Queries

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

SQL code:

```
SELECT C.Fname, C.Lname, C.customer_ID, C.street1, C.street2, C.city, C.country,
C.postal
FROM AUTHOR A, BOOK B, ORDER O, CUSTOMER C, BUYS Y
WHERE C.customer_ID = Y.customer_ID AND Y.book_ID = B.book_ID AND B.book_ID =
A.book_ID AND (A.Fname, A.Lname) IN
        (SELECT K.Fname, K.Lname
        FROM AUTHOR K
        WHERE A.book_ID = B.book_ID and B.book_ID = O.book_ID
        GROUP BY O.book_ID
        HAVING max(sum(count(O.book_ID) * B.price)));
```

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

SQL code:

```
SELECT A.Fname, A.Lname
FROM AUTHOR A, BOOK B, CUSTOMER C, ORDER O, BUYS Y
WHERE C.customer_ID = Y.customer_ID AND Y.book_ID = B.book_ID AND B.book_ID =
A.book_ID AND C.customer_ID IN
        (SELECT C.customer_ID
        WHERE C.customer_ID = O.customer_ID
        HAVING SUM(O.total) > AVG(SUM(O.total));
```

3. 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.

SQL code:

```
SELECT B.title, sum(O.book_ID)*B.Price
FROM BOOK B, CUSTOMER C, ORDER O
WHERE C.customer_ID = O.customer_ID AND B.book_ID = O.book_ID
GROUP BY B.title
ORDER BY sum(O.book_ID)*B.Price DESC;
```

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

SQL code:

```
SELECT A.Fname, A.Lname
FROM AUTHOR A, BOOK B, ORDER O
WHERE A.book_ID = B.book_ID and B.book_ID = O.book_ID
GROUP BY O.book_ID
HAVING max(count(O.book_ID);
```

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

SQL code:

```
SELECT A.Fname, A.Lname
FROM AUTHOR A, BOOK B, ORDER O
WHERE A.book_ID = B.book_ID and B.book_ID = O.book_ID
GROUP BY O.book_ID
HAVING max(sum(count(O.book_ID) * B.price));
```

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

Relational algebra expression:

$\text{Pratchett\_BOOK\_ORDER } (\sigma_{\text{Lname = 'Pratchett'}} (\text{AUTHOR})) \bowtie_{\text{bookID=bookID}} \text{ORDER}$
$\text{Pratchett\_BOOK } (\sigma_{\text{Lname='Pratchett'}} \text{AUTHOR}) \bowtie_{\text{authorID=authorID}} \text{BOOK}$
$\text{CUSTOMER\_Pratchett\_BOOK Pratchett\_BOOK\_ORDER} \bowtie_{\text{customerID=customerID}}$
CUSTOMER
$\text{RESULT } \pi_{\text{Fname,Lname,title}} \text{ CUSTOMER\_Pratchett\_BOOK}$

SQL code:

```
SELECT C.fname, C.lname, B.title
FROM Customer C, Book B, Orders O, Author A, Buys K
WHERE C.customer_ID=K.customer_ID AND O.order_ID=K.order_ID AND
B.author_ID=A.author_ID AND K.book_ID = B.book_ID AND A.lname='Pratchett'
ORDER BY C.fname;
```

7. Find the total number of books purchased by A B.

Relational algebra expression:

$\text{SINGLE\_CUSTOMER } \sigma_{\text{customerID = customerID}} (\text{CUSTOMER})$
$\text{CUSTOMER\_ORDERS SINGLE\_CUSTOMER} \bowtie_{\text{customerID = customerID}} (\text{ORDER})$
$\text{ORDER\_DETAILS CUSTOMER\_ORDERS} \bowtie_{\text{bookID = bookID}} (\text{BOOK})$
$\text{BMAX } \mathcal{F}_{\text{COUNT bookID}} (\text{ORDER\_DETAILS}))$
$\text{RESULT } \pi_{\text{ total\_book}} \ (\sigma_{\text{ Fname = 'John' AND Lname = 'Smith'}} (\text{BMAX}))$

SQL code:

```
SELECT C.Fname, C.Lname, count(A.book_ID)
FROM CUSTOMER C, BUYS A
WHERE C.Fname = 'A' AND C.lname = 'B' AND A.customer_ID = C.customer_ID;
```

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

Relational algebra expression:

$C\_BOOKS\ \pi_{customerID,\ total\_book}\ (CUSTOMER)$

$RESULT\ customerID\ \mathcal{F}_{MAX\ total\_book}\ (C\_BOOKS)$

SQL code:

```
SELECT C.Fname, C.lname, max(y.num)
FROM Customer C, Buys A, (SELECT count(A.book_ID) as num
FROM BUYS A, Customer C
WHERE A.customer_ID = C.customer_ID
GROUP BY C.customer_ID) y
WHERE C.customer_ID = A.customer_ID;
```

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

Relational algebra expression:

$AUTHOR\_PRATCHET\ \sigma_{Lname\ =\ 'Pratchett'}\ (AUTHOR)$

$BOOKS\_BY\_PRATCHETT\ AUTHOR\_PRATCHETT\ \bowtie_{authorID\ =\ authorID}\ BOOK$

$BOOKS\_UNDER\_10\ \sigma_{price\ <\ 10}\ (BOOKS\_BY\_PRATCHETT)$

$RESULT\ \pi_{title}\ (BOOKS\_UNDER\_10)$

SQL code:

```
SELECT B.title
FROM Book B, Author A
WHERE B.author_ID=A.author_ID AND A.Lname='Pratchett' AND B.Price<10
GROUP BY B.title;
```

10. Give all the titles and their dates of purchase made by A B.

Relational algebra expression:

$CUSTOMER\_ORDERS\ CUSTOMER\ \bowtie_{customerID\ =\ customerID}\ ORDER$

$ORDER\_DETAILS\ CUSTOMER\_ORDERS\ \bowtie_{bookID\ =\ bookID}\ BOOK$

$RESULT\ \pi_{title,\ date}\ (\sigma_{Fname\ =\ 'John'\ AND\ Lname\ =\ 'Smith'}\ (ORDER\_DETAILS))$

SQL code:

```
SELECT B.title, O.date
FROM Customer C, Orders O, Book B, Buys A
WHERE C.customer_ID = A.customer_ID AND A.order_ID = O.order_ID AND B.book_ID =
A.book_ID AND C.fname = 'A'
GROUP BY B.title;
```

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

Relational algebra expression:

STOCK WAREHOUSE $\bowtie_{\text{bookID = bookID}}$ BOOK

LOW_STOCK $\sigma_{\text{amount < 5}}$ STOCK

RESULT $\pi_{\text{title, ISBN}}$ (LOW_STOCK)

SQL code:

```
SELECT B.ISBN, B.title
FROM Book B, WAREHOUSE W
WHERE B.book_ID = W.book_ID
GROUP BY 1,2
HAVING count(*)<5;
```

12. Find the employee with the highest salary (aggregate MAX, and extra entity).

Relational algebra expression:

SALARIES $\pi_{\text{salary,employeeID}}$ (EMPLOYEE)

RESULT $\pi_{\text{employeeID}}$ (employeeID $\mathcal{F}_{\text{MAX salary}}$ (SALARIES))

SQL code:

```
SELECT E.employeeID, MAX(E.salary)
FROM Employee E;
```

13. Find all books published by Pearson.

Relational algebra expression:

BOOK_DETAILS PUBLISHER * BOOK

RESULT $\sigma_{\text{publisher\_name = 'pearson'}}$ (BOOK_DETAILS)

SQL code:

```
FROM PUBLISHER P, BOOK B
WHERE P.publisher_name = 'Pearson'
AND P.book_ID = B.book_ID;
```

14. Find how many books are in each warehouse.

Relational algebra expression:

BOOKS_IN_WAREHOUSE BOOK * WAREHOUSE

RESULT π_{amount} (BOOKS_IN_WAREHOUSE)

SQL code:

```
SELECT W.warehouse_ID, count(B.book_ID)
FROM WAREHOUSE W, BOOK B
WHERE W.book_ID = B.book_ID
GROUP BY W.warehouse_ID;
```