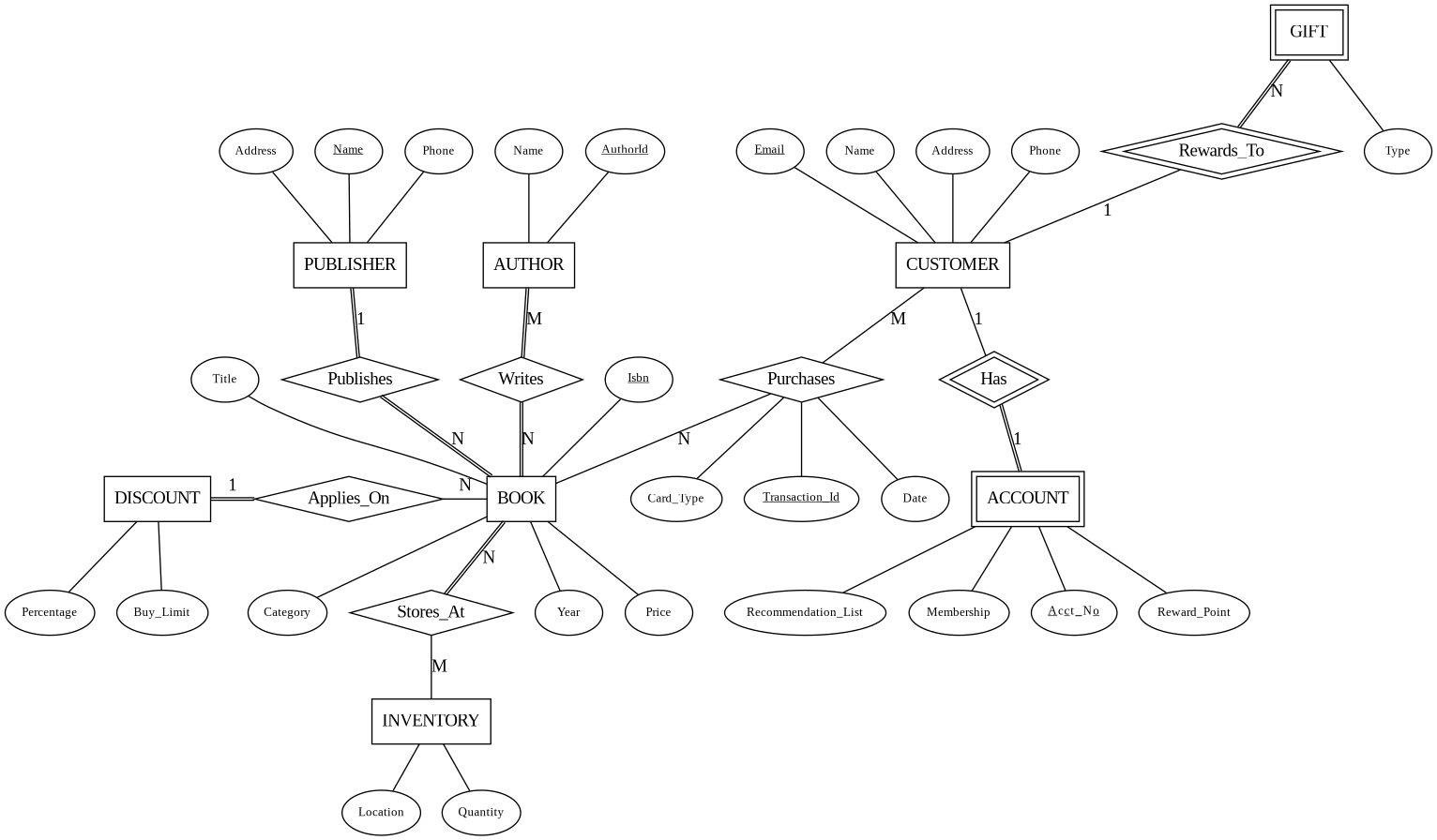
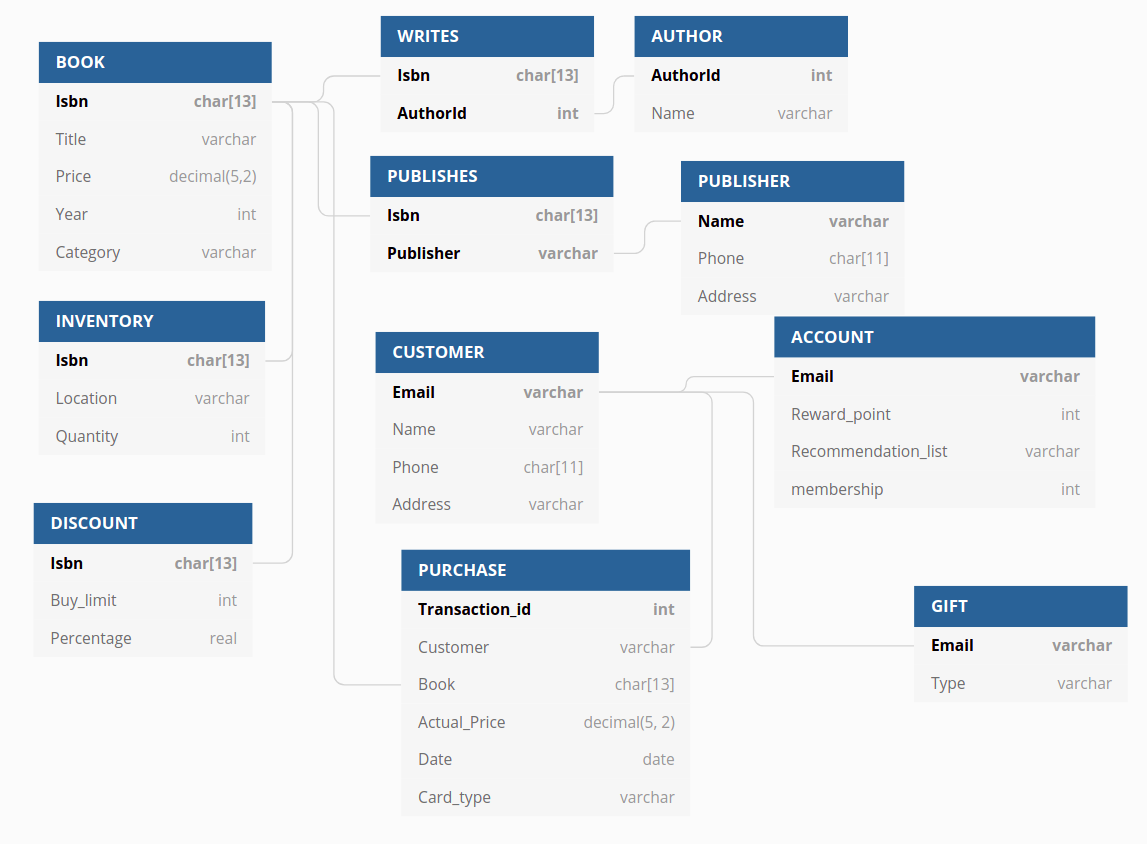
1. **Entity Relationship Diagram**
2. **Relational Database Schema**



1. **Level of Normalization for Each Table**

Book: BCNF

Publisher: BCNF

Author: BCNF

Customer: BCNF

Account: BCNF

Purchases: BCNF

Publishes: BCNF

Writes: BCNF

1. **A description of each of the indexes that you have chosen to implement on your database, along with rationale for each.**

We chose the Tree-based index for our BOOK table since the tree-based index is good for looking up values based on range tests. It will speed up our queries when we want to retrieve the book based on the range of the year or the range of the price. Also it is not too bad for looking up values based on equality tests, so it also can slightly speed up our queries when we are trying to look up books based on the titles, categories, authors, and publishers.

1. **Views**
2. Description:

This view is able to show all the titles and their dates of purchase made by each customer. And this could be useful to make book recommendations for a customer by looking at his or her purchase history.

Relational algebra expression to produce this view:

SQL statements to produce the view:

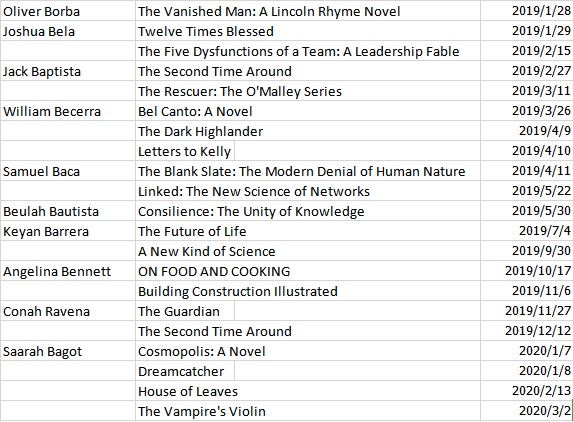
CREATE VIEW CUSTOMER\_P

AS SELECT B.Title, P.Date

FROM BOOK AS B, PURCHASE AS P, C AS CUSTOMER

WHERE B.Isbn = P.Book

5-10 sample outputs:



1. Description:

This view is able to show the total number of books purchased by each customer. And this could be useful to see if this customer deserves a gift by making a certain amount of purchases in this store.

Relational algebra expression to produce this view:

SQL statements to produce the view:

CREATE VIEW CUSTOMER\_N

AS SELECT P.Customer, COUNT(Book)

FROM PURCHASE AS P, CUSTOMER AS C

5-10 sample outputs:



6. **A professionally presented description of three sample transactions useful for your database. This should include the sample SQL code for each transaction as well as an English language description of what “unit of work” the transaction represents. Remember – a transaction is a sequence of SQL statements taken as a unit – this can be reads and writes together or just a sequence of writes. One example of a sample transaction you might want to consider is the user making changes to an order – what might need to be considered a transaction in that case?**

Description:

The customer adds a book to a order and update the book quantity in the inventory

SQL:

UPDATE Purchase

SET Book = Book + \*\*\*, Actual\_Price = Actual\_Price + \*\*\*

WHERE Transaction\_id = \*\*\*

UPDATE Inventory

SET Quantity = Quantity - 1

WHERE Isbn = \*\*\* AND Location = \*\*\*

B.

Description:

A certain amount(10) of books(Isbn: 616601654) transmitted from one inventory(Columbus) to another(Cleveland)

SQL:

UPDATE Inventory

SET Quantity = Quantity + 10

WHERE Isbn = 616601654 AND Location = Cleveland

UPDATE Inventory

SET Quantity = Quantity - 10

WHERE Isbn = 616601654 AND Location = Columbus

C.

Description:

Customer redeem 100 reward points to a keychain

SQL:

UPDATE Account

SET Reward\_point = Reward\_point - 100

WHERE Email = \*\*\*

UPDATE Gift

SET type = “keychain”

WHERE Email = \*\*\*