User Manual

Table	Entity	Attribute	Data type	Description	Constraints
BOOK	book	Isbn	CHAR(13)	The unique identifier of the book; The primary key	NOT NULL; fixed length of 13
		Title	VARCHAR(100)	Title of the book	NOT NULL; Up to length 100
		Author	VARCHAR(100)	Author of the book	NOT NULL; Up to length 100
		Price	DECIMAL(5,2)	Price of the book	NOT NULL; total 5 digits, two digits after the decimal point
		Year	INT	Year of the book got published	NOT NULL
		Category	VARCHAR(100)	Category of the book	NOT NULL; Up to length 100
INVENTORY	inventory	Isbn	CHAR(13)	The foreign key references to BOOK table; the primary key	NOT NULL; fixed length of 13
		Location	VARCHAR(100)	Location of the inventory	NOT NULL; Up to length 100
		Quantity	INT	the number of books in inventory	
DISCOUNT	discount	Isbn	CHAR(13)	The foreign key references to BOOK table	NOT NULL; fixed length of 13
		Buy_limit	INT	Purchase limit to apply the discount	
		Percentage	REAL	The discount applied to the book	NOT NULL
CUSTOMER	customer	Email	VARCHAR(100)	The unique identifier of the customer; The primary key	NOT NULL; Up to length 100
		Name	VARCHAR(100)	The name of the customer	NOT NULL; Up to length 100
		Phone	CHAR(10)	The phone of the customer	fixed length of 10
		Address	VARCHAR(100)	The address of the customer	NOT NULL; Up to length 100
AUTHOR	author	AuthorId	INT	The unique identifier of the author; The primary key	NOT NULL
		Name	VARCHAR(100)	The name of the author	NOT NULL
PUBLISHER	publisher	Name	VARCHAR(100)	The unique identifier of the publisher; The primary key	NOT NULL; Up to length 100
		Phone	CHAR(10)	The phone of the publisher	fixed length of 10
		Address	VARCHAR(100)	The address of the publisher	NOT NULL; Up to length 100
GIFT	gift	Email	VARCHAR(100)	The foreign key references to CUSTOMER table; The primary key	NOT NULL; Up to length 100
		Type	VARCHAR	The type of the gift	
ACCOUNT	account	Account_no	INT	The unique identifier of the account; The primary key	NOT NULL
		Email	VARCHAR(100)	The foreign key references to CUSTOMER table	NOT NULL; Up to length 100
		Reward_point	INT	The reward point of the account	
		Recommendation_list	VARCHAR(500)	The recommendation list for the customer	Up to length 500
		membership	INT	The level of the membership	
WRITES	author, book	Isbn	CHAR(13)	The foreign key references to BOOK table; One of the primary key	NOT NULL; fixed length of 13
		Authorid	INT	The foreign key references to AUTHOR table; One of the primary key	
PUBLISHES	book, publisher	Isbn	CHAR(13)	The foreign key references to BOOK table	NOT NULL
		Publisher	VARCHAR(100)	The foreign key references to PUBLISHER table	NOT NULL
PURCHASE	customer, book	Transaction_id	INT	The unique identifier of the purchase; The primary key	NOT NULL
		Customer	VARCHAR(100)	The foreign key references to CUSTOMER table	NOT NULL; Up to length 100
		Book	CHAR(13)	The foreign key references to BOOK table	NOT NULL
		Date	DATE	The date of purchase	
		Card_type	VARCHAR(100)	The type of the card that customer use to purchase	Up to length 100
		Actural_Price	DECIMAL(5,2)	The price of the transaction	NOT NULL; total 5 digits, two digits after the decimal point

Queries

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

Return a table that contains the title of that books written by Pratchett and cost less than $10

$$BOOKS \leftarrow \Pi_{Title}(\sigma_{Price<10 \text{ AND } Author='Pratchett'}(BOOK))$$

```
SELECT Title
FROM BOOK AS B
WHERE B.Price < 10 AND B.Author = 'Pratchett';
```

b. Give all the titles and their dates of purchase made by a single customer

Return a table that contains the title and date of the purchases made by a single customer

$$BOOKS \leftarrow BOOK \bowtie_{Isbn=Book} (\sigma_{Customer=Email}(PURCHASE))$$

$$RESULT \leftarrow \pi_{Title,Date}(BOOK)$$

```
SELECT B.Title, P.Date
FROM BOOK AS B, PURCHASE AS P, C AS CUSTOMER
WHERE B.Isbn = P.Book AND P.Customer = C.Email;
```

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

Return a table that contains the title and ISBN for all books with less than 5 copies in inventory

$$STOCK(Isbn,\ Quantity) \leftarrow_{Isbn} \mathcal{F}_{SUM\ Quantity}(INVENTORY)$$

$$RESULT \leftarrow \pi_{Title,Isbn}(\sigma_{Quantity<5}(STOCK))$$

```
SELECT B.Title, B.Isbn
FROM BOOK AS B, INVENTORY AS I
WHERE B.Isbn = I.Isbn
GROUP BY B.Title, B.Isbn
HAVING sum(I.Quantity) < 5;
```

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

Return a table that contains the customers who purchased a book by Pratchett and the titles of Pratchett books they purchased

$$PRATCHETTS \leftarrow (\sigma_{Author=Pratchett}(PUBLISH) * BOOK)$$

$$SALES \leftarrow (PRATCHETTS * PURCHASE)$$

$$RESULT \leftarrow (\pi_{Email,Name,Title}(SALES))$$

```
SELECT C.Email, C.Name, B.Title
FROM CUSTOMER AS C, BOOK AS B, PUBLISHES AS P, PURCHASE AS PUR
WHERE P.Isbn = B.Isbn AND P.Author = 'Pratchett' AND PUR.Book = B.Isbn;
```

e. Find the total number of books purchased by a single customer

Return a table that contains the total number of books a customer has purchased and the email of the customer

$$COUNT(Customer,\ \#\ of\ Books) \leftarrow_{Customer} \mathcal{F}_{COUNT\ BOOK}(PURCHASE)$$

$$RESULT \leftarrow \sigma_{Customer=Email}(COUNT)$$

```
SELECT P.Customer, COUNT(Book)
FROM PURCHASE AS P, CUSTOMER AS C
WHERE P.Customer = C.Email;
```

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

Return a table that contains the customer who has purchased the most books and the amount of the books they have purchased

$$COUNT(Customer,\ No) \leftarrow_{Customer} \mathcal{F}_{COUNT\ BOOK}(PURCHASE)$$

$$RESULT \leftarrow_{Customer} \mathcal{F}_{MAX\ No}(COUNT)$$

```
SELECT L.Customer, L.Num
FROM PURCHASE AS P, CUSTOMER AS C,
(SELECT P.Customer AS Customer, COUNT(Book) AS Num
FROM PURCHASE AS P, CUSTOMER AS C
WHERE P.Customer = C.Email) AS L
WHERE L.Num = MAX(L.Num);
```

g. Find the CUSTOMER with the most Reward_point on his/her account

Return a table that contains the email and name of the customer who has the most reward point on his or her account

$$CACCT \leftarrow CUSTOMER * ACCOUNT$$

$$RESULT \leftarrow \pi_{Email,\ Name}(_{Email,\ Name}\mathcal{F}_{MAX\ Reward\_point}(CACCT))$$

```
SELECT Email, Name
FROM CUSTOMR C, ACCOUNT A
WHERE C.Email = A.Email
GROUP BY C.Email, C.Name
HAVING A.Reward_point = MAX(A.Reward_point)
```

h. Find the most expensive BOOK with all the DISCOUNT applied

Return a table that contains ISBN and title of books that has the most expensive price with applied discount

$$DIS\_BOOKS \leftarrow BOOK \rtimes DISCOUNT$$

$$RESULT \leftarrow_{Isbn,\ Title} \mathcal{F}_{MAX(Price*percentage)}(DIS\_BOOKS)$$

```
SELECT Isbn, Title
FROM (BOOK LEFT OUTER JOIN DISCOUNT)
WHERE Percentage IS NOT NULL
GROUP BY Isbn, Title
HAVING Price*Percentage = MAX(Price*Percentage);
```

i. Find the total price of all the BOOK for each stock (quantity * price)

Return a table that contains the total price of all the books and their ISBN for each stock

$$STOCK \leftarrow BOOK *_{Isbn} \mathcal{F}_{SUM\ Quantity}(INVENTORY)$$

$$RESULT \leftarrow \pi_{Isbn,Quantity\ *\ Price}(STOCK)$$

```
SELECT Isbn, Quantity*Price
FROM BOOK NATURAL JOIN INVENTORY
GROUP BY Isbn
HAVING Quantity*Price=SUM(Quantity);
```

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

Return a table that contains the names of the customers and the total dollar amount each customer has spent

$$Customer\_spend \leftarrow CUSTOMER * (CUSTOMER \Im_{SUM Actural\_Price}(PURCHASE))$$
$$RESULT \leftarrow \Pi_{Name, SUM(Actual\_Price)} Customer\_spend$$

```
SELECT C.Name, SUM(P.Actual_Price)
FROM PURCHASE AS P, CUSTOMER AS C
WHERE P.Customer = C.Email
GROUP BY P.Customer
```

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

Return a table that contains the names and e-mail addresses of the customers who have spent more than the average customer

$$\rho_{Total(Email, Total\_amount)}(CUSTOMER \Im_{SUM Actual\_Price}(PURCHASE))$$
$$CUSINFO \leftarrow Total * CUSTOMER;$$
$$RESULT \leftarrow \Pi_{Name, Email}(\sigma_{Total\_amount > AVG Total\_amount}(CUSINFO))$$

```
SELECT Name, Email
FROM CUSTOMER NATURAL JOIN (SELECT SUM Actural_Price AS Total_amount
                                    FROM PURCHASE AS P
                                    GROUP BY P.Customer ) AS TOTAL
HAVING TOTAL.Toal_amount > AVG (TOTAL.Total_amount)
```

l. 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.

Return a table that contains the total number of each book that has been sold to customers, and the titles of those books, sorted by the number of books in descending number.

$$SOLD \leftarrow PURCHASE \bowtie_{Book=Isbn} BOOK$$
$$COUNT \leftarrow Isbn \Im_{COUNT Book} SOLD$$
$$RESULT \leftarrow \Pi_{Title, COUNT Book} COUNT$$

```
SELECT B.Title, COUNT(P.Book)
 FROM PURCHASE AS P, BOOK AS B
 WHERE P.Book = B.Isbn
 GROUP BY P.Book
 ORDER BY COUNT(P.Book) DESC
```

m. 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.

Return a table that contains the total dollar amount of each book that has been sold to customers, and the titles of those books, sorted by the dollar amount in descending order.

$$SOLD \leftarrow PURCHASE \bowtie_{Book=Isbn} BOOK$$
$$\rho_{Total\_book(Isbn,count)} Isbn\Im_{COUNT(Book)}(SOLD)$$
$$RESULT \leftarrow \Pi_{Title,Count*Actual\_Price(Total\_book)}$$

```
SELECT B.Title, P.Actual_Price * COUNT(P.Book) AS Total_dollars
 FROM PURCHASE AS P, BOOK AS B
 WHERE P.Book = B.Isbn
 GROUP BY P.Book
 ORDER BY P.Actual_Price * COUNT(P.Book) DESC
```

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

Return a table that contains the name of the author who sold the most books

$$SOLD \leftarrow (PURCHASE \bowtie_{Book=Isbn} BOOK) \bowtie_{Book=Isbn} WRITES$$
$$\rho_{R1(Total\_amount)}(AuthorId\Im_{COUNTBook}(SOLD))$$
$$RESULT \leftarrow Name\Im_{MAX(Total\_amount)}(R1)$$

```
SELECT A.Name, MAX(Total)
FROM AUTHOR AS A, (SELECT A.AuthorId, COUNT(P.Book) AS Total
                   FROM PURCHASE AS P, BOOK AS B, WRITES AS W
                   WHERE P.Book = B.Isbn AND B.Isbn = W.Isbn
                   GROUP BY W.AuthorId) AS R1
 WHERE R1.AuthorId = A.AuthorId
 GROUP BY A.AuthorId, A.Name
```

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

Return a table that contains the name of the author who brought in the most money and the dollar amount

$$SOLD \leftarrow (PURCHASE \bowtie_{Book=Isbn} BOOK) \bowtie_{Book=Isbn} WRITES$$
$$\rho_{R1(Total\_amount)}(AuthorId\Im_{COUNTBook}(SOLD))$$
$$Total\_dollar \leftarrow \Pi_{Name,Actual\_Price*Total\_amount} R1$$
$$MAX \leftarrow (\Im_{MAX(Actual\_Price*Total\_amount)} Total\_Dollar) * Total\_dollar$$
$$RESULT \leftarrow \Pi_{Name,Actual\_Price*Total\_amount} MAX$$

```
SELECT A.Name, MAX(Total)
FROM AUTHOR AS A, (SELECT A.AuthorId, P.Actual_Price * COUNT(P.Book) AS Total
                   FROM PURCHASE AS P, BOOK AS B, WRITES AS W
                   WHERE P.Book = B.Isbn AND B.Isbn = W.Isbn
                   GROUP BY W.AuthorId) AS R1
 WHERE R1.AuthorId = A.AuthorId
 GROUP BY A.AuthorId, A.Name
```

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

Return a table that contains the information of customers who purchased anything written by the most profitable author

$$SOLD \leftarrow (PURCHASE \bowtie_{Book=Isbn} BOOK) \bowtie_{Book=Isbn} WRITES$$
$$\rho_{R1(Total\_amount)}(AuthorId\Im_{COUNTBook}(SOLD))$$
$$Total\_dollar \leftarrow \Pi_{Name,AuthorId,Actual\_Price*Total\_amount}R1$$
$$Max\_author \leftarrow (Name, AuthorId\Im_{MAX(Actual\_Price*Total\_amount)}Total\_Dollar)$$
$$CUS \leftarrow CUSTOMER \bowtie_{Email=Customer} PURCHASE$$
$$Cust\_author \leftarrow (CUS \bowtie_{Book=Isbn} WRITES) * Max\_author$$
$$RESULT \leftarrow \Pi_{Name,Email,Phone,Address(Cust\_author)}$$

```
SELECT C.Name, C.Email, C.Phone, C.Address
FROM (SELECT A.AuthorId, MAX(Total)
        FROM (SELECT A.AuthorId, P.Actual_Price * COUNT(P.Book) AS Total
                FROM PURCHASE AS P, BOOK AS B, WRITES AS W
                WHERE P.Book = B.Isbn AND B.Isbn = W.Isbn
                GROUP BY W.AuthorId) AS R1,
              AUTHOR AS A
        WHERE R1.AuthorId = A.AuthorId
        GROUP BY A.AuthorId) AS R2, CUSTOMER AS C
WHERE P.Book = B.Isbn AND B.Isbn = W.Isbn AND W.AuthorId = R2.AuthorId
```

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

Return a table that contains the name of the authors whose books are purchased by the customers who have spent more than average customer

$$Author\_sold \leftarrow (WRITES * AUTHOR) \bowtie_{Isbn=Book} PURCHASE$$
$$\rho_{Spent(Email,total\_spent)}(customer\Im_{SUMActual\_Price}PURCHASE)$$
$$CUS \leftarrow \sigma_{total\_spent>AUGtotal\_spent}Spent$$
$$RESULT \leftarrow Author\_sold \bowtie_{customer=Email} CUS$$

```
SELECT A.Name
FROM AUTHOR AS A, WRITES AS W, PURCHASE AS P, CUSTOMER AS C,
     (SELECT C1.Email AS Email, sum(P1.Actual_Price) AS Spent
      FROM PURCHASE P1, CUSTOMER C1
      WHERE P1.Customer = C1.Email
      GROUP BY P1.Customer) AS personal_sum
WHERE A.AuthorId = W.AuthorId AND W.Isbn = B.Isbn AND C.Email = personal_sum.Email AND
personal_sum.Spent > avg(personal_sum.Spent) AND P.Customer = C.Email AND P.Book = B.Isbn
```

Insert syntax for adding books, publisher, authors and customers

a. Add books

/*Since each book must have publisher and author, and each author, publisher must write/publish a book the book insert will followed by writes and publishes relation insert, then followed by author and publisher insert, vise versa*/

```
INSERT INTO BOOK(ISBN, Title, Year, Price, Category)
VALUES ('0782140661','OCP:Oracle9iCertificationKit',2002,104.97,'Computer');

INSERT INTO WRITES(ISBN,AuthorId)
VALUES ('0782140661',12345)

INSERT INTO AUTHOR(AuthorId,name)
VALUES (12345, 'ChipDawes');

INSERT INTO PUBLISHES(ISBN,Publisher)
VALUES ('0782140661', 'Sybex');

INSERT INTO PUBLISHER(Name,Phone,Address)
VALUES ('Sybex','1234567890','this is an address');
```

b. Add publisher

```
INSERT INTO PUBLISHER(Name,Phone,Address)
VALUES ('Sybex','1234567890','this is an address');

INSERT INTO PUBLISHES(ISBN,Publisher)
VALUES ('0782140661', 'Sybex');

INSERT INTO BOOK(ISBN, Title, Year, Price, Category)
VALUES ('0782140661','OCP:Oracle9iCertificationKit',2002,104.97,'Computer');

INSERT INTO WRITES(ISBN,AuthorId)
VALUES ('0782140661',12345)

INSERT INTO AUTHOR(AuthorId,name)
VALUES (12345, 'ChipDawes');
```

c. Add authors

```
INSERT INTO AUTHOR(AuthorId,name)
VALUES (12345, 'ChipDawes');

INSERT INTO WRITES(ISBN,AuthorId)
VALUES ('0782140661',12345)

INSERT INTO BOOK(ISBN, Title, Year, Price, Category)
VALUES ('0782140661','OCP:Oracle9iCertificationKit',2002,104.97,'Computer');

INSERT INTO PUBLISHES(ISBN,Publisher)
VALUES ('0782140661', 'Sybex');
```

```
INSERT INTO PUBLISHER(Name,Phone,Address)
VALUES ('Sybex','1234567890','this is an address');
```

d. Add customer

```
/*A customer can have 0 or 1 account, and make 0 or more purchases, so no dependencies*/

INSERT INTO CUSTOMER(Email,Name,Phone,Address)
VALUES ('email@email.com','A Name','0123456789', 'This is an address');
```

Delete syntax for removing books, publisher, authors and customers

a. Remove books

```
/*When delete a book, delete its publisher and writer, if no other book written/published by same ones*/
DELETE FROM BOOK
WHERE ISBN = '0782140661';

DELETE FROM AUTHOR
WHERE (SELECT ISBN
        FROM (AUTHOR LEFT OUTER JOIN WIRTES)
      WHERE ISBN = '0782140661' )

DELETE FROM WRITES
WHERE ISBN = '0782140661';

DELETE FROM PUBLISHER
WHERE (SELECT ISBN
        FROM (PUBLISHER LEFT OUTER JOIN PUBLISHES)
      WHERE ISBN = '0782140661' )

DELETE FROM PUBLISHES
WHERE ISBN = '0782140661';
```

b. Remove publisher

```
/*When deleting publisher, delete all books publishes by them, and the books' author*/

DELETE FROM PUBLISHER
WHERE Name = 'Sybex';

DELETE FROM AUTHOR
WHERE (SELECT Authorid
        FROM WRITES
        WHERE (SELECT ISBN
                FROM PUBLISHES
                WHERE Publisher = 'Sybex'));

DELETE FROM WRITES
WHERE (SELECT ISBN
        FROM PUBLISHES
        WHERE Publisher = 'Sybex');

DELETE FORM BOOK
WHERE (SELECT ISBN
```

```
        FROM (PUBLISHERES RIGHT OUTER JOIN BOOK)
        WHERE PUBLISHER.Publisher = 'Sybex');

DELETE FROM PUBLISHES
WHERE Publisher = 'Sybex';
```

c. Remove authors

```
/*When deleting author, delete all books written by them(if they are the only author), and the books' publisher*/

DELETE FROM AUTHOR
WHERE AuthoreId = '12345';

DELETE FROM PUBLISHER
WHERE (SELECT Publisher
        FROM PUBLISHES
        WHERE (SELECT ISBN
                FROM WRITES
                WHERE AuthorId='12345'));

DELETE FROM PUBLISHES
WHERE (SELECT ISBN
        FROM WRITES
        WHERE AuthorId='12345');

DELETE FORM BOOK
WHERE (SELECT ISBN
     FROM (WRITES RIGHT OUTER JOIN BOOK)
     WHERE AuthorId='12345');

DELETE FROM WRITES
WHERE AuthorId='12345';
```

d. Remove customer

```
/*Remove the account also, if the customer has one*/

DELETE FROM ACCOUNT
WHERE Email = 'email@email.com'

DELETE FROM CUSTOMER
WHERE Email = 'email@email.com'
```