CIS3400

THE BOOKSTORE DATABASE PROJECT

Honsam Fan: Honsam.Fan@baruchmail.cuny.edu
Moshe Khalili: Moshe.Khalili@baruchmail.cuny.edu
Audite Talukder: Audite.Talukder@baruchmail.cuny.edu
Jonathan Zhao: Jonathan.Zhao@baruchmail.cuny.edu

Group 13

I. BUSINESS SCENARIO

For the past five years, our business has maintained and managed a small bookstore at a local neighborhood in Park Slope, Brooklyn. In recent times, an increase in productivity and service within the bookstore has deemed recording through the conventional method of a paper logbook as inefficient. To improve efficacy, we want to transition from our traditional process onto a more advance digital platform such as a database management system—to keep track of our customers, payments, books and wholesale distributors' transactions.

Customers can rent or buy their favorite books at our bookstore. As our business expands, we need to oversee information such as the Customer Order (purchase date, order payment), Customer Order Details (quantity, price, sold or rented) and Book details (title, year published, genre, version number, and price). To continue to supply unique and popular books, we also need to monitor details for the wholesale distributing company; in many instances, customers demand for certain books that we must request from the Distributing company. We must also regulate the Distributor Order and Order Details to keep a record of our transactions with them.

Alongside keeping a record of the physical addresses and contact information of customers, we also must manage the new membership plan we introduced with our growing business. Through the membership, the customer is assigned to one of the three Membership Levels: Bronze, Silver or Gold. As active customers accumulate membership points, they are upgraded to a new level and are offered promotional discounts with their purchases.

We believe we can create a more efficient system of aggregating data through a database management system and in hopes, create a more supportive and welcoming experience for our customers.

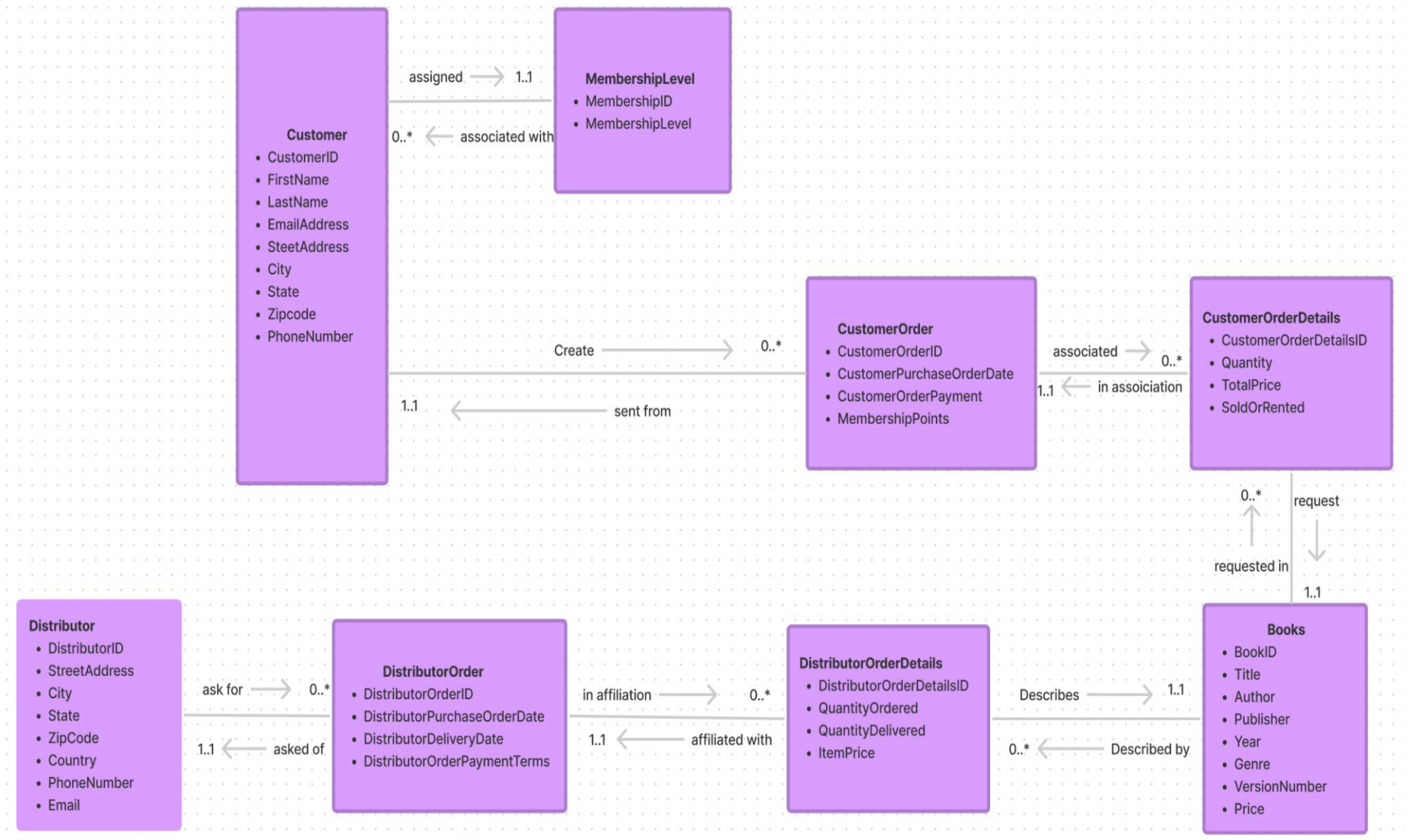
II. ENTITIES IDENTIFIED

a. Customers
b. Membership Level
c. Customer Order
d. Customer Order Details
e. Book
f. Distributor Order
g. Distributor Order Details
h. Distributor

III. ROLES

a. Honsam Fan: Application Developer
b. Moshe Khalili: Systems Analyst
c. Audite Talukder: Documentation Writer
d. Jonathan Zhao: Systems Analyst

IV. E-R DIAGRAM



V. UML Notation to Sentences

a. One Customer *must be* assigned to one and only one Membership Level
b. One Membership Level *may be* associated with one or more Customers
c. One Customer *may* create one or more Customer Orders
d. One Customer Order *must be* sent by one and only one Customer
e. One Customer Order *may be* associated with many Customer Order Details
f. One Customer Order Details *must be* in association with one and only one Customer Order
g. One Customer Order Details *must have* a request of one and only one Books
h. One Books *may be* requested in one or more Customer Order Details
i. One Book *may be* described by many Distributor Order Details
j. One Distributor Order Detail *must* describe one and only one Book
k. One Distributor Order Detail *must be* affiliated with one and only one Distributor Order
l. One Distributor Order *may be* in affiliation with one or more Distributor Order Detail
m. One Distributor Order *must be* asked of by one and only one Distributors
n. One Distributor *may* ask for many Distributor Orders

VI. UML Notation to Relations

a. MembershipLevel [MembershipID (key), MembershipLevel]

b. Customer [CustomerID (key), FirstName, LastName, EmailAddress, StreetAddress, City, State, ZipCode, PhoneNumber, MembershipID (fk)]

c. CustomerOrder [CustomerOrderID (key), CustomerPurchaseOrderDate CustomerOrderPayment, MembershipPoints, CustomerID (fk)]

d. CustomerOrderDetails [CustomerOrderDetailsID (key), Quantity, TotalPrice, SoldOrRented, CustomerOrderID (fk), BooksID (fk)]

e. Books [BookID (key) , Title, Author, Publisher, Year, Genre, VersionNumber, Price]

f. DistributorOrderDetails [DistributorOrderDetailsID (key) , QuantityOrdered, QuantityDelivered, ItemPrice, DistributorOrderID (fk), BookID (fk)]

g. DistributorOrder [DistributorOrderID (key), DistributorPurchaseOrderDate, DistributorDeliveryDate, DistributorOrderPaymentTerms, DistributorID (key)]

h. Distributor [DistributorID (key), StreetAddress, City, State, ZipCode, Country, PhoneNumber, Email]

VII. Normalization

MembershipLevel (MembershipID (key), MembershipLevel)
Key: MembershipID
FD1: MembershipID → MembershipLevel
1NF: Yes, because it meets all the criteria of 1NF
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

Customer (CustomerID (key), FirstName, LastName, EmailAddress, StreetAddress, City, State, ZipCode, PhoneNumber, MembershipID (fk))
Key: CustomerID
FD1: CustomerID → FirstName, LastName, EmailAddress, StreetAddress, City, State, ZipCode, PhoneNumber, MembershipID
FD2: ZipCode → City, State
1NF: Yes, because it meets all the criteria of 1NF
2NF: Yes, there are no partial dependencies
3NF: No, because ZipCode → City, State is a transitive dependency

Because FD2: ZipCode → City, State is the functional dependency that is causing error we must split Customer into two new relations and copy the attribute on the left of the arrow and remove the attribute on the right to the arrow. In this case, we must

FD2: Copy ZipCode → Remove City, State

Customer (CustomerID, FirstName, LastName, EmailAddress, StreetAddress, City, State, ZipCode, PhoneNumber, MembershipID) → CustInfo (CustomerID, FirstName, LastName, EmailAddress, StreetAddress, ZipCode, PhoneNumber, MembershipID) + Zipcodes (ZipCode, City, State)

CustInfo (CustomerID, FirstName, LastName, EmailAddress, StreetAddress, ZipCode, PhoneNumber, MembershipID)
Key: CustomerID
FD1: CustomerID → FirstName, LastName, EmailAddress, StreetAddress, ZipCode, PhoneNumber, MembershipID
1NF: Yes, because it is split off a relation
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

Zipcodes (ZipCode, City, State)
Key: ZipCode
FD1: ZipCode → City, State
1NF: Yes, because it is split off a relation
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

We know will denormalize the Zipcodes (ZipCode, City, State) back to the original relation to avoid redundancy to simplify and improve the model for performance.

CustomerOrder (CustomerOrderID (key), CustomerPurchaseOrderDate, CustomerOrderPayment, MembershipPoints, CustomerID (fk))
Key: CustomerOrderID
FD1: CustomerOrderID → CustomerPurchaseOrderDate, CustomerOrderPayment, MembershipPoints, CustomerID
1NF: Yes, because it meets all the criteria of 1NF
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

CustomerOrderDetails (CustomerOrderDetailsID (key), Quantity, TotalPrice, SoldOrRented, CustomerOrderID (key)(fk), BooksID (fk))
Key: CustomerOrderDetailsID, CustomerOrderID
FD1: CustomerOrderDetailsID, CustomerOrderID → Quantity, TotalPrice, SoldOrRented, BooksID
1NF: Yes, because it meets all the criteria of 1NF
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

Books (BookID (key) , Title, Author, Publisher, Year, Genre, VersionNumber, Price)
Key: BookID
FD1: BookID → Title, Author, Publisher, Year, Genre, VersionNumber, Price
1NF: Yes, because it meets all the criteria of 1NF
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

DistributorOrderDetails (DistributorOrderDetailsID (key), QuantityOrdered, QuantityDelivered, ItemPrice, DistributorOrderID (key)(fk), BookID (fk))
Key: DistributorOrderDetailsID, DistributorOrderID
FD1: DistributorOrderDetailsID, DistributorOrderID → QuantityOrdered, QuantityDelivered, ItemPrice, BookID
1NF: Yes, because it meets all the criteria of 1NF
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

DistributorOrder (DistributorOrderID (key), DistributorPurchaseOrderDate, DistributorDeliveryDate, DistributorOrderPaymentTerms, DistributorID (fk))
Key: DistributorOrderID
FD1: DistributorOrderID → DistributorPurchaseOrderDate, DistributorDeliveryDate, DistributorOrderPaymentTerms, DistributorID
1NF: Yes, because it meets all the criteria of 1NF
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

Distributor (DistributorID (key), StreetAddress, City, State, ZipCode, Country, PhoneNumber, Email)
Key: DistributorID
FD1: DistributorID → StreetAddress, City, State, ZipCode, Country, PhoneNumber, Email
FD2: ZipCode → City, State
1NF: Yes, because it meets all the criteria of 1NF
2NF: Yes, there are no partial dependencies
3NF: No, because ZipCode → City, State is a transitive dependency

We must split Distributor into two new relations and from the functional dependency
FD2: ZipCode → City, State we must copy ZipCode and remove City, State

Distributor (DistributorID (key), StreetAddress, City, State, ZipCode, Country, PhoneNumber, Email) → DistributorInfo (DistributorID (key), StreetAddress, ZipCode, Country, PhoneNumber, Email) + DistributorZipcodes (ZipCode, City, State)

DistributorInfo (DistributorID (key), StreetAddress, ZipCode, Country, PhoneNumber, Email)
Key: DistributorID
FD1: DistributorID → StreetAddress, ZipCode, Country, PhoneNumber, Email
1NF: Yes, because it is split off a relation
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

DistributorZipcodes (ZipCode, City, State)
Key: ZipCode
FD1: ZipCode → City, State
1NF: Yes, because it is split off a relation
2NF: Yes, there are no partial dependencies
3NF: Yes, there are no transitive dependencies

We know will denormalize the DistributorZipcodes (ZipCode, City, State) back to the original relation to avoid redundancy to simplify and improve the model for performance.

Final Relations:

MembershipLevel (MembershipID (key), MembershipLevel)

Customer (CustomerID (key), FirstName, LastName, EmailAddress, StreetAddress, City, State, ZipCode, PhoneNumber, MembershipID (fk))

CustomerOrder (CustomerOrderID (key), CustomerPurchaseOrderDate CustomerOrderPayment, MembershipPoints, CustomerID (fk))

CustomerOrderDetails (CustomerOrderDetailsID (key), Quantity, TotalPrice, SoldOrRented, CustomerOrderID (key)(fk), BooksID (fk))

Books (BookID (key), Title, Author, Publisher, Year, Genre, VersionNumber, Price)

DistributorOrderDetails (DistributorOrderDetailsID (key), QuantityOrdered, QuantityDelivered, ItemPrice, DistributorOrderID (key)(fk), BookID (fk))

DistributorOrder (DistributorOrderID (key), DistributorPurchaseOrderDate, DistributorDeliveryDate, DistributorOrderPaymentTerms, DistributorID (fk))

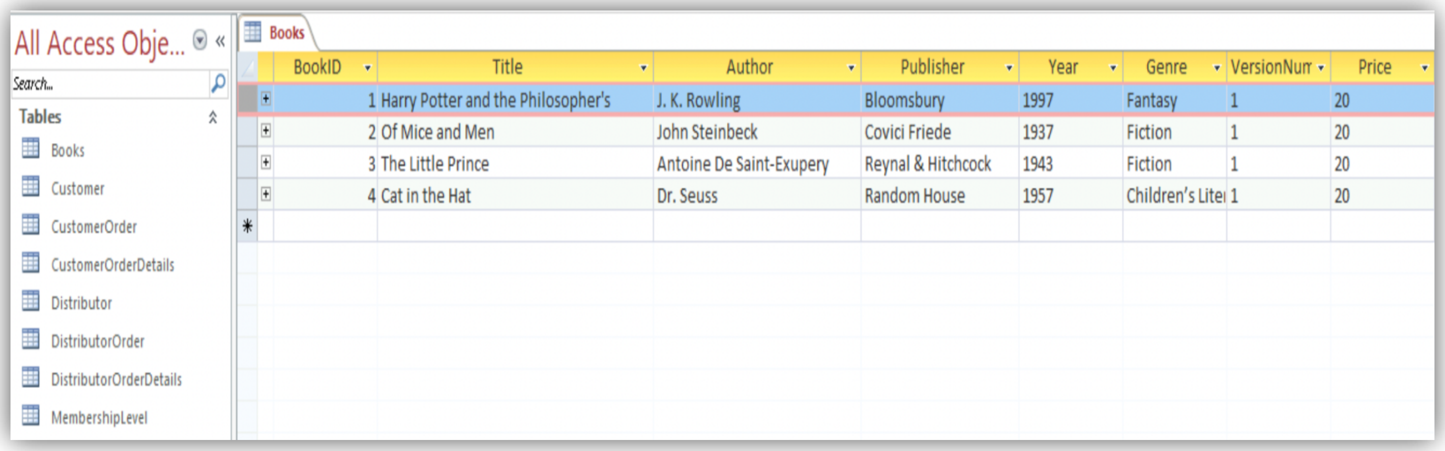
Distributor (DistributorID (key), StreetAddress, City, State, ZipCode, Country, PhoneNumber, Email)

VIII. CREATE TABLE SQL

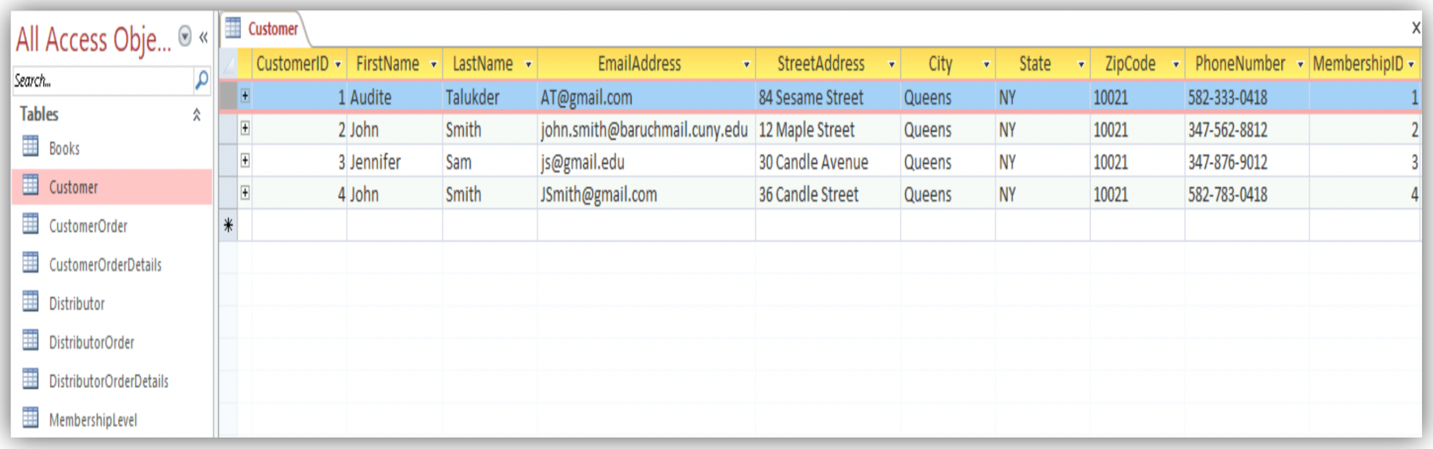
```
CREATE TABLE MembershipLevel
(
        MembershipID NUMBER NOT NULL,
        Membership_Level VARCHAR(20),
        CONSTRAINT pk_membershiplevel
        PRIMARY KEY (MembershipID)
)

CREATE TABLE Customer
(
        CustomerID NUMBER NOT NULL,
        FirstName VARCHAR(35),
        LastName VARCHAR(35),
        EmailAddress VARCHAR(35),
        StreetAddress VARCHAR(35),
        City VARCHAR(35),
        State VARCHAR(4),
        ZipCode VARCHAR(20),
        PhoneNumber VARCHAR(20),
        MembershipID NUMBER,
        CONSTRAINT pk_customer
        PRIMARY KEY (CustomerID)
)

CREATE TABLE CustomerOrder
(
        CustomerOrderID NUMBER NOT NULL,
        CustomerPurchaseOrderDate DATE,
        CustomerOrderPayment VARCHAR(35),
        MembershipPoints VARCHAR(35),
        CustomerID NUMBER,
        CONSTRAINT pk_customerorder
        PRIMARY KEY (CustomerOrderID)
)

CREATE TABLE Books
(
        BookID NUMBER NOT NULL,
        Title VARCHAR(35),
        Author VARCHAR(35),
        Publisher VARCHAR(35),
        Year VARCHAR(20),
        Genre VARCHAR(35),
        VersionNumber VARCHAR(35),
        Price VARCHAR(35),
        CONSTRAINT pk_books
        PRIMARY KEY (BookID)
)
```

```
CREATE TABLE CustomerOrderDetails
(
        CustomerOrderDetailsID NUMBER NOT NULL,
        Quantity VARCHAR(35),
        TotalPrice VARCHAR(35),
        SoldOrRented VARCHAR(35),
        CustomerOrderID NUMBER,
        BookID NUMBER,
        CONSTRAINT pk_customerorderdetails
        PRIMARY KEY (CustomerOrderDetailsID)
)

CREATE TABLE Distributor
(
        DistributorID NUMBER NOT NULL,
        StreetAddress VARCHAR(35),
        City VARCHAR(35),
        State VARCHAR(4),
        ZipCode VARCHAR(20),
        PhoneNumber VARCHAR(20),
        EmailAddress VARCHAR(35),
        CONSTRAINT pk_distributor
        PRIMARY KEY (DistributorID)
)

CREATE TABLE DistributorOrder
(
        DistributorOrderID NUMBER NOT NULL,
        DistributorPurchaseOrderDate DATE,
        DistributorDeliveryDate DATE,
        DistributorOrderPaymentTerms VARCHAR(35),
        DistributorID NUMBER,
        CONSTRAINT pk_distributororder
        PRIMARY KEY (DistributorOrderID)
)

CREATE TABLE DistributorOrderDetails
(
        DistributorOrderDetailsID NUMBER,
        QuantityOrdered VARCHAR(35),
        QuantityDelivered VARCHAR(35),
        ItemPrice VARCHAR(35),
        DistributorOrderID NUMBER,
        BookID NUMBER,
        CONSTRAINT pk_distributororderdetails
        PRIMARY KEY (DistributorOrderDetailsID)
)
```

SAMPLE TABLES



BookID	Title	Author	Publisher	Year	Genre	VersionNum	Price
1	Harry Potter and the Philosopher's	J. K. Rowling	Bloomsbury	1997	Fantasy	1	20
2	Of Mice and Men	John Steinbeck	Covici Friede	1937	Fiction	1	20
3	The Little Prince	Antoine De Saint-Exupery	Reynal & Hitchcock	1943	Fiction	1	20
4	Cat in the Hat	Dr. Seuss	Random House	1957	Children's Liter	1	20

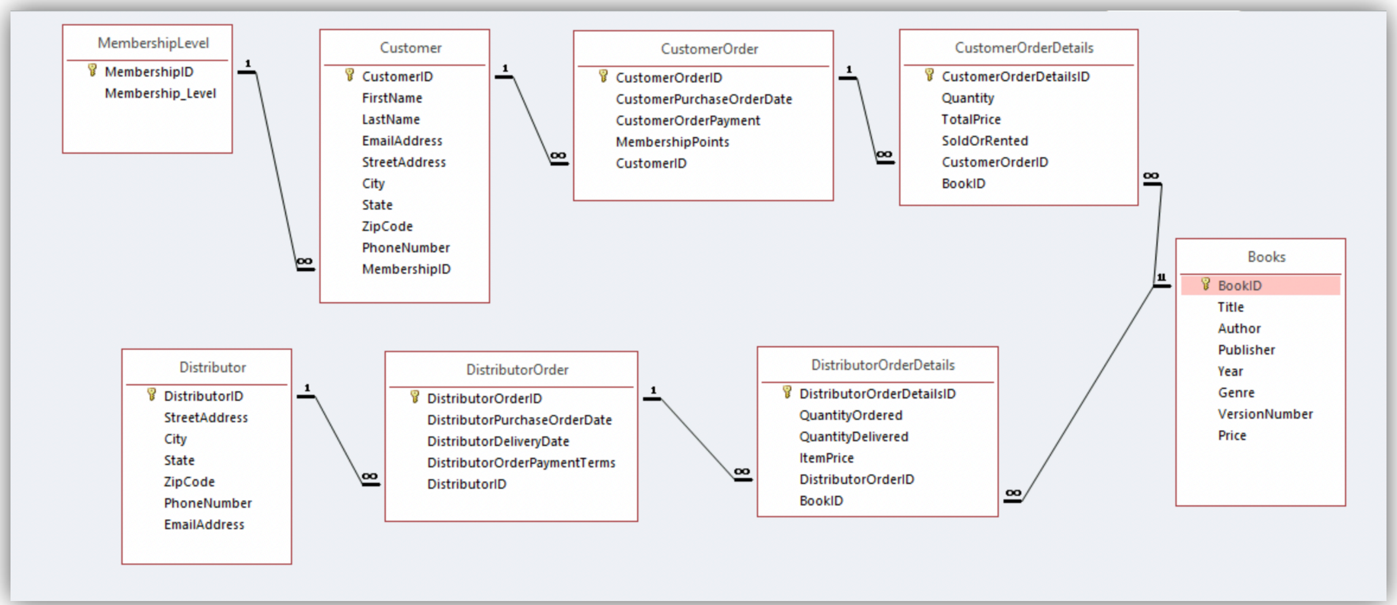


CustomerID	FirstName	LastName	EmailAddress	StreetAddress	City	State	ZipCode	PhoneNumber	MembershipID
1	Audite	Talukder	AT@gmail.com	84 Sesame Street	Queens	NY	10021	582-333-0418	1
2	John	Smith	john.smith@baruchmail.cuny.edu	12 Maple Street	Queens	NY	10021	347-562-8812	2
3	Jennifer	Sam	js@gmail.edu	30 Candle Avenue	Queens	NY	10021	347-876-9012	3
4	John	Smith	JSmith@gmail.com	36 Candle Street	Queens	NY	10021	582-783-0418	4

IX. ALTER TABLE SQL

```
ALTER TABLE Customer
        ADD CONSTRAINT fk_customer_membershiplevel
                FOREIGN KEY (MembershipID)
                        REFERENCES MembershipLevel(MembershipID)

ALTER TABLE CustomerOrder
        ADD CONSTRAINT fk_customerorder_customer
                FOREIGN KEY (CustomerID)
                        REFERENCES Customer(CustomerID)

ALTER TABLE CustomerOrderDetails
        ADD CONSTRAINT fk_customerorderdetails_customerorder
                FOREIGN KEY (CustomerOrderID)
                        REFERENCES CustomerOrder(CustomerOrderID)

ALTER TABLE CustomerOrderDetails
        ADD CONSTRAINT fk_customerorderdetails_books
                FOREIGN KEY (BookID)
                        REFERENCES Books(BookID)

ALTER TABLE DistributorOrder
        ADD CONSTRAINT fk_distributororder_distributor
                FOREIGN KEY (DistributorID)
                        REFERENCES Distributor(DistributorID)

ALTER TABLE DistributorOrderDetails
        ADD CONSTRAINT fk_distributororderdetails_books
                FOREIGN KEY (BookID)
                        REFERENCES Books(BookID)
```

SAMPLE RELATIONSHIP



X. INSERT STATMENTS SQL

```
INSERT INTO MembershipLevel
VALUES (1, "Gold");

INSERT INTO MembershipLevel
VALUES (2, "Silver");

INSERT INTO MembershipLevel
VALUES (3, "Bronze");

INSERT INTO MembershipLevel
VALUES (4, "Gold");

INSERT INTO MembershipLevel
VALUES (5, "Silver");

INSERT INTO Customer
VALUES (001, "Audite", "Talukder", "AT@gmail.com", "84 Sesame Street", "Queens", "NY",
"10021", "582-333-0418", 1);

INSERT INTO Customer
VALUES (002, "John", "Smith", "john.smith@baruchmail.cuny.edu", "12 Maple Street", "Queens",
"NY", "10021", "347-562-8812", 2);

INSERT INTO Customer
VALUES (003, "Jennifer", "Sam", "js@gmail.edu", "30 Candle Avenue", "Queens", "NY", "10021",
"347-876-9012", 3);

INSERT INTO Customer
VALUES (004, "John", "Smith", "JSmith@gmail.com", "36 Candle Street", "Queens", "NY", "1002",
"582-783-0418", 4);

INSERT INTO CustomerOrder
VALUES (1, "12/12/2021", "Cash", "20", 2) ;

INSERT INTO CustomerOrder
VALUES( 2, "11/12/2021", "Cash", "20", 1);

INSERT INTO CustomerOrder
VALUES( 3, "10/12/2021", "Credit", "20", 3);

INSERT INTO CustomerOrder
VALUES( 4, "9/10/2021", "Credit", "20", 4);

INSERT INTO Books VALUES (1, "Harry Potter and the Philosopher's Stone", "J. K. Rowling",
"Bloomsbury", "1997", "Fantasy", "1", "20");

INSERT INTO Books VALUES (2, "Of Mice and Men", "John Steinbeck", "Covici Friede", "1937",
"Fiction", "1", "20");

INSERT INTO Books VALUES (3, "The Little Prince", "Antoine De Saint-Exupery", "Reynal &
Hitchcock", "1943", "Fiction", "1", "20");

INSERT INTO Books VALUES (4, "Cat in the Hat", "Dr. Seuss", "Random House", "1957", "Children’s
Literature", "1", "20");
```

```
INSERT INTO CustomerOrderDetails
VALUES(1, "1", "20", "Sold", 1, 1);

INSERT INTO CustomerOrderDetails
 VALUES(2, "1", "20", "Sold", 2, 1);

INSERT INTO CustomerOrderDetails
 VALUES(3, "1", "20", "Sold", 3, 1);

INSERT INTO CustomerOrderDetails
 VALUES(4, "1", "20", "Sold", 3, 1);

INSERT INTO Distributor
VALUES(1, "24 Bow Street", "Brooklyn", "NY", "10432", "2120982239", "D1@gmail.com");

INSERT INTO Distributor
VALUES(2, "144 SE. Fremont Street", "Flushing", "NY", "11230", "917-858-9920",
"D2@gmail.com");

INSERT INTO Distributor
VALUES(3, "52 Smith Avenue", "Brooklyn", "NY", "10320", "2120982123", "D3@gmail.com");

INSERT INTO Distributor
VALUES(4, "120 James Lane", "Brooklyn", "NY", "11233", "917-858-6898", "D4@gmail.com");

INSERT INTO DistributorOrder
VALUES(1, "11/11/2021", "12/12/2021", COD, 1);

INSERT INTO DistributorOrder
VALUES(2, "10/10/2021", "10/20/2021", COD, 2);

INSERT INTO DistributorOrder
VALUES(3, "10/15/2021", "10/25/2021", COD, 3);

INSERT INTO DistributorOrder
VALUES(4, "11/11/2021", "12/12/2021", COD, 4);

INSERT INTO DistributorOrderDetails
VALUES(1, "40", "40", "20", 1, 1);

INSERT INTO DistributorOrderDetails
VALUES(2, "32", "32", "20", 2, 2);

INSERT INTO DistributorOrderDetails
VALUES(3, "50", "50", "20", 3, 3);

INSERT INTO DistributorOrderDetails
VALUES(4, "30", "30", "20", 4, 4);
```