Section 2 - User Manual

The following document details explains the usage of our database by relating our conceptual database to the statements that can be used to interact with it using DBMS services (our queries were written in SQLite).

a) All of the different aspects of our bookstore are represented with the following entities and attributes:

Book:	A book. All books have a unique ISBN, title, and author's last name who wrote the book. Books can also have additional attributes, such as FirstName, MiddleName, Price, Publisher, Category, Year, Quantity, and QuantitySold. FirstName is the first name of the author, MiddleName is the middle name of the author, Price is the price of the book, Publisher is the publisher of the book, Category is the genre of the book, Year is the year the book was written, Quantity is the amount of books supplied in the bookstore, and QuantitySold is the amount of book sold by the bookstore.
Cart:	An online shopping cart. Each cart needs a title of a book within the cart and an account ID from the customer who has the book in his or her shopping cart.
Contained_In:	Stores all the ISBN and TransID values in the table. This table contains all the ISBN digits from each book made in all transactions by customers.
Customer:	A customer at the bookstore. Each customer has a unique account ID. All customers have an account ID, first name, last name, and email address.
Displayed_In:	Stores all the ISBN and WishlistID values in the table. This table contains all the ISBN digits from each book entered into a wishlist by customers.
Held_In:	Stores all the ISBN and RecommendID values in the table. This table contains all the ISBN digits from each book entered into a recommended list by employees.
Rec._List:	A list containing recommended books for each customer. Books are added to the customers' lists by employees or bots, perhaps based books in customers' wish lists or carts. All recommended lists have a unique ID. Additionally, they need a list ID and an Account ID in order to match a list to a customer.
Trans:	Any transaction made between a customer and the bookstore. Each transaction has a unique ID number. All transactions needs the transaction ID number, a customer's ID, and the total amount. The table also has attributes, such as Date, Downloadable, Shipping, and Paymethod. Date is the date of the transaction. Downloadable distinguishes whether or not a customer downloaded a book. Shipping distinguishes whether or not a customer purchased the book to have it shipped to his or her home. Paymethod distinguishes the type of payment used in a transaction.
Wishlist:	A list containing a customer's desired or notable books. A customer adds books to his or her wishlist. Wish lists have a unique ID, and they have a name.

b) The following SQL queries were made during Checkpoints 02 and 03. Each SQL statement is preceded by the equivalent English statement, and is followed by its equivalent Relational Algebra statement.

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

```
SELECT Title
FROM BOOK
WHERE LastName = "Pratchett" AND Price < 10;
```

$$\pi_{\text{Title}} (\sigma_{\text{LastName = 'Pratchett' AND Price < 10}} (\text{BOOK}))$$

2. Give all the titles and their dates of purchase made by a single customer (the customer named Darius Kharazi):

```
SELECT B.Title, T.Date
FROM BOOK AS B, CONTAINED_IN AS Co, TRANS AS T, CUSTOMER AS Cu
WHERE Co.ISBN = B.ISBN AND Co.TransID = T.TransID AND T.AccountID = Cu.AccountID
AND Cu.LastName = "Kharazi" AND Cu.FirstName = "Darius";
```

$$\text{Temp1} <\!- ((\text{BOOK}) \bowtie_{\text{ISBN = ISBN}} (\text{CONTAINED\_IN})) \bowtie_{\text{TransID = TransID}} (\text{TRANS})$$

$$\text{Temp2} <\!- ((\text{Temp1}) \bowtie_{\text{AccountID = AccountID}} (\text{CUSTOMER}))$$

$$\pi_{\text{Title, Date}} (\sigma_{\text{LastName = 'Kharazi' AND FirstName = 'Darius'}} (\text{Temp2})$$

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

```
SELECT DISTINCT Title, ISBN
FROM BOOK
WHERE Quantity < 5;
```

$$\pi_{\text{Title, ISBN}} (\sigma_{\text{Quantity < 5}} (\text{BOOK}))$$

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

```
SELECT Cu.AccountID, Cu.FirstName, Cu.LastName, B.Title
FROM BOOK AS B, CONTAINED_IN AS Co, TRANS AS T, CUSTOMER AS Cu
WHERE Co.ISBN = B.ISBN AND Co.TransID = T.TransID AND T.AccountID = Cu.AccountID
AND B.LastName = "Pratchett";
```

$$\text{Temp1} \leftarrow ((\text{BOOK}) \bowtie_{\text{ISBN} = \text{ISBN}} (\text{CONTAINED\_IN})) \bowtie_{\text{TransID} = \text{TransID}} (\text{TRANS})$$

$$\text{Temp2} \leftarrow ((\text{Temp1}) \bowtie_{\text{AccountID} = \text{AccountID}} (\text{CUSTOMER}))$$

$$\pi_{\text{AccountID, FirstName, LastName, Title}} (\sigma_{\text{LastName} = \text{'Pratchett'}} (\text{Temp2})$$

5. Find the total number of books purchased by a single customer (the customer named Darius Kharazi):

```
SELECT Cu.AccountID, Cu.FirstName, Cu.LastName, COUNT(T.TransID)
FROM CONTAINED_IN AS Co, TRANS AS T, CUSTOMER AS Cu
WHERE Co.TransID = T.TransID AND T.AccountID = Cu.AccountID AND Cu.AccountID = 3;
```

$$\text{Temp1} \leftarrow (\text{CUSTOMER}) \bowtie_{\text{AccountID=AccountID}} ((\text{CONTAINED\_IN}) \bowtie_{\text{TransID=TransID}} (\text{TRANS}))$$

$${}_{\text{AccountID, FirstName, LastName}} \mathcal{F}_{\text{COUNT(TransID)}} (\sigma_{\text{AccountID} = 3} (\text{Temp1}))$$

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

```
SELECT Cu.AccountID, Cu.FirstName, Cu.LastName, COUNT(T.TransID)
FROM CONTAINED_IN AS Co, TRANS AS T, CUSTOMER AS Cu
WHERE Co.TransID = T.TransID AND T.AccountID = Cu.AccountID
GROUP BY T.TransID
ORDER BY COUNT(T.TransID) DESC, Cu.AccountID ASC, Cu.FirstName ASC, Cu.LastName ASC
LIMIT 1;
```

$$\text{Temp1} \leftarrow (\text{CUSTOMER}) \bowtie_{\text{AccountID=AccountID}} ((\text{CONTAINED\_IN}) \bowtie_{\text{TransID=TransID}} (\text{TRANS}))$$

$${}_{\text{TransID, AccountID, FirstName, LastName}} \mathcal{F}_{\text{COUNT(TransID)}} ((\text{Temp1}))$$

7. Count the number of books written by all authors:

```
SELECT ISBN, FirstName, LastName, COUNT(ISBN)
FROM BOOK
GROUP BY LastName;
```

$$_{\text{ISBN, FirstName, LastName}}\mathcal{F}_{\text{COUNT(ISBN)}}\,(\text{BOOK})$$

8. Find the first and last name of authors who have written books in 2003:

```
SELECT FirstName, LastName
FROM BOOK
WHERE Year = "2003";
```

$$\pi_{\text{FirstName, LastName}}\left(\sigma_{\text{Year = 1995}}\left(\text{BOOK}\right)\right)$$

9. Display the titles of books in customers' wish lists named "TheList.":

```
SELECT B.Title
FROM BOOK AS B, DISPLAYED_IN AS D, WISHLIST AS W
WHERE B.ISBN = D.ISBN AND D.WishlistID = W.WishListID AND WishlistName = "TheList";
```

$$\text{Temp1} \leftarrow ((\text{BOOK}) \bowtie_{\text{ISBN = ISBN}} (\text{DISPLAYED\_IN})) \bowtie_{\text{WishlistID = WishlistID}} (\text{Wishlist})$$

$$\pi_{\text{Title}}\,(\sigma_{\text{WishlistName = 'TheList'}}\,(\text{Temp1})$$

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

```
SELECT Cu.FirstName, Cu.LastName, SUM(T.Amount)
FROM TRANS AS T, CUSTOMER AS Cu
WHERE T.AccountID = Cu.AccountID
GROUP BY Cu.AccountID, Cu.FirstName, Cu.LastName;
```

$$\text{Temp1} \leftarrow (\text{TRANS}) \bowtie_{\text{AccountID = AccountID}} (\text{CUSTOMER})$$

$$_{\text{AccountID, FirstName, LastName}}\mathcal{F}_{\text{SUM(Amount)}}(\text{Temp1})$$

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

```
SELECT Cu.FirstName, Cu.LastName, Cu.Email, SUM(T.Amount)
FROM TRANS AS T, CUSTOMER as Cu
WHERE T.AccountID = Cu.AccountID
GROUP BY Cu.AccountID, Cu.FirstName, Cu.LastName
HAVING SUM(T.Amount) > (SELECT AVG(amt)
                        FROM (SELECT SUM(Tr.Amount) AS amt
                              FROM TRANS AS Tr, CUSTOMER as Cus
                             WHERE Tr.AccountID = Cus.AccountID
                              GROUP BY Cus.AccountID, Cus.FirstName, Cus.LastName));
```

No Translatable Relational Algebra

12. 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:

```
SELECT B.Title, COUNT(B.ISBN)
FROM BOOK AS B, CONTAINED_IN AS C
WHERE B.ISBN = C.ISBN
GROUP BY B.Title
ORDER BY  COUNT(B.ISBN) DESC;
```

$$\text{Temp1} \leftarrow (\text{BOOK}) \bowtie_{\text{ISBN = ISBN}} (\text{CONTAINED\_IN})$$

$$_{\text{Title}}\mathcal{F}_{\text{COUNT(ISBN)}}(\text{Temp1})$$

13. 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:

```
SELECT Title, QuantitySold
FROM BOOK
ORDER BY QuantitySold DESC;
```

$$\pi_{\text{Title, QuantitySold}}(\text{BOOK})$$

14. Find the most popular author in the database (the one who has sold the most books):

```
SELECT FirstName, LastName, SUM(QuantitySold)
FROM BOOK
GROUP BY LastName, FirstName
ORDER BY SUM(QuantitySold) DESC
LIMIT 1;
```

$${}_{\text{LastName, FirstName}}\mathcal{F}_{\text{SUM(QuantitySold)}}(\text{BOOK})$$

15. Find the most profitable author in the database for this store (the one who has brought in the most money):

```
SELECT FirstName, LastName, SUM(QuantitySold*Price)
FROM BOOK
GROUP BY LastName, FirstName
ORDER BY SUM(QuantitySold*Price) DESC
LIMIT 1;
```

$${}_{\text{LastName, FirstName}}\mathcal{F}_{\text{SUM(QuantitySold)}}(\text{BOOK})$$

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

```
SELECT Cu.FirstName, Cu.LastName, Cu.Email
FROM BOOK AS B, CONTAINED_IN AS Co, TRANS AS T, CUSTOMER AS Cu
WHERE B.ISBN = Co.ISBN AND Co.TransID = T.TransID AND T.AccountID = Cu.AccountID AND
((B.LastName) IN (SELECT LastName
                            FROM BOOK
                            GROUP BY LastName, FirstName
                            ORDER BY SUM(QuantitySold*Price) DESC
                            LIMIT 1));
```

No Translatable Relational Algebra

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

```
SELECT B.FirstName, B.LastName
FROM TRANS AS T, CUSTOMER as Cu, CONTAINED_IN AS Co, BOOK AS B
WHERE T.AccountID = Cu.AccountID AND T.TransId = Co.TransID AND Co.ISBN = B.ISBN
GROUP BY Cu.AccountID, Cu.LastName, Cu.FirstName
HAVING SUM(T.Amount) > (SELECT AVG(amt)
                            FROM (SELECT SUM(Tr.Amount) AS amt
                                   FROM TRANS AS Tr, CUSTOMER as Cus
                                  WHERE Tr.AccountID = Cus.AccountID
                                   GROUP BY Cus.AccountID, Cus.FirstName, Cus.LastName));
```

No Translatable Relational Algebra

18. Count the number of books written by each author:

```
SELECT LastName, COUNT(Title)
FROM BOOK
GROUP BY LastName;
```

$${}_{\text{LastName}}\mathcal{F}_{\text{COUNT(Title)}}(\text{BOOK})$$

19. Get the last name of the customers who have books currently being shipped:

```
SELECT C.LastName
FROM CUSTOMER AS C, TRANS AS T
WHERE C.AccountID = T.AccountID AND T.Shipping = TRUE;
```

$$\pi_{\text{LastName}}(\sigma_{\text{Shipping = TRUE}}(\text{TRANS}) \bowtie_{\text{AccountID = AccountID}} (\text{CUSTOMER}))$$

20. Display the titles of books in the customer's wish list with an ID of "396":

```
SELECT B.Title
FROM WISHLIST AS W, DISPLAYED_IN AS D, BOOK AS B
WHERE W.WishlistID = D.WishlistID AND D.ISBN = B.ISBN AND W.WishlistID = 396;
```

$$\text{Temp1} \leftarrow (\text{DISPLAYED\_IN}) \bowtie_{\text{WishtlistID = WishlistID}} (\text{WISHLIST})$$

$$\pi_{\text{Title}}(\sigma_{\text{WishlistID = '396'}}(\text{BOOK}) \bowtie_{\text{ISBN = ISBN}} (\text{Temp1}))$$

c) When using SQL Insert statements, there are some restrictions on what is a valid insertion. To add a new book into the database, the properties ISBN, Title, LastName and Price must be present. Also, the ISBN must be unique, else the insert statement will be invalid, and the book will not be added.

An example of a valid insertion, where all values are specified, is:

```
INSERT INTO BOOK VALUES ("7777777777", "Thomas the Tank Engine", "John", NULL,
"Smith", 104.59, "Sybex", "Children’s", 2002, 3, 4);
```

Insertions can also be valid with only some of the properties specified, as long as the ones mentioned before are still included:

```
INSERT INTO BOOK (ISBN, Title, LastName, Price) VALUES ("7777777777", "Thomas the
Tank Engine", "Smith", 104.59);
```

Here is another example of an Insert statement for the Customer table. The restrictions for Customer are that every attribute must be specified.

```
INSERT INTO CUSTOMER VALUES(132, "Christopher", "Mowery", “cm@gmail.com");
```

Lastly, publishers and authors may be added to the database. However, as stated above, one would need the ISBN, Title (of book), LastName (of author), and Price (of book) in order to add a publisher, who publishes a certain book, and an author, who writes a certain book.

d) Delete statements are also used in order to remove entries from the database. It does not matter which order that delete operations occur, and there are no constraints on Delete statements, other than the specified entry must already exist. Additionally, one should think twice about removing a data entry containing a primary, when another relation contains a foreign key referencing that very primary key. Unless necessary, one should not delete a data entry in BOOK with a unique ISBN, since it is most likely being referenced in another entity, such as CONTAINED_IN. Deleting publishers and authors would be acceptable, since they are not part of the primary key. But, as stated above, this could cause trouble, since one would need to delete a data entry from BOOK containing a unique ISBN. Lastly, unless necessary, deleting a data entry from CUSTOMER would be acceptable, as long as the unique AccountID is not being referenceed by a foreign key from another entity, such as AccountID from TRANS.

Below are some examples of Delete statements.

```
DELETE FROM BOOK WHERE ISBN = "7777777777";

DELETE FROM CUSTOMER WHERE AccountID = 132;
```