption:

In this project we assume the following assumptions and built our schema based on them whish are:

- **Every book has only one publisher, but it can have more than one author.**
- **Every book has one category.**
- **The user can be manager or customer. Customer can buy books and edit his personal information, the manager can do the same operations of the customer but he can add, modify, place order, confirm order, view users and make anyone of them a manager.**
- **The manager can modify any numbers of books but a specific book can be modified only by only one user at a time.**

4) Indexes:

We have added many indexes to some specific columns for two reasons:

- **The first for every primary key we made an index for it if we reference it by a foreign key in another table.**
- **The second for the column that the user can search by to get some records so we made indexes on them to retrieve the data as fast as we can.**

And these are the indexes we added to our database:

For BOOK table:

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	ISBN
Publisher_ID	BTREE	No	No	Publisher_ID
Category_ID	BTREE	No	No	Category_ID
Title	BTREE	No	No	Title

For author table:

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	ISBN
				Author_Name

For ORDER table:

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	ISBN
				Order_Date

For QUANTITY table:

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	ISBN

For CATEGORY table:

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	Category_ID
Category_Name	BTREE	No	No	Category_Name

For PUBLISHER table:

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	Publisher_ID
Name	BTREE	No	No	Name

For SHIPPING_CART table:

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	Username
				ISBN
Username	BTREE	No	No	Username
ISBN	BTREE	No	No	ISBN

For USERS table:

Keyname	Type	Unique	Packed	Column
PRIMARY	BTREE	Yes	No	Username