THE OHIO STATE UNIVERSITY

CSE 3241 Project Final Report

Jiahan Bao, Qisheng Wu, Zhaoyuan Yang, Zhe Huang, Ruiyang Liu

04/24/2016

1 Database Description

1.1 ER-model

The lastest ER-model is shown in Figure 1. Here's the modification history of the ER-model:

In checkpoint1, we create the original ER-model which has 7 entities: ShoppingCart, Customer, Order, Book, Warehouse, Publisher and Supplier.

In checkpoint2, the updates of ER-model include: (1) CreditCard, ShippingAddress and BillingAddress in Customer entity and Author in Book entity become multi-valued attributes; (2) Delete the relationship between Customer entity and Book entity, Add the relationship Ocontain and Scontain to connect Corder entity and ShoppingCart entity with Book entity; (3) Add Worder entity and three relationships to connect Warehouse entity, Supplier entity and Book entity; (4) Delete the relationship between Publisher entity and Supplier entity.

In checkpoint3, we modify some key attributes in ER-model.

In checkpoint4, there is no update.

And in final report, we delete the ShoppingCart entity.

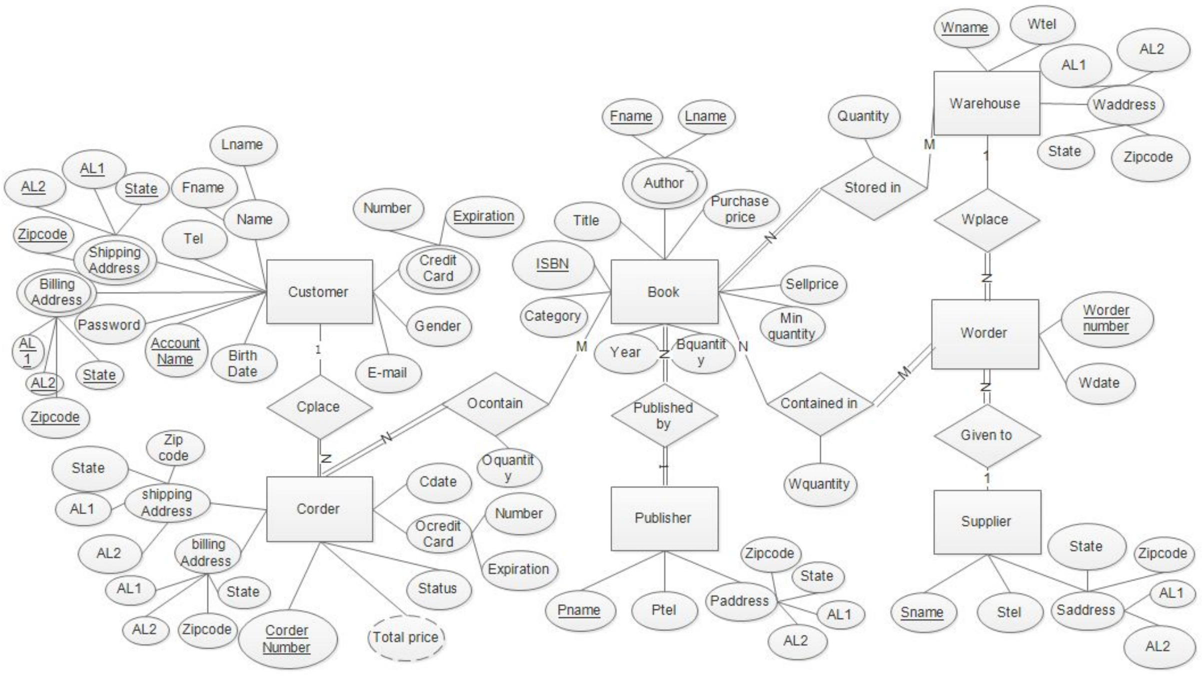


Figure 1: ER-model of the database

1.2 Relational Schema

1. Customer(<u>AccountName,</u>Fname,Lname,Tel,Password,BirthDate,Email,Gender)
No foreign key
Functional Dependencies:
AccountName→(Fname,Lname,Tel,Password,BirthDate,Email,Gender)
Email→AccountName

2. ShippingAddress (AccountName, Addressline1, Addressline2, State, Zipcode)
Foreign key **AccountName** references Customer
Functional Dependencies:
None

3. BillingAddress(AccountName, Addressline1, Addressline2, State, Zipcode)
Foreign key **AccountName** references Customer
Functional Dependencies:
None

4. CreditCard(AccountName, Number, Expiration)
Foreign key **AccountName** references Customer
Functional Dependencies:
None

5. Corder(OrderNumber, AccountName, ShippingZipcode, ShippingState, ShippingAddressline1, ShippingAddressline2, BillingZipcode, BillingState, BillingAddressline1, BillingAddressline2, Status, Cdate, CreditCardNumber, CreditCardExpiration)
Foreign key **AccountName** references Customer
Functional Dependencies:
OrderNumber→ (AccountName, ShippingZipcode, ShippingState, ShippingAddressline1, ShippingAddressline2, BillingZipcode, BillingState, BillingAddressline1, BillingAddressline2, Status, Cdate, CreditCardNumber, CreditCardExpiration)
CreditCardNumber→(CreditCardExpiration,BillingZipcode, BillingState, BillingAddressline1, BillingAddressline2)

6. Worder(WorderNumber, Wname, Sname, Wdate)
Foreign key **Wname** references Warehouse
Foreign key **Sname** references Supplier
Functional Dependencies:
WorderNumber→(Wname, Sname, Wdate)

7. Contained_in(ISBN, WorderNumber, WQuantity)
Foreign key **ISBN** references Book
Foreign key **WorderNumber** references Worder
Functional Dependencies:
(ISBN, WorderNumber)→WQuantity

8. Ocontain(OrderNumber, ISBN, OQuantity)
Foreign key **ISBN** references Book
Foreign key **OrderNumber** references Corder
Functional Dependencies:
(OrderNumber,ISBN)→Qquantity

9. Book(ISBN, Pname, Title, PurchasePrice,SellPrice, MinimumQuantity, Bquantity, Year, Category)
Foreign key **Pname** references Publisher

Functional Dependencies:
ISBN→(Pname, Title, PurchasePrice,SellPrice, MinimumQuantity, Bquantity, Year, Category)

10. Warehouse(<u>Wname</u>, Wtel, Zipcode, Addressline1, Addressline2, State)
Functional Dependencies:
Wname→(Wtel, Zipcode, Addressline1, Addressline2, State)

11. Publisher(<u>Pname</u>, Ptel, Zipcode, Addressline1, Addressline2, State)
Functional Dependencies:
Pname→(Ptel, Zipcode, Addressline1, Addressline2, State)

12. Supplier(<u>Sname</u>, Stel, Zipcode, Addressline1, Addressline2, State)
Functional Dependencies:
Sname→(Stel, Zipcode, Addressline1, Addressline2, State)

13. Stored_in(<u>ISBN</u>, quantity, <u>Wname</u>)
Foreign key **ISBN** references Book
Foreign key **Wname** references Warehouse
Functional Dependencies:
(ISBN, Wname)→quantity

14. Author(<u>ISBN</u>, <u>Fname</u>,<u>Lname</u>)
Foreign key **ISBN** references Book
Functional Dependencies:
None

1.3 Normalization

1. Customer:

This table is in BCNF.

2. ShippingAddress:

This table is in BCNF.

3. BillingAddress:

This table is in BCNF.

4. CreditCard:

This table is in BCNF.

5. Corder:

This table is in second normal form because credit number can also determine the corresponding credit information; however, there is no need to further normalize corder table into 3rd normal form because in corder table, it will be meaningless if we just want to update or add credit information without having an order number.

6. Worder:

This table is in BCNF.

7. Contained_in:

This table is in BCNF.

8. Ocontain:

This table is in BCNF.

9. Book:

This table is in BCNF.

10. Warehouse:

This table is in BCNF.

11. Publisher:

This table is in BCNF.

12. Supplier:

This table is in BCNF.

13. Stored_in:

This table is in BCNF.

14. Author:

This table is in BCNF.

1.4 Indexes

1. CREATE INDEX category_index ON BOOK (Category);

This index is used to return books' information according to the category that customers are interested in. Because when customers enter the online book store, in most cases, they just decide the category of the book that they want to buy instead of a specific book. Through this search, they can narrow the range of browsing.

2. CREATE INDEX sellprice_index ON BOOK (Sellprice ASC);

This index will return books' information according to the sell price. Those books' sell price will from cheap to expensive. This index can be combined with last one. When customers get the category that they are interested in, they may make the decision based on sell price. Customers can just arrange books according to ascending order or set a price range that they prefer.

3. CREATE INDEX accountname_index ON Corder (AccountName);

This index will return the order history of customers. For customers, this index can help them check their order history. For the seller of book store, it can help them understand the buying habits of different customers.

4. CREATE INDEX author_index ON AUTHOR (Fname, Lname);

This index is used to order authors according to them name. This index is also used to help customers to pick up books which they are interested in. If the customer is a fan of one author, he or she may want to find all books written by that author. This index can return the all ISBN that related to that author.

1.5 Views

1. Purchasing_behavoir:

- Description:

  List the quantity of books which belong to a specific category purchased by each user. This view is very useful when analysts want to see the statistic data of users' purchasing behavior.

- Relational algebra expression:

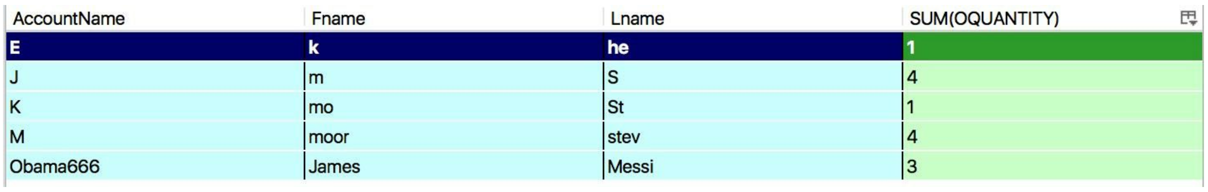
  $\pi_{\text{accountName, Fname, Lname, sum(OQuantity)}}$ (accountName $\mathcal{F}$ SUM $_{\text{OQuantity}}$ (ORDER $\bowtie_{\text{Ordernumber=Ordernumber}}$ ($\sigma_{\text{category='CATEGORY'}}$(BOOK)$\bowtie_{\text{ISBN=ISBN}}$ OCONTAIN)))

- SQL statements:

```
CREATE VIEW purchasing_behavoir
AS SELECT CU.AccountName, CU.Fname, CU.Lname, SUM(OQUANTITY)
FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO
WHERE CU.AccountName = CO.AccountName AND OC.OrderNumber =
CO.OrderNumber AND OC.ISBN = BO.ISBN AND BO.Category = 'CATEGORY'
GROUP BY CU.AccountName;
```

- Sample output:

CATEGORY = 'fiction'. Display the quantity of fiction books purchased by each user and corresponding user information.



AccountName	Fname	Lname	SUM(OQUANTITY)
E	k	he	1
J	m	S	4
K	mo	St	1
M	moor	stev	4
Obama666	James	Messi	3

Figure 2: Sample output of purchasing_behavoir view

2. Popular_book:

- Description:

  List the information and the sales of each book purchased by women. We can see which books are popular among women readers.

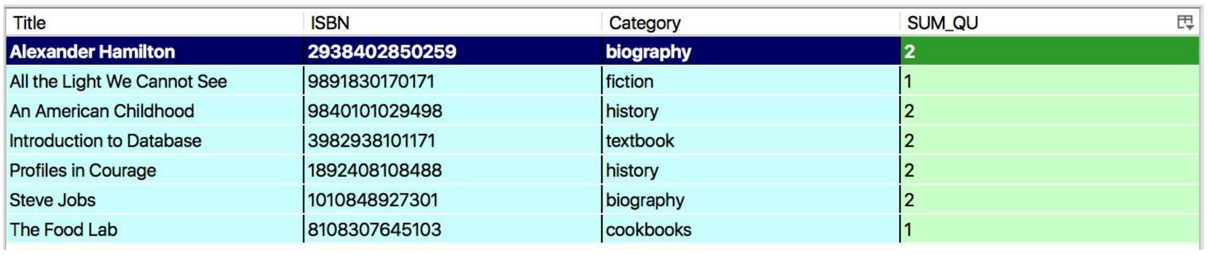
- Relational algebra expression:

  $\pi_{\text{Titile, ISBN, Category, Sum(OQuantity)}}$(Title$\mathcal{F}$ SUM $_{\text{OQuantity}}$(CORDER $\bowtie$ Ordernumber=Ordernumber BOOK $^{\bowtie}$ISBN=ISBN $^{\text{OCONTAIN}}$))

- SQL statements:

```
CREATE VIEW POPULAR_BOOK
AS SELECT BO.TITLE, BO.ISBNm, BO.CATEGORY, SUM(Oquantity) AS SUM_QU
FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO
WHERE CU.ACCOUNTNAME= CO.ACCOUNTNAME AND
OC.ORDERNUMBER=CO.ORDERNUMBER AND OC.ISBN=BO.ISBN AND CU.gender='f'
GROUP BY BO.TITLE;
```

- Sample output:



Title	ISBN	Category	SUM_QU
Alexander Hamilton	2938402850259	biography	2
All the Light We Cannot See	9891830170171	fiction	1
An American Childhood	9840101029498	history	2
Introduction to Database	3982938101171	textbook	2
Profiles in Courage	1892408108488	history	2
Steve Jobs	1010848927301	biography	2
The Food Lab	8108307645103	cookbooks	1

Figure 3: Sample output of popular_book view

1.6 Transactions

1. A customer orders a book

Description of Unit:

When customer order a book, the DBMS need to insert a new order into Corder, insert a new contain into Ocontain and update the number for Bquantity in Book.

SQL Code:

```
begin transaction
insert into CORDER
values( '188', 'yyy', '43202', 'OH', '566 Harley Dr. ', Apt 2', '43202', 'OH', '566 Harley Dr.',
'Apt2', '1', '4/24/16', '123456789954632', '1120');
insert into OCONTAIN
values ( '188', '123456789954632','1');
update BOOK
set BQuantity = BQuantity -1
where ISBN = '123456789954632';
commit;
end transaction;
```

2. Add a New Book with a New Publisher

Description of Unit:

When Add a new book with a new publisher , the DBMS need to insert a new publisher into Publisher, insert a new book into Book. Note that the Pname in Book is a foreign key referring to Publisher, so we need to insert Publisher first.

SQL Code:

```
begin transaction
insert into PUBLISHER
values( 'osu', '6142643308', '43270', '566 Neil Ave ', Apt 2', 'OH');
insert into BOOK
values ('123456789954632', 'osu', 'database',  ' 6.14', ' 8.88', ' 6', '14',  '2010', 'textbook');
commit;
end transaction;
```

3. Add a New Book by a New Author

Description of Unit:

When Add a new book by a new author, the DBMS need to insert a new publisher into Publisher, insert a new book into Book. Note that the Pname in Book is a foreign key referring to Publisher, so we need to insert Publisher first. The ISBN in the Author is a foreign key refering to the Book so we need to insert Author after Book.

SQL Code:

```
begin transaction
insert into PUBLISHER
```

```
values( 'osu', '6142643308', '43270', '566 Neil Ave ', Apt 2', 'OH');
insert into BOOK
values ('123456789954632', 'osu', 'database',  ' 6.14', ' 8.88', ' 6', '14',  '2010', 'textbook');
insert into AUTHOR
values ('123456789954632', 'Kelli', 'Steve');
Commit;
end transaction;
```

2 User Manual

2.1 Description of Tables

1. Customer:

Account Name	Fname	Lname	Tel	Password	BirthDate	E-mail	Gender

Table customer represents the basic information of the bookstore's customers. This table includes 8 attributes.

- AccountName represents the account of the user, it is the primary key of this table, which means each customer can only have one account name.We set the data type to varchar with the maximum number of 15, and set it to not null.
- Fname and Lname here represent the name of the customer which is not null and the types of them are both varchar with the maximum number of 15.
- We use the Tel attribute in the table to represent the telephone information of the customer, the type of this attribute is varchar with the maximum number of 15 and constrained by not null.
- Password is used to represent the password which the customer use when logging into the online bookstore website, the type is varchar with the maximum number of 9 and constrained by not null.
- Birthdate here is a date attribute represents the DOB of the users.
- Gender is the gender of the users with a type of char(1).
- Email is used to record the email address of customers and the type is varchar(20).

2. ShippingAddress:

AccountName	state	Addressline1	Addressline2	zipcode

Table ShippingAddress represents the address that books should be shipped to. This table includes 5 attributes.

- AccountName represent the account of customer, it is a foreign key refering to customer entity, and is a key attribute in this table, the datatype of the AccountName is varchar with maximum length of 25 and is set to not null.
- State represents the name of state, and it is a primary key attribute of this table.The datatype of state is varchar(20) and is set to not null.

- Addressline1 represents the name of street or road and the number of the house. It is a primary key attribute of this table.The datatype is varchar(150) and cannot be null.
- Addressline2 represents the number of the apartment. It is also a primary key attribute of this table. The datatype is varchar(150) and cannot be null.
- Zipcode represents the zipcode of address above. It is a primary key attribute of this table. The datatype is int and can be null.

3. BillingAddress:

AccountName	state	Addressline1	Addressline2	zipcode

Table BillingAddress represents the address of customer's bill. This table includes 5 attributes.

- AccountName represent the account of customer, it is a foreign key refering to customer entity, and is a key attribute in this table, the datatype of the AccountName is varchar with maximum length of 25 and is set to not null.
- State represents the name of state, and it is a primary key attribute of this table.The datatype of state is varchar(20) and is set to not null.
- Addressline1 represents the name of street or road and the number of the house. It is a primary key attribute of this table.The datatype is varchar(150) and cannot be null.
- Addressline2 represents the number of the apartment. It is also a primary key attribute of this table. The datatype is varchar(150) and cannot be null.
- Zipcode represents the zipcode of address above. It is a primary key attribute of this table. The datatype is int and can be null.

4. CreditCard:

AccountName	Number	Expiration

Table CreditCard represents the information of customer's credit card. This table includes 3 attributes.

- AccountName represent the account of customer, it is a foreign key refering to customer entity, and is a key attribute in this table, the datatype of the AccountName is varchar with maximum length of 25 and is set to not null.
- Number represents the number of credit card, and it is a primary key attribute of this table. The datatype of Number is int and is set to not null.
- Expiration represents the expiration time of credit card. It is also a primary key attribute of this table. The datatype is timestamp and the value is also not null.

5. Corder:

OrderNu	Account	Shipping	Shipping	ShippingAd	shippingAd	billingzip	billings	billingAdd	billingAdd	Sta	Cda	CreditCar	CreditCardE

mb er	Na me	Zipc ode	sta te	dressl ine1	dressl ine2	co de	ta te	ressli ne1	ressli ne2	t u s	t e	dNu mber	xpirat ion

Table Corder represents orders of the bookstore's customers. This table includes 14 attributes.

- OrderNumber represents the sequence number of orders, it is the primary key of this table. We set the datatype to be int and not null.
- AccountName represent the account of customer, it is a foreign key refering to customer entity, and is a key attribute in this table, the datatype of the AccountName is varchar with maximum length of 25 and is set to not null.
- Shippingstate represents the state that orders should be shipped to.The datatype of state is varchar(20) and is set to not null.
- ShippingAddressline1 represents the street or road that orders should be shipped to.The datatype is varchar(150) and cannot be null.
- Shippingsddressline2 represents the apartment that orders should be shipped to. The datatype is varchar(150) and cannot be null.
- Shippingzipcode represents the zipcode of shipping address above. The datatype is int and can be null.
- Billingstate represents the state of customer's bill.The datatype of state is varchar(20) and is set to not null.
- BillingAddressline1 represents the street or road of customer's bill.The datatype is varchar(150) and cannot be null.
- Billingsddressline2 represents the apartment of customer's bill. The datatype is varchar(150) and cannot be null.
- Billingzipcode represents the zipcode of billing address above. The datatype is int and can be null.
- Status represents the status of the order which including in the processing, canceled, shipping and finished. The datatype is varchar with the maximum number of 30 and not null.
- Cdate represents the date of the order. The datatype is timestamp.
- CreditcardNumber represents the number of credit card for this order, and it is a primary key attribute of this table. The datatype of CreditcardNumber is int and is set to not null.
- CreditcardExpiration represents the expiration time of credit card for this order. The datatype is timestamp.

6. Worder:

Worder number	Wname	Sname	Wdate

Table Worder represents the order which the warehouse place to the supplier. This table including 4 attributes.

- Wordernumber represents the sequence number of the order of warehouse. It is a primary key in this table and its datatype is int.
- Wname represents the warehouse that places the order, It is a foreign key referring to Table Warehouse. The datatype is varchar(15).

- Sname represents the supplier that receives this order, It is a foreign key referring to Table Supplier. The datatype is varchar(15) too.
- Wdate represents the date of warehouse's order. The datatype is timestamp.

7. Contained_in:

<u>ISBN</u>	<u>Worder number</u>	WQuantity

Table Contained_in models the relationship between Book entity and Worder entity. This table includes 3 attributes.

- ISBN represent the ISBN of book, it is a foreign key refering to BOOK entity, and is a key attribute in this table, the datatype of the ISBN is varchar with maximum length of 25 and is set to not null.
- Worder_number represent the order number from warehouse to supplier, it is a key attribute of the Contained_in table. The datatype of Worder_number is int and is set to not null.
- WQuantity represents the quantity of each book contained in the order, the datatype is int.

8. Ocontain:

<u>OrderNumber</u>	<u>ISBN</u>	OQuantity

Table Ocontain models the relationship between Corder and BOOK. This table includes 3 attributes.

- OrderNumber is the order number of the order placed by the customer, it is a foreign key referring to OrderNumber in Corder, and it a key attribute in Ocontain. The datatype is int and is set to not null.
- ISBN represent the ISBN of a book, it is a foreign key referring to ISBN in BOOK entity. The datatype is varchar with maximum length of 25, and is set to not null.
- OQuantity represents the quantity of each book in the order, and the datatype is int.

9. Book:

<u>ISBN</u>	Pname	Title	Purchase price	Sellprice	Minimum quantity	BQuantity	Year	Category

Table Book represents the basic information of the books stored in the bookstore. This table includes 9 attributes.

- ISBN represents the ISBN number of a book, it is the primary key. The datatype is varchar with maximum length of 25, and is set to not null.

- Pname represents the publisher name of the book, it is a foreign key referring to Pname in Publisher. .The datatype is varchar with maximum length of 15, and is set to not null.
- Title represents the title of the book. The datatype is varchar with maximum length of 15, and is set to not null.
- Purchase price represents the price of each book bought by the bookstore from suppliers. The datatype is double.
- Sell price is the price of each book sold by the bookstore. The datatype is double and is set to not null.
- Minimumquantity represents the mimimum quantity of each book stored in the warehouse. The datatype is double.
- BQuantity represents the current quantity of each book stored in the warehouse. The datatype is int.
- Year represents the publish year of each book. The datatype is int and is set to not null.
- Category represents the category of each book. The datatype is varchar with maximum length of 15.

10. Warehouse:

<u>Wname</u>	Wtel	zipcode	Addressline1	Addressline2	state

Table warehouse represents the basic information of warehouses where the books stored in.

- Wname represents the name of the warehouse and is the primary key. The datatype is varchar with maximum length of 15 and is set to not null.
- Zipcode represents the zipcode of the warehouse. The datatype is int.
- Addressline1 represents the first line of the warehouse address. The datatype is varchar with the maximum length of 150 and is set to not null.
- Addressline2 represents the second line of the warehouse address. The datatype is varchar with the maximum length of 150.
- State represents the state of the warehouse. The datatype is varchar with the maximum length of 20 and is set to not null.

11. Publisher:

<u>Pname</u>	Ptel	zipcode	Addressline1	Addressline2	state

Table publisher represents the basic information of the publisher of each book.

- Pname represents the name of the publisher and is the primary key. The datatype is varchar with the maximum length of 15.
- Ptel represents the telephone number of the publisher. The datatype is varchar with the maximum length of 15.
- Zipcode represents the zipcode of the publisher. The datatype is int.
- Addressline1 represents the first line of the publisher address. The datatype is varchar with the maximum length of 150.

- Addressline2 represents the second line of the publisher address. The datatype is varchar with the maximum length of 150.
- State represents the state of the warehouse. The datatype is varchar with the maximum length of 20.

12. Suppliers:

<u>Sname</u>	Stel	zipcode	Addressline1	Addressline2	state

Table Suppliers represents the basic information of book suppliers. This table includes 6 attributes.

- Sname represents the name of the supplier. It is the primary key of this table and each of the supplier's name should be different. Its datatype is varchar with maximum length of 15 and it is set to not null.
- Stel represents the contact telephone number of the supplier. The type of Stel is varchar with maximum length of 15 and it can be null.
- Zip code represents the zip code of the supplier. It has a datatype of int and it can be null.
- Address line 1 represents the first line of supplier's address, specifically the street and the house number. Its datatype is varchar with maximum length of 150 and it is set to not null.
- Address line 2 represents the second line of supplier's address, specifically the number of the apartment. Its datatype is varchar with maximum length of 150 and it can be null.
- State represents the name of state. The datatype is varchar(20). It cannot be null.

13. Store_in:

<u>ISBN</u>	quantity	<u>Wname</u>

Table store_in models the relationship between Book entity and Warehouse entity. It has 3 attributes.

- ISBN represents the ISBN of the book, which is not only a key attribute of this table but also a foreign key refering to Book entity. It is varchar(25) and cannot be null.
- Quantity represents the quantity of the book which has a specific ISBN in a specific warehouse. It is datatype is int.
- Wname represents the name of the warehouse. It is a key attribute of this table and a foreign key refering to Warehouse entity. It is varchar(15) and cannot be null.

14. Author:

<u>ISBN</u>	<u>Lname</u>	<u>Fname</u>

Table Author models the author of the book. It has 3 attributes.

- ISBN represents the ISBN of the book, which is not only a key attribute of this table but also a foreign key refering to Book entity. It is varchar(25) and cannot be null.
- Lname represents the last name of the author. It is a key attribute and has a datatype of varchar(15). It is set to not null.
- Fname represents the first name of the author. It is a key attribute and has a datatype of varchar(15). It is set to not null.

2.2 Sample SQL Queries

2.2.1 Checkpoint2 :

a. Find the titles of all books by Pratchett that cost less than $10

$\pi_{\text{title}}(\sigma_{\text{Sellprice < 10}}(\text{BOOK} \bowtie_{\text{ISBN = ISBN}} (\sigma_{\text{Lname = 'Pratchett'}}(\text{AUTHOR}))))$

SELECT title
FROM Author AS a, Book AS b
WHERE Lname='Pratchett' AND a.ISBN=b.ISBN And b.sellprice<10;

b. Give all the titles and their dates of purchase made by a single customer (you choose how to designate the customer)

$\pi_{\text{Title, Date}}(((\sigma_{\text{accountName='Obama666'}}(\text{ORDER})) \bowtie_{\text{Ordernumber=Ordernumber}} \text{OCONTAIN}) \bowtie_{\text{ISBN=ISBN}} \text{BOOK})$

SELECT title, CDate
FROM BOOK AS b, OCONTAIN AS oc, CORDER AS c
WHERE c.accountName='Obama666' AND c.Ordernumber=oc.Ordernumber AND oc.ISBN=b.ISBN;

c. Find the titles and ISBNs for all books with less than 5 copies in stock

$\pi_{\text{title,ISBN}}(\sigma_{\text{Bquantity<5}}(\text{BOOK}))$

SELECT title,ISBN
FROM BOOK
WHERE Bquantity<5;

d. Give all the customers who purchased a book by Pratchett and the titles of Pratchett books they purchased (8,9,10)

$\pi_{\text{Title,Accou}}(\text{ORDER} \bowtie_{\text{Ordernumber=Ordernumber}} (\sigma_{\text{Lname='Pratchett'}}(\text{BOOK}) \bowtie_{\text{ISBN=ISBN}} \text{OCONTAIN}))$

SELECT title,AccountName
FROM CORDER AS o, BOOK AS b, OCONTAIN AS oc, AUTHOR as a
WHERE oc.ISBN=b.ISBN AND a.Lname='Pratchett' AND o.Ordernumber=oc.Ordernumber AND a.ISBN=b.ISBN;

e. Find the total number of books purchased by a single customer (you choose how to designate the customer)

$\mathcal{F}$ SUM $_{\text{oQuantity}}$ (($\sigma_{\text{accountName='Obama012'}}$(CORDER)) $\bowtie_{\text{Ordernumber=Ordernumber}}$ OCONTAIN)

```
SELECT SUM(OQuantity)
FROM CORDER AS c, Ocontain AS o
WHERE c.AccountName='Obama666' AND c.Ordernumber=o.Ordernumber;
```

f. Find the customer who has purchased the most books and the total number of books they have purchased

accountName $\mathcal{F}$ MAX $_{\text{OQuantity}}$ (accountName $\mathcal{F}$ SUM $_{\text{OQuantity}}$ (CORDER $\bowtie_{\text{Ordernumber=Ordernumber}}$ OCONTAIN))

```
SELECT AccountName,MAX(A)
FROM (SELECT AccountName, SUM(OQuantity) AS A
        FROM Corder AS c, Ocontain AS o
        WHERE c.OrderNumber=o.OrderNumber
        GROUP BY AccountName);
```

2.2.2 Checkpoint3:

a. Find the titles of all books by Pratchett that cost less than $10

A ← $\sigma_{\text{Lname = 'Pratchett'}}$(Author)
B ← Book $\bowtie_{\text{ISBN = ISBN}}$(A)
S ← $\sigma_{\text{sellprice < 10}}$ (B)
T ← π_{Title}(S)

```
SELECT Title
FROM Book AS b, Author AS a
WHERE a.Lname='Pratchett' AND a.ISBN=b.ISBN AND Sellprice<10;
```

b. Give all the titles and their dates of purchase made by a single customer (you choose how to designate the customer)

C ← $\sigma_{\text{Lname='ja' AND Fname='a'}}$(Customer)
CO ← Corder$\bowtie_{\text{AccountName = AccountName}}$ (C)
OC ← OContain$\bowtie_{\text{OrderNumber = OrderNumber}}$(CO)
B ← Book$\bowtie_{\text{ISBN = ISBN}}$ (OC)
Result ← $\pi_{\text{Cdate, Title}}$ (B)

```
SELECT Cdate, Title
```

FROM Customer AS c, Corder AS co, Book AS b, Ocontain AS oc
WHERE c.Lname='jam' AND c.Fname='ad' AND co.AccountName=c.AccountName AND co.OrderNumber=oc.OrderNumber AND oc.ISBN=b.ISBN;

c. Find the titles and ISBNs for all books with less than 5 copies in stock

$\pi_{\text{Title, ISBN}} (\sigma_{\text{Bquantity} < 5} (\text{Book}))$

```
SELECT Title, ISBN
FROM Book
WHERE Bquantity<5;
```

d. Give all the customers who purchased a book by Pratchett and the titles of Pratchett books they purchased

Pratchett $\leftarrow \sigma_{\text{Lname = 'Pratchett'}}$ (Author)
R $\leftarrow \pi_{\text{ISBN}}$ (Pratchett)
B $\leftarrow$ Book* (R)
OC $\leftarrow$ Ocontain* (B)
CO $\leftarrow$ Corder* (OC)
C $\leftarrow$ Customer* (CO)
Result $\leftarrow \pi_{\text{Lname, Fname, Title}}$ (C)

```
SELECT c.Lname, c.Fname, b.Title
FROM customer AS c, Book as b, Corder AS co, Ocontain AS oc, Author as a
WHERE a.Lname='Pratchett' AND a.ISBN=b.ISBN AND oc.ISBN=b.ISBN AND
oc.OrderNumber=co.OrderNumber AND co.AccountName=c.AccountName;
```

e. Find the total number of books purchased by a single customer (you choose how to designate the customer)

C $\leftarrow \sigma_{\text{Lname = 'ja' AND Fname = 'a'}}$ (Customer)
CO $\leftarrow$ Corder$\bowtie_{\text{AccountName = AccountName}}$ (C)
OC $\leftarrow$ Ocontain $\bowtie_{\text{OrderNumber = OrderNumber}}$ (CO)
Result $\leftarrow F_{\text{SUM(OQuantity)}}$ (OC)

```
SELECT SUM(Oquantity)
FROM Customer AS c, Corder AS co, Ocontain AS oc
WHERE c.Lname='jam' AND c.Fname='ad' AND c.AccountName=co.AccountName AND
co.OrderNumber=oc.OrderNumber;
```

f. Find the customer who has purchased the most books and the total number of books they have purchased

Temp1 $\leftarrow$ Customer * (Corder)
Temp2 $\leftarrow$ Ocontain * (Temp1)
Temp3 $\leftarrow$ AccountName $F_{\text{SUM(OQuantity)}}$ (Temp2)

Temp4 $\leftarrow \rho_{(m)} F_{MAX(SUM\_OQuantity)}$(Temp3)
Temp5 $\leftarrow$(Temp3)$\bowtie_{SUM\_OQuantity = m}$(Temp4)
Temp6 $\leftarrow$ Customer * (Temp5)
Result $\leftarrow \pi_{Lname,Fname,m}$(Temp6)

```
SELECT Lname, Fname, MAX(m)
FROM
(
SELECT SUM(Oquantity) AS m, c.Lname, c.Fname
FROM Customer AS c, Corder AS co, Ocontain AS oc
WHERE c.AccountName=co.AccountName AND co.OrderNumber=oc.OrderNumber
GROUP BY c.Lname, c.Fname
);
```

2.2.3 Checkpoint 2 and 3 additional interesting query:

a. List the quantity of fiction books purchased by Obama666.

$\mathcal{F}$ SUM $_{OQuantity}$ ($\sigma_{accountName='Obama666'}$(CORDER $\bowtie_{Ordernumber=Ordernumber}$
($\sigma_{category='fiction'}$(BOOK)$\bowtie_{ISBN=ISBN}$ OCONTAIN)))

```
SELECT SUM(OQUANTITY)
FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO
WHERE CU.AccountName= CO.AccountName AND OC.OrderNumber=CO.OrderNumber AND
OC.ISBN=BO.ISBN AND CU.AccountName='Obama666' AND BO.Category='fiction';
```

b. List the title of most popular book purchased by women.

Title$\mathcal{F}$ MAX $_{oquantity}$ ($\sigma_{Gender='female'}$(Customer) $\bowtie_{Accountname=Accountname}$ (AccountName$\mathcal{F}$ SUM $_{OQuantity}$(ORDER $\bowtie_{Ordernumber=Ordernumber}$ BOOK $\bowtie_{ISBN=ISBN}$ OCONTAIN)))

```
SELECT RE.TITLE, MAX (RE.SUM_QU)  AS SUM_FEMALE
FROM (SELECT TITLE, SUM(Oquantity) AS SUM_QU
FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO
WHERE CU.ACCOUNTNAME= CO.ACCOUNTNAME AND
OC.ORDERNUMBER=CO.ORDERNUMBER AND OC.ISBN=BO.ISBN AND CU.gender='f'
GROUP BY BO.TITLE) AS RE;
```

c. List the name of the most popular publisher.

Pname$\mathcal{F}$ MAX $_{oquantity}$(Pname$\mathcal{F}$ SUM $_{OQuantity}$(PUBLISHER $\bowtie_{PNAME=PNAME}$ (BOOK$\bowtie_{ISBN=ISBN}$ OCONTAIN))

```
SELECT RE.pname, MAX(RE.SUM_QU)
FROM (SELECT PU.pname, SUM(OC.Oquantity) AS SUM_QU
FROM  OCONTAIN AS OC, BOOK AS BO, PUBLISHER AS PU
WHERE OC.ISBN=BO.ISBN AND BO.pname=PU.pname
```

GROUP BY PU.pname) AS RE;

2.2.4 Advanced queries from checkpoint 3:

a. Provide a list of customer names, along with the total dollar amount each customer has spent.

Relational Algebra:

A1$\leftarrow\rho_{(ordernumber,totaloneach)}(\pi_{ORDERNUMBER,Oquantity*SELLPRICE}$ (OCONTAIN$\bowtie_{ISBN = ISBN}$ BOOK))
A2$\leftarrow\rho_{(ordernumber,eachordertotal)}$ordernumber$F_{SUM(totaloneach)}$ (A1)
A3$\leftarrow$A2$\bowtie_{ORDERNUMBER = ORDERNUMBER}$ CORDER
A4$\leftarrow$A3$\bowtie_{ORDERNUMBER = ORDERNUMBER}$ CUSTOMER
A5$\leftarrow\rho_{(ACCOUNTNAME,TOTAL)}$ACCOUNTNAME $F_{SUM(eachordertotal)}$ (A2)
RESULTA$\leftarrow\pi_{FNAME,LNAME,TOTAL}$ (A5$\bowtie_{ACCOUNTNAME = ACCOUNTNAME}$ CUSTOMER)

SQL:

```
SELECT CU.FNAME, CU.LNAME, SUM(TEMP.OTATAL) AS TOTAL_SPENT
FROM CUSTOMER AS CU, CORDER AS CO, (Select  OC.ORDERNUMBER,
SUM(OC.Oquantity*BO.SELLPRICE) AS OTATAL
FROM OCONTAIN AS OC, BOOK AS BO
WHERE OC. ISBN=BO.ISBN
GROUP BY OC.ORDERNUMBER) AS TEMP
WHERE CU.ACCOUNTNAME=CO.ACCOUNTNAME AND
TEMP.ORDERNUMBER=CO.ORDERNUMBER
GROUP BY CU.FNAME,CU.LNAME;
```

b. Provide a list of customer names and e-mail addresses for customers who have spent more than the average customer.

Relational Algebra:

B1$\leftarrow\rho_{(AVERAGESPENT)}F_{AVG\ TOTAL}$ (A5)
B2$\leftarrow$B1XA5
B3$\leftarrow\sigma_{TOTAL>TOTALSPENT}$(B2)
RESULTB$\leftarrow\pi_{FNAME,LNAME,EMAIL}$ (B3)

SQL:

```
SELECT CU.FNAME, CU.LNAME, CU.email
FROM CUSTOMER AS CU, CORDER AS CO, (Select  OC.ORDERNUMBER,
SUM(OC.Oquantity*BO.SELLPRICE) AS OTATAL
                        FROM OCONTAIN AS OC, BOOK AS BO
                        WHERE OC. ISBN=BO.ISBN
                        GROUP BY OC.ORDERNUMBER) AS TEMP
WHERE CU.ACCOUNTNAME=CO.ACCOUNTNAME AND
TEMP.ORDERNUMBER=CO.ORDERNUMBER
GROUP BY CU.FNAME,CU.LNAME
HAVING    SUM(TEMP.OTATAL)> (SELECT AVG(RE1.TOTAL_SPENT)
                                        FROM  ( SELECT  SUM(TEMP.OTATAL) AS TOTAL_SPENT
                                        FROM CUSTOMER AS CU, CORDER AS CO, (Select
OC.ORDERNUMBER, SUM(OC.Oquantity*BO.SELLPRICE) AS OTATAL
                                                   FROM OCONTAIN AS OC, BOOK AS BO
                                                   WHERE OC. ISBN=BO.ISBN
                                                   GROUP BY OC.ORDERNUMBER) AS TEMP
        WHERE CU.ACCOUNTNAME=CO.ACCOUNTNAME AND
TEMP.ORDERNUMBER=CO.ORDERNUMBER
        GROUP BY CU.FNAME,CU.LNAME) AS RE1) ;
```

c. Provide a list of the titles in the database and associated total copies sold to customers, sorted from the title that has sold the most individual copies to the title that has sold the least.

Relational Algebra:

C1$\leftarrow$ CORDER$\bowtie_{\text{ORDERNUMBER=ORDERNUMBER}}$(OCONTAIN$\bowtie_{\text{ISBN = ISBN}}$ BOOK)

RESULTC$\leftarrow\rho_{\text{(TITLE,TOTALCOPY)}}$TITLE $F_{\text{SUM(OQUANTITY)}}$ (C1)

SQL:

```
SELECT RE.TITLE, RE.SUM_QU
FROM (SELECT TITLE, SUM(Oquantity) AS SUM_QU
      FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO
      WHERE CU.ACCOUNTNAME= CO.ACCOUNTNAME AND
OC.ORDERNUMBER=CO.ORDERNUMBER AND OC.ISBN=BO.ISBN
      GROUP BY BO.TITLE) AS RE
ORDER BY RE.SUM_QU DESC;
```

d. Provide a list of the titles in the database and associated dollar totals for copies sold to customers, sorted from the title that has sold the highest dollar amount to the title that has sold the smallest.

Relational Algebra:

D1$\leftarrow$RESULTC$\bowtie_{\text{title=title}}$BOOK

RESULTD$\leftarrow\rho_{\text{(title,totalsale)}}$(π $_{\text{title,TOTALCOPY*SELLPRICE}}$ (D1))

SQL:

```
SELECT RE.TITLE, RE.TOTAL_SALE
FROM (SELECT TITLE, SUM(Oquantity)*BO.SELLPRICE AS TOTAL_SALE
      FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO
      WHERE CU.ACCOUNTNAME= CO.ACCOUNTNAME AND
OC.ORDERNUMBER=CO.ORDERNUMBER AND OC.ISBN=BO.ISBN
      GROUP BY BO.TITLE) AS RE
ORDER BY RE.TOTAL_SALE DESC;
```

e. Find the most popular author in the database (i.e. the one who has sold the most books)

Relational Algebra:

$E1 \leftarrow RESULTC \bowtie_{title=title} BOOK$
$E2 \leftarrow \rho_{(FNAME,LNAME,SALE)}((FNAME,LNAME) F_{SUM\ TOTALCOPY}\ (E1 \bowtie_{ISBN=ISBN} AUTHOR))$
$RESULTE \leftarrow (FNAME,LNAME) F_{MAX\ SALE}(E2)$

SQL:

```
SELECT RE.FNAME, RE.LNAME, MAX(RE.SUM_QU)
FROM (SELECT AU.FNAME,AU.LNAME, SUM(Oquantity) AS SUM_QU
          FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO, AUTHOR
AS AU
          WHERE CU.ACCOUNTNAME= CO.ACCOUNTNAME AND
OC.ORDERNUMBER=CO.ORDERNUMBER AND OC.ISBN=BO.ISBN AND AU.ISBN=BO.ISBN
         GROUP BY AU.FNAME, AU.LNAME) AS RE;
```

f. Find the most profitable author in the database for this store (i.e. the one who has brought in the most money)

Relational Algebra:

$F1 \leftarrow RESULTD \bowtie_{title=title} BOOK$
$F2 \leftarrow AUTHOR \bowtie_{ISBN=ISBN} F1$
$F3 \leftarrow \rho_{(title,FNAME,LNAME,TOTALPROFITEACH)}(\pi_{title,FNAME,LNAME,,totalsale*SELLPRICE}\ (F2))$
$F4 \leftarrow \rho_{(FNAME,LNAME,SALE)}(FNAME,LNAME) F_{SUM\ TOTALPROFITEACH}(F3)$
$RESULTF \leftarrow (FNAME,LNAME) F_{MAX\ SALE}(F4)$

SQL:

```
SELECT RE.FNAME,RE.LNAME, MAX(TOTAL_SALE)
 FROM (SELECT AU.FNAME,AU.LNAME, SUM(Oquantity)*BO.SELLPRICE AS TOTAL_SALE
          FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO, AUTHOR
AS AU
          WHERE CU.ACCOUNTNAME= CO.ACCOUNTNAME AND
OC.ORDERNUMBER=CO.ORDERNUMBER AND OC.ISBN=BO.ISBN AND AU.ISBN=BO.ISBN
          GROUP BY AU.FNAME, AU.LNAME) AS RE;
```

g. Provide a list of customer information for customers who purchased anything written by the most profitable author in the database.

Relational Algebra:

$G1 \leftarrow \pi_{ISBN}\ (RESULTF*AUTHOR*BOOK)$
$G2 \leftarrow G1*OCONTAIN*CORDER*CUSTOMER$
$RESULTG \leftarrow \pi_{Lname\ ,\ Fname}\ G2$

SQL:

```
SELECT DISTINCT CU1.LNAME,CU1.FNAME
 FROM CUSTOMER AS CU1,CORDER AS CO1, OCONTAIN AS OC1
 WHERE CU1.ACCOUNTNAME= CO1.ACCOUNTNAME AND
OC1.ORDERNUMBER=CO1.ORDERNUMBER AND OC1.ISBN IN (
       SELECT AU3.ISBN
```

```
        FROM (SELECT RE.FNAME,RE.LNAME, MAX(TOTAL_SALE)
                FROM (SELECT AU.FNAME,AU.LNAME, SUM(Oquantity)*BO.SELLPRICE AS
TOTAL_SALE
                    FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO,
AUTHOR AS AU
                    WHERE CU.ACCOUNTNAME= CO.ACCOUNTNAME AND
OC.ORDERNUMBER=CO.ORDERNUMBER AND OC.ISBN=BO.ISBN AND AU.ISBN=BO.ISBN
                    GROUP BY AU.FNAME, AU.LNAME) AS RE) AS RE2 INNER JOIN AUTHOR AS
AU3 ON RE2.FNAME=AU3.FNAME AND RE2.LNAME=AU3.LNAME);
```

h. Provide the list of authors who wrote the books purchased by the customers who have spent more than the average customer.

Relational Algebra:

H1← π_{ISBN} (RESULTB*CUSTOMER*CORDER*OCONTAIN*BOOK)
RESULTH← $\pi_{FNAME,LNAME}$(H1*AUTHOR)

SQL:

```
SELECT DISTINCT AU5.LNAME, AU5.FNAME
FROM AUTHOR AS AU5, (SELECT OC2.ISBN
            FROM CORDER AS CO2, OCONTAIN AS OC2,(SELECT CU.ACCOUNTNAME
                        FROM CUSTOMER AS CU, CORDER AS CO, (Select  OC.ORDERNUMBER,
SUM(OC.Oquantity*BO.SELLPRICE) AS OTATAL
                                    FROM OCONTAIN AS OC, BOOK AS BO
                                    WHERE OC. ISBN=BO.ISBN
                                    GROUP BY OC.ORDERNUMBER) AS TEMP
                        WHERE CU.ACCOUNTNAME=CO.ACCOUNTNAME AND
TEMP.ORDERNUMBER=CO.ORDERNUMBER
                        GROUP BY CU.FNAME,CU.LNAME
                        HAVING    SUM(TEMP.OTATAL)> (SELECT AVG(RE1.TOTAL_SPENT)
                                                FROM  ( SELECT  SUM(TEMP.OTATAL) AS
TOTAL_SPENT
                                                    FROM CUSTOMER AS CU, CORDER AS CO,
(Select  OC.ORDERNUMBER, SUM(OC.Oquantity*BO.SELLPRICE) AS OTATAL
                                                            FROM OCONTAIN AS OC,
BOOK AS BO
                                                            WHERE OC. ISBN=BO.ISBN
                                                            GROUP BY
OC.ORDERNUMBER) AS TEMP
                                                    WHERE
CU.ACCOUNTNAME=CO.ACCOUNTNAME AND TEMP.ORDERNUMBER=CO.ORDERNUMBER
                                                    GROUP BY CU.FNAME,CU.LNAME) AS RE1) )
AS AVGUSR
            WHERE AVGUSR.ACCOUNTNAME=CO2.ACCOUNTNAME AND
CO2.ORDERNUMBER=OC2.ORDERNUMBER) AS BOISBN
WHERE BOISBN.ISBN=AU5.ISBN;
```

2.3 INSERT syntax

1. Add a new book into database.

When we add a book into the database, we find that in our database entity Book has a foreign key Pname referring to the Publisher entity. So if we want to add a book into the book entity, we must first add a publisher into the publisher entity.

Insert syntax:

```
insert into publisher
values('Pname', 'Ptel', 'zipcode', 'addressline1', 'addressline2', 'state');

insert into book
values ('ISBN', 'Pname', 'Title', 'Purchaseprice', 'Sellprice', 'Minimumquantity', 'BQuantity',
'Year', 'Category' );
```

2. Add a publisher into database

Since there is no foreign key in the publisher table, so we can simply insert the new tuple into the publisher table.

Insert syntax:

```
insert into publisher
values('Pname', 'Ptel', 'zipcode', 'addressline1', 'addressline2', 'state');
```

3. Add an author into database

When we add an author into the database, we find that in our database entity Author has a foreign key ISBN referring to the Book entity and Book has a foreign key Pname referring to the Publisher entity. So if we want to add a author into the book entity, we must first add a publisher and book into the publisher entity.

```
insert into publisher
values('Pname', 'Ptel', 'zipcode', 'addressline1', 'addressline2', 'state');

insert into book
values ('ISBN', 'Pname', 'Title', 'Purchaseprice', 'Sellprice', 'Minimumquantity', 'BQuantity',
'Year', 'Category' );

insert into author
values('ISBN', 'Fname', 'Lname');
```

4. Add a customer into database

Since there is no foreign key in the customer table, so we can simply insert the new tuple into the customer table.

```
insert into customer
values('AccountName', 'Lname', 'Fname', 'Tel',' Password', 'Birthdate', 'Gender', 'Email');
```

2.4 DELETE syntax

1. Delete a new book

When we want to delete a new book from the table book, we should first delete tables in which foreign keys of these tables refer to the book table. As we know that in our database, Author has a foreign key ISBN referring to book, Stored_in has a foreign key ISBN referring to Book, Ocontain has a foreign key ISBN referring to Book. So we need to delete these table first then delete a book.

Delete syntax:

```
delete from Author
where ISBN = 'varchar(25)';

delete from Stored_in
where ISBN = 'varchar(25)';

delete from Ocontain
where ISBN = 'varchar(25)';
```

2. Delete a publisher into database

When we want to delete a new book from the table book, we should first delete tables in which foreign keys of these tables refer to the book table. As we know that in our database, Book has a foreign key Pname referring to book. So we need to delete this table first then delete a book.

Delete syntax:

```
delete from Book
where Pname = 'varchar(15)';

delete from publisher
where Pname = 'varchar(15)';
```

3. Delete an author into database

Since there is no table in our database that has a foreign key referring to the author table, so we can directly delete the author here.

delete syntax:

delete from Author
where Fname = 'varchar(15)' AND Lname = 'varchar(15)';

4. Delete a customer into database

When we want to delete a new book from the table book, we should first delete tables in which foreign keys of these tables refer to the book table. As we know that in our database, BillingAddress has a foreign key AccountName referring to Customer, Coder has a foreign key AccountName referring to Customer, CreditCard has a foreign key AccountName referring to Customer, Ocontain has a foreign key AccountName referring to Customer, Shippingaddress has a foreign key AccountName referring to Customer. So we need to delete this table first then delete a book.

Delete syntax:

Delete from BillingAddress
where AccountName = 'varchar(15)';

Delete from Corder
where AccountName = 'varchar(15)';

Delete from CreditCard
where AccountName = 'varchar(15)';

Delete from Shippingaddress
where AccountName = 'varchar(15)';

3. Graded Checkpoint Documents

CSE 3241 Project Checkpoint 01 – Entities and Relationships
Names:
Jiahan Bao, Zhe Huang, Ruiyang Liu, Zhaoyuan Yang , Qisheng Wu
Date 02/09/2016

In a **<u>NEATLY TYPED</u>** document, provide the following:
1. Based on the requirements given in the project overview, list the entities to be modeled in this database. For each entity, provide a list of associated attributes.
Customer: name, tel, shipping address, billing address, account name, password, date of birth, email, credit card, gender, order history
Book: ISBN, Title, Author, Year, purchase price, sell price, category, quantity, minimum quantity

Warehouse: name, address, tel, zip code
Supplier: name, tel, address

2. Based on the requirements given in the project overview, what are the various relationships between entities? (For example, "CUSTOMER entities purchase BOOK entities").
Customer entities purchase book entities.
Book entities are stored in warehouse entities.
Book entities are supplied by supplier entities.
Supplier entities supply warehouse entities.

3. Propose at least two additional entities that it would be useful for this database to model beyond the scope of the project requirements. Provide a list of possible attributes for the additional entities and possible relationships they may have with each other and the rest of the entities in the database. Give a brief, one sentence rationale for why adding these entities would be interesting/useful to the stakeholders for this database project.

Shopping cart and order entities are necessary for online bookstore.
Shopping cart: account name, quantity, total price, is paid
Order: account name, shipping address, status, order number, date, credit card, billing address
Customer create shopping cart so we add shopping cart entities.
Customer place order so we add order entity.

4. Give at least four examples of some informal queries/reports that it might be useful for this database might be used to generate. Include one example for each of the additional entities you proposed in question 3 above.

List all books written by an author.
List all books under minimum quantity.
List all suppliers that have a specific book.
List all books that published in 2015.
List the order history of a customer.
List all books that added into a customer's shopping cart.

5. Suppose we want to add a new publisher to the database. How would we do that given the entities and relationships you've outlined above? Given your above description, is it possible to add a new publisher to your database without knowing the title of any books they have published? If not, revise your model to allow for publishers to be added as separate entities.

We have to first know the book published by this publisher, then add the publisher information into the book entity.

No.
Customer: name, tel, shipping address, billing address, account name, password, date of birth, email, credit card, gender, order history

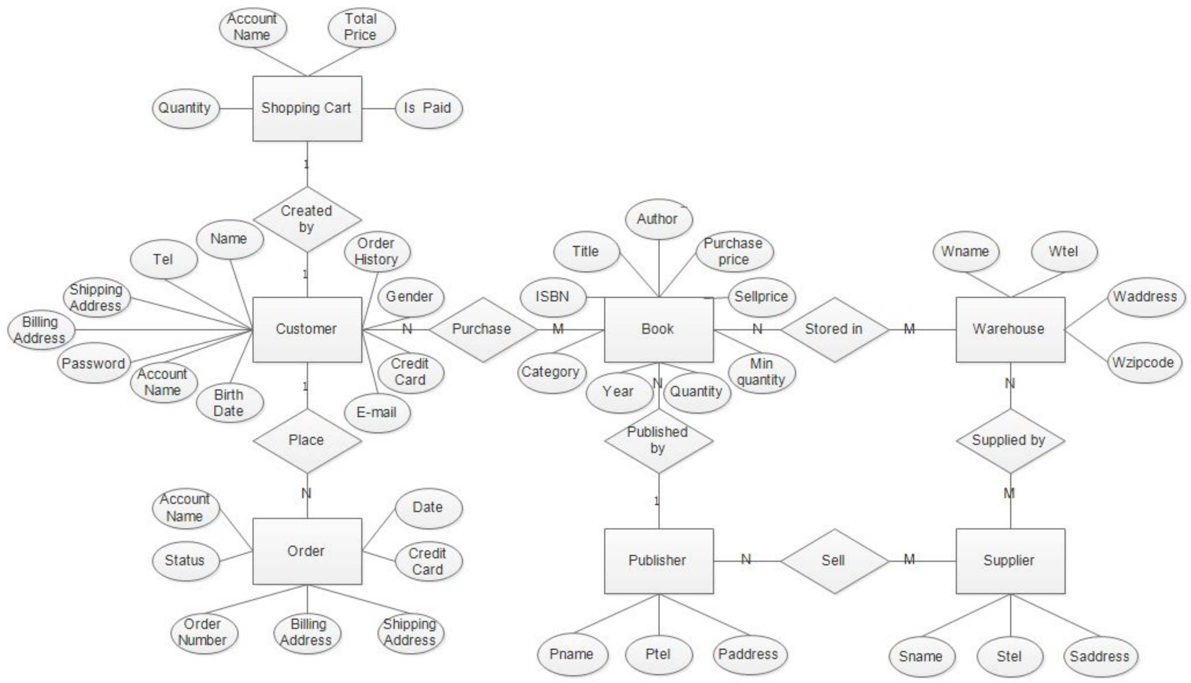
Book: ISBN, Title, Author, Year, purchase price, sell price, Category, quantity, minimum quantity
Publisher: publisher name, tel, address
warehouse: name, address, tel, zipcode
supplier: name, tel, address
shopping cart: account name, quantity, total price, is paid
order: account name, shipping address, status, order number, date, credit card, billing address

6. Determine at least three other informal update operations and describe what entities would need to have attributes altered and how they would need to be changed given your above descriptions. Include one example for each of the additional entities you proposed in question 3 above.

We want to change the storage quantity of a book. So the quantity attribute in the warehouse entity need to be altered.
We want to update the price of a book. So the price attribute in the book entity need to be altered.
A customer wants to change the shipping address. So the shipping address in the customer entity needs to be changed.

A customer adds a new book into the shopping cart. So the quantity and totalprice attributes in the shopping cart entity need to be changed.
A customer wants to cancel the placed order. So the status attribute in the order entity and the quantity attributes in the warehouse entity which are affected need to be changed.

7. Provide an ER diagram for your database. Make sure you include all of the entities and relationships you determined in the questions above ***INCLUDING the entities for question 3 above***, and remember that ***EVERY*** entity in your model needs to connect to another entity in the model via some kind of relationship.



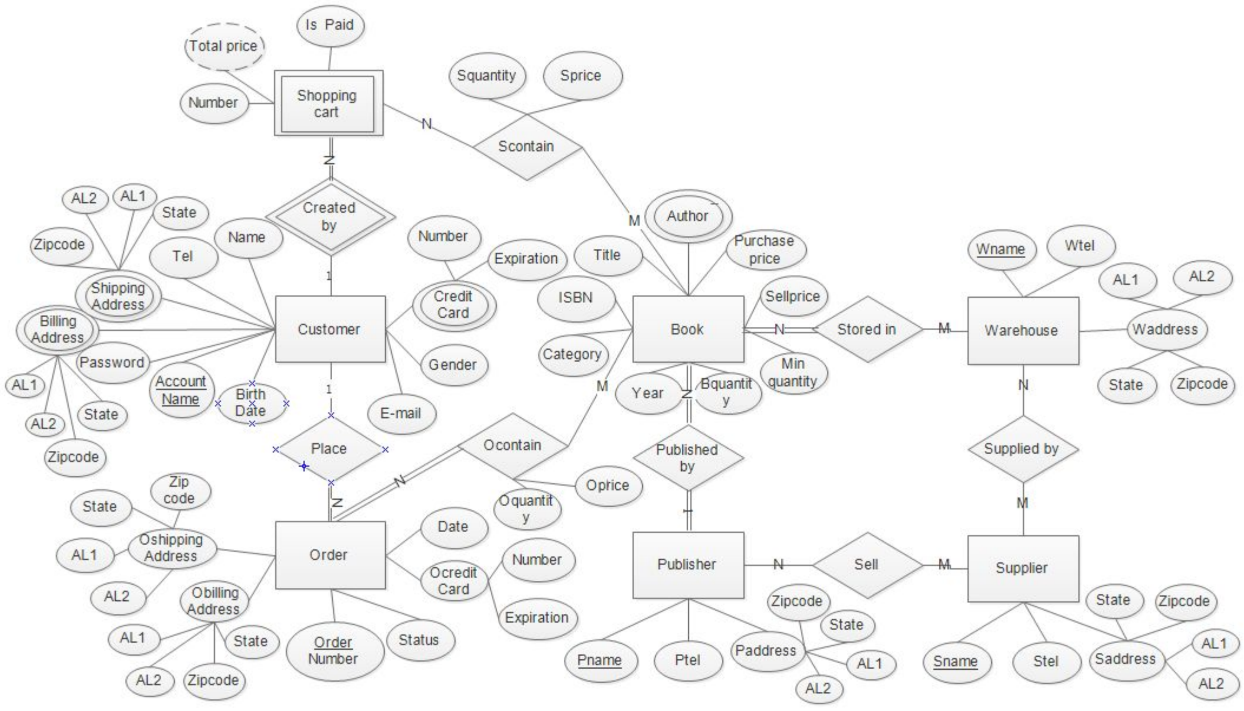
(See Section 1 Page 1 for the new E-R model)

CSE 3241 Project Checkpoint 02 – Relational Model and Relational Algebra

Names Jiahan Bao, Qisheng Wu , Zhaoyuan Yang, Zhe Huang, Ruiyang Liu

Date 02/29/2016

1. Provide a current version of your ER Model as per Project Checkpoint 01. If you were instructed to change the model for Project Checkpoint 01, make sure you use the revised version of your ER Model. (See Section 1 Page 1 for the new E-R model)



2. Map your ER model to a relational schema. Indicate all primary and foreign keys. (See Section 1 Page 1 for the new relational schema)

Customer

AccountName	Name	Tel	Password	BirthDate	E-mail	Gender

ShippingAddress

AccountName	zip code	address line 1	address line 2	state

BillingAddress

AccountName	zip code	address line 1	address line 2	state

CreditCard

AccountName	Number	Expiration

ShoppingCart

AccountName	Number	Is Paid

Order

OrderNumber	AccountNumber	Shipping Zip code	Shipping state	Shipping address line 1	shipping address line 2	billing zip code	billing state	billing address line 1	billing address line 2	Status	Date	Credit Card Number	Credit Card Expiration

Scontain:

Account Name	Number	ISBN	Price	Quantity

Ocontain:

OrderNumber	ISBN	Price	OQuantity

Book:

ISBN	Pname	Title	Purchase price	Sellprice	Minimum quantity	BQuantity	Year	Category

Warehouse:

Wname	Wtel	zip code	address line 1	address line 2	state

Publisher:

Pname	Ptel	zip code	address line 1	address line 2	state

Suppliers:

Sname	Stel	zip code	address line 1	address line 2	state

Store in:

ISBN	Wname

Supplied by:

Wname	Sname

Sell:

Pname	Sname

Author:

ISBN	Author

3. Given your relational schema, provide the relational algebra to perform the following queries. If your schema cannot provide answers to these queries, revise your ER Model and your relational schema to contain the appropriate information for these queries:

a. Find the titles of all books by Pratchett that cost less than $10

$\pi_{\text{title}}(\sigma_{\text{Sellprice < 10}}(\text{BOOK} \bowtie_{\text{ISBN = ISBN}} (\sigma_{\text{Author = 'Pratchett'}}(\text{AUTHOR}))))$

b. Give all the titles and their dates of purchase made by a single customer (you choose how to designate the customer)

$\pi_{\text{Title, Date}}(((\sigma_{\text{accountName='Obama012'}}(\text{ORDER})) \bowtie_{\text{Ordernumber=Ordernumber}} \text{OCONTAIN}) \bowtie_{\text{ISBN=ISBN}} \text{BOOK})$

c. Find the titles and ISBNs for all books with less than 5 copies in stock

$\pi_{\text{title,ISBN}}(\sigma_{\text{Bquantity<5}}(\text{BOOK}))$

d. Give all the customers who purchased a book by Pratchett and the titles of Pratchett books they purchased

$\pi_{\text{Title,Account}}(\text{ORDER} \bowtie_{\text{Ordernumber=Ordernumber}} (\sigma_{\text{author='Pratchett'}}(\text{BOOK}) \bowtie_{\text{ISBN=ISBN}} \text{OCONTAIN}))$

e. Find the total number of books purchased by a single customer (you choose how to designate the customer)

$\mathcal{F}\ \text{SUM}_{\text{oQuantity}}((\sigma_{\text{accountName='Obama012'}}(\text{ORDER})) \bowtie_{\text{Ordernumber=Ordernumber}} \text{OCONTAIN})$

f. Find the customer who has purchased the most books and the total number of books they have purchased (See Section 2 Page 1 for the new relational algebra)

$\text{accountName}\ \mathcal{F}\ \text{MAX}_{\text{OQuantity}}(\text{accountName}\ \mathcal{F}\ \text{SUM}_{\text{OQuantity}}(\text{ORDER} \bowtie_{\text{Ordernumber=Ordernumber}} \text{OCONTAIN}))$

4. Come up with three additional interesting queries that your database can provide. Give what the queries are supposed to retrieve in plain English and then as relational algebra. Your queries should include joins and at least one should include an aggregate function. At least one of your queries should use "extra" entities you added to your model in Checkpoint

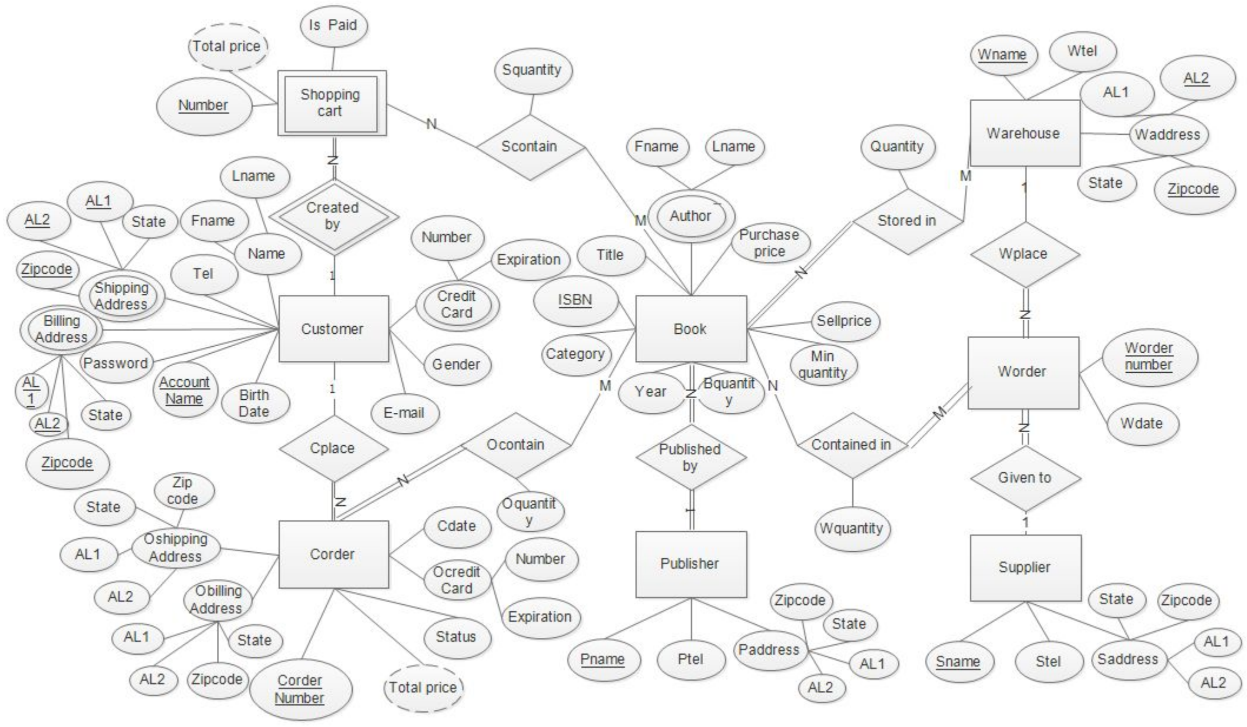
1.List the quantity of comic books purchased by Obama666.

$\mathcal{F}\ \text{SUM}_{\text{OQuantity}}\ (\sigma_{\text{accountName='Obama666'}}(\text{ORDER} \bowtie_{\text{Ordernumber=Ordernumber}}$
$(\sigma_{\text{category='comic'}}(\text{BOOK})\bowtie_{\text{ISBN=ISBN}} \text{OCONTAIN})))$

2. List the title of most popular book purchased by women.
$\text{Title}\mathcal{F}\ \text{MAX}_{\text{oquantity}}\ (\sigma_{\text{Gender='female'}}(\text{Customer}) \bowtie_{\text{Accountname=Accountname}} (\mathcal{F}\ \text{SUM}_{\text{OQuantity}}(\text{ORDER} \bowtie_{\text{Ordernumber=Ordernumber}} \text{BOOK} \bowtie_{\text{ISBN=ISBN}} \text{OCONTAIN})))$
3. List the name of the most popular publisher
$\text{Pname}\mathcal{F}\ \text{MAX}_{\text{oquantity}}(\mathcal{F}\ \text{SUM}_{\text{OQuantity}}(\text{ORDER} \bowtie_{\text{Ordernumber=Ordernumber}} \text{BOOK}\bowtie_{\text{ISBN=ISBN}} \text{OCONTAIN}))$

CSE 3241 Project Checkpoint 03
Names Jiahan Bao, Qisheng Wu , Zhaoyuan Yang, Zhe Huang, Ruiyang Liu

1. ER Model (See Section 1 Page 1 for the new E-R model)



2. Relational schema (See Section 1 Page 1 for the new relational schema)

Customer

Account Name	Fname	Lname	Tel	Password	BirthDate	E-mail	Gender

ShippingAddress

AccountName	zip code	address line 1	address line 2	state

BillingAddress

AccountName	zip code	address line 1	address line 2	state

CreditCard

AccountName	Number	Expiration

ShoppingCart

AccountName	Number	Is Paid

Corder

Order Number	AccountName	Shipping Zip code	Shipping state	Shipping address line 1	shipping address line 2	billing zip code	billing state	billing address line 1	billing address line 2	Status	Cdate	Credit Card Number	Credit Card Expiration

Worder:

Worder number	Wname	Sname	wdate

Contained in:

ISBN	Worder number	WQuantity

Scontain:

Account Name	Number	ISBN	SQuantity

Ocontain:

OrderNumber	ISBN	OQuantity

Book:

ISBN	Pname	Title	Purchase price	Sellprice	Minimum quantity	BQuantity	Year	Category

Warehouse:

Wname	Wtel	zip code	address line 1	address line 2	state

Publisher:

Pname	Ptel	zip code	address line 1	address line 2	state

Suppliers:

<u>Sname</u>	Stel	zip code	address line 1	address line 2	state

Stored in:

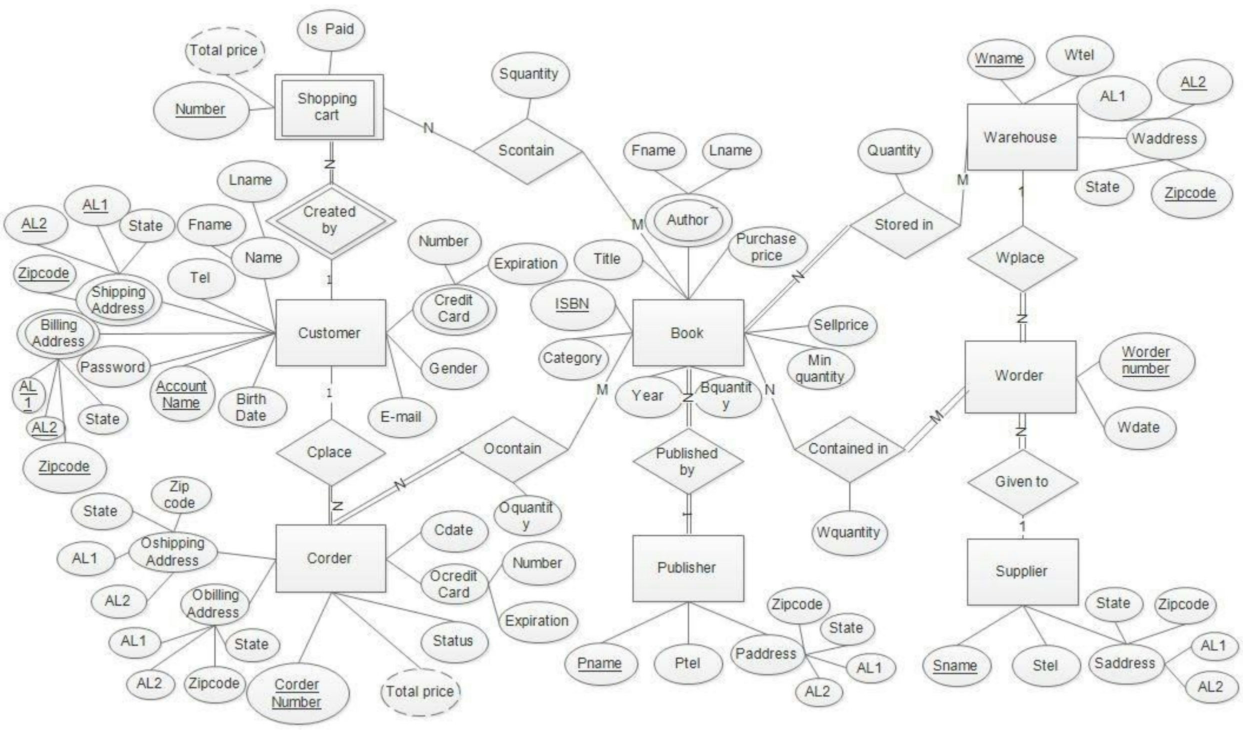
<u>ISBN</u>	quantity	<u>Wname</u>

Author:

<u>ISBN</u>	<u>Lname</u>	<u>Fname</u>

CSE 3241 Project Checkpoint 04
Names Jiahan Bao, Qisheng Wu , Zhaoyuan Yang, Zhe Huang, Ruiyang Liu

In a **<u>NEATLY TYPED</u>** document, provide the following:
1. Provide a current version of your ER Diagram and Relational Model as per Project Checkpoint 03. **If you were instructed to change the model for Project Checkpoint 03, make sure you use the revised versions of your models.**
 (1).   ER Model (See Section 1 Page 1 for the new E-R model)



(2). Relational schema (See Section 1 Page 1 for the new relational schema)

Customer

AccountName	Fname	Lname	Tel	Password	BirthDate	Email	Gender

ShippingAddress

AccountName	zip code	address line 1	address line 2	state

BillingAddress

AccountName	zip code	address line 1	address line 2	state

CreditCard

AccountName	Number	Expiration

ShoppingCart

AccountName	Number	Is Paid

Corder

OrderNumber	AccountName	Shipping Zip code	Shipping state	Shipping address line1	shipping address line2	billingzip code	billingstate	billingaddress line1	billingaddress line2	Status	Cdate	CreditCardNumber	CreditCardExpiration

Worder:

Wordernumber	Wname	Sname	wdate

Contained in:

ISBN	Wordernumber	WQuantity

Scontain:

Account Name	Number	ISBN	SQuantity

Ocontain:

OrderNumber	ISBN	OQuantity

Book:

ISBN	Pname	Title	Purchase price	Sellprice	Minimum quantity	BQuantity	Year	Category

Warehouse:

Wname	Wtel	zipcode	Addressline1	Addressline 2	State

Publisher:

Pname	Ptel	zip code	Addressline1	Addressline2	state

Suppliers:

Sname	Stel	zipcode	Addressline1	Addressline2	state

Stored in:

ISBN	quantity	Wname

Author:

ISBN	Lname	Fname

2. For each relation schema in your model, indicate the functional dependencies. Think carefully about what you are modeling here make sure you consider all the possible dependencies in each relation and not just the ones from your primary keys. For example, a customer's credit card number is unique, and so will uniquely identify a customer even if you have another key in the same table (in fact, if the customer can have multiple credit card numbers, the dependencies can get even more involved). (See Section 1 Page 1 for the new functional dependencies)

Customer

AccountName	Fname	Lname	Tel	Password	BirthDate	Email	Gender

AccountName→Fname
AccountName→Lname
AccountName→Tel
AccountName→Password
AccountName→BirthDate
AccountName→Email
AccountName→Gender
Email→AccountName

ShippingAddress

AccountName	zip code	Addressline1	Addressline2	state

AccountName, Zipcode, Addressline1, Addressline2 → state
Zipcode →state

BillingAddress

AccountName	zipcode	Addressline1	Addressline2	state

AccountName, Zipcode, Addressline1, Addressline2 → state
Zipcode →state

CreditCard

AccountName	Number	Expiration

AccountName,number → expiration
Number→ expiration

ShoppingCart

AccountName	Number	Is Paid

AccountName,Number→Is Paid

Corder

OrderNumber	AccountName	ShippingZipcode	Shipping state	Shippingaddressline1	shippingAddressline2	billingzipcode	billingstate	billingaddressline1	billingaddressline2	Status	Cdate	CreditCardNumber	CreditCardExpiration

OrderNumber→AccountName
OrderNumber→ShippingZipcode
OrderNumber→shippingstate
OrderNumber→shippingAddressline1
OrderNumber→shippingAddressline2
OrderNumber→billingzipcode
OrderNumber→billingstate
OrderNumber→billingAddressline1 OrderNumber→billingAddressline2
OrderNumber→status
OrderNumber→cdate

OrderNumber→CreditCardNumber
OrderNumber→CreditCardExpiration
CreditCardNumber→CreditCardExpiration
CreditCardNumber→billingAddressline1
CreditCardNumber→billingAddressline2
CreditCardNumber→billingstate
CreditCardNumber→billingzipcode Billingzipecode→billingstate
shippingzipecode→shippingstate
shippingAddressline1,shippingAddressline2,shippingstate→shippingzipecode
billingAddressline1,billingAddressline2,billingstate→billingzipecode

Worder:

Worder number	Wna me	Sname	wdate

Worder number→Wname
Worder number→Sname
Worder number→Wdate

Contained in:

ISBN	Worder number	WQuantity

ISBN, Wordernumber→WQuantity

Scontain:

Account Name	Number	ISBN	SQuantity

AccountName,Number→ISBN,SQuantity

Ocontain:

OrderNumber	ISBN	OQuantity

OrderNumber,ISBN→Qquantity

Book:

ISBN	Pname	Title	Purchase price	Sellprice	Minimum quantity	BQuantity	Year	Category

ISBN → Pname
ISBN → Title
ISBN → Purchase price
ISBN → Sellprice
ISBN → Minimum quantity
ISBN → BQuantity
ISBN → Year
ISBN → Category

Warehouse:

Wname	Wtel	zip code	address line 1	address line 2	state

Wname → Wtel
Wname → zip code
Wname → address line 1
Wname → addreess line 2
Zip code → state
Wname → state

Publisher:

<u>Pname</u>	Ptel	zip code	address line 1	address line 2	state

Pname→ Ptel
Pname → zip code
Pname → address line 1
Pname → addreess line 2
Pname → state zipcode→state

Suppliers:

<u>Sname</u>	Stel	zip code	address line 1	address line 2	state

Sname→Stel
Sname→zip code
Sname→address line 1
Sname→address line 2
Sname→state zipcode→state

Stored_in:

<u>ISBN</u>	quantity	<u>Wname</u>

ISBN, Wname→quantity

Author:

<u>ISBN</u>	<u>Lname</u>	<u>Fname</u>

ISBN→**Lame**
ISBN→Fname

3. For each relation schema in your model, determine the highest normal form of the relation. If the relation is not in 3NF, rewrite your relation schema so that it is in at least 3NF. (See Section 1 Page 3 for the new normalization)

Customer

<u>AccountNa m e</u>	Fname	Lname	Tel	Passwo rd	BirthDate	Email	Gender

Third normal form

ShippingAddress

<u>AccountName</u>	<u>zip code</u>	<u>address line 1</u>	<u>address line 2</u>	state

First normal form Change to ShippingAddress

AccountName	zip code	address line 1	address line 2

ShippingState

zip code	state

BillingAddress

AccountName	zip code	address line 1	address line 2	state

First normal form Change to
BillingAddress

AccountName	zip code	address line 1	address line 2

BillingState

zip code	state

CreditCard

AccountName	Number	Expiration

First normal form Change to:
CreditCard

Number	Expiration

CreditCard number

AccountName	Number

ShoppingCart

AccountName	Number	Is Paid

Third normal form

Corder

OrderNumber	AccountName	ShippingZipcode	Shippingstate	ShippingAddressline1	ShippingAddressline2	billingzipcode	billingstate	billingAddressline1	billingAddressline2	Status	Cdate	CreditCardNumber	CreditCardExpiration

First normal form Change to
Corder

Order Numbe	Account Name	Shipping Zip code	Shipping address line1	shipping address line2	Status	Cdate	Credit Card Number

r							

CShippingState

Shipping Zip code	Shipping state

CBillingState

Billing zip code	billing state

CCreditCard

Credit Card Number	Credit Card Expiration

CCreditCard_Billing

billingzipecode	billingAddressline1	billingAddressline2	CreditCardNumber

Worder:

Worder number	Wna me	Sname	wdate

Third normal form

Contained in:

ISBN	Wordernumbe r	WQuantity

Third normal form

Scontain:

Account Name	Numb er	ISBN	SQuantity

Third normal form

Ocontain:

OrderNumber	ISBN	OQuantity

Third Normal Form

Book:

ISBN	Pname	Title	Purchase price	Sellprice	Minimum quantity	BQuantity	Year	Category

Third Normal Form

Warehouse:

Wname	Wtel	zip code	address line 1	address line 2	state

First Normal Form Change to:
Warehouse

Wname	Wtel	zip code	address line 1	address line 2

WarehouseState

zip code	state

Publisher:

Pname	Ptel	zip code	address line 1	address line 2	state

First Normal Form Change to:
Publisher

Pname	Ptel	address line 1	address line 2

PublisherState

zip code	state

Suppliers:

Sname	Stel	zip code	address line 1	address line 2	state

First Normal Form Change to:
Suppliers

Sname	Stel	zip code	address line 1	address line 2

SupplierState

Zip code	state

Stored in:

ISBN	quantity	Wname

Third normal form

Author:

<u>ISBN</u>	<u>Lname</u>	<u>Fname</u>

Third normal form

4. For each relation schema in your model that is in 3NF but not in BCNF, either rewrite the relation schema to BCNF or provide a short justification for why this relation should be an exception to the rule of putting relations into BCNF.
All the relation schemas in the model are BCNF.

5. For your database, propose at least two interesting views that can be built from your relations. These views must involve joining at least two tables together each and must include some kind of aggregation in the view. Each view must also be able to be described by a one or two sentence description in plain English. Provide the code for constructing your views along with the English language description of what the view is supposed to be providing.
1.List the quantity of fiction books purchased by Obama666.

```
CREATE VIEW QUANTITY_OF_OBAMA
AS SELECT SUM(OQUANTITY)
FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO
WHERE CU.AccountName= CO.AccountName AND OC.OrderNumber=CO.OrderNumber AND
OC.ISBN=BO.ISBN AND CU.AccountName='E' AND BO.Category='fiction'；
```

2. List the title of most popular book purchased by women.
Title$\mathcal{F}$ MAX oquantity($\mathcal{F}$ SUM OQuantity(CORDER ⋈ Ordernumber=Ordernumber B
OOK⋈ISBN=ISBN O CONTAIN))

```
CREATE VIEW POPULAR_BOOK
AS SELECT RE.TITLE, MAX (RE.SUM_QU)  AS SUM_FEMALE
FROM (SELECT TITLE, SUM(Oquantity) AS SUM_QU
FROM CUSTOMER AS CU, CORDER AS CO, OCONTAIN AS OC, BOOK AS BO
WHERE CU.ACCOUNTNAME= CO.ACCOUNTNAME AND
OC.ORDERNUMBER=CO.ORDERNUMBER AND OC.ISBN=BO.ISBN AND CU.gender='f'
GROUP BY BO.TITLE) AS RE；
```