Chapter Seven Questions:

1. Write a query to display the different years in which books have been published. Include each year only once and sort the results by year (Figure P7.60). (P60)

```
SELECT DISTINCT Book_Year FROM BOOK ORDER BY Book_Year;
```

```
MariaDB [Ch07_Fact]> SELECT DISTINCT Book_Year FROM BOOK ORDER BY Book_Year;
+-----------+
| Book_Year |
+-----------+
|      2014 |
|      2015 |
|      2016 |
|      2017 |
+-----------+
4 rows in set (0.00 sec)
```

2. Write a query to display the different subjects on which FACT has books. Include each subject only once and sort the results by subject (Figure P7.61). (P61)

```
SELECT DISTINCT Book_Subject FROM BOOK ORDER BY Book_Subject;
```

```
MariaDB [Ch07_Fact]> SELECT DISTINCT Book_Subject FROM BOOK ORDER BY Book_Subject;
+--------------+
| Book_Subject |
+--------------+
| Cloud        |
| Database     |
| Middleware   |
| Programming  |
+--------------+
4 rows in set (0.01 sec)
```

3. Write a query to display the book number, title, and cost for all books that cost $59.95 sorted by book number (Figure P7.65). (P65)

```
SELECT Book_Num, Book_Title, Book_Cost FROM BOOK
WHERE Book_Cost = 59.95
ORDER BY Book_Num;
```

```
MariaDB [Ch07_Fact]> SELECT Book_Num, Book_Title, Book_Cost FROM BOOK
    -> WHERE Book_Cost = 59.95
    -> ORDER BY Book_Num;
+----------+-------------------------------+-----------+
| Book_Num | Book_Title                    | Book_Cost |
+----------+-------------------------------+-----------+
|     5235 | Beginner's Guide to JAVA      |     59.95 |
|     5238 | Conceptual Programming        |     59.95 |
|     5242 | C# in Middleware Deployment   |     59.95 |
|     5251 | Thoughts on Revitalizing Ruby |     59.95 |
+----------+-------------------------------+-----------+
4 rows in set (0.00 sec)
```

4. Write a query to display the patron ID, first and last name of all patrons who are students, sorted by patron ID (Figure P7.72). (44 rows) (P72)

```
SELECT PAT_ID, PAT_FNAME, PAT_LNAME FROM PATRON
```

```
WHERE UPPER(PAT_TYPE) = "STUDENT"
ORDER BY PAT_ID;
```

```
MariaDB [Ch07_Fact]> SELECT PAT_ID, PAT_FNAME, PAT_LNAME FROM PATRON
    -> WHERE UPPER(PAT_TYPE) = "STUDENT"
    -> ORDER BY PAT_ID;
+--------+-----------+------------+
| PAT_ID | PAT_FNAME | PAT_LNAME  |
+--------+-----------+------------+
|   1166 | Vera      | Alvarado   |
|   1171 | Peggy     | Marsh      |
|   1172 | Tony      | Miles      |
|   1174 | Betsy     | Malone     |
|   1180 | Nadine    | Blair      |
|   1181 | Allen     | Horne      |
|   1182 | Jamal     | Melendez   |
|   1184 | Jimmie    | Love       |
|   1185 | Sandra    | Yang       |
|   1200 | Lorenzo   | Torres     |
|   1201 | Shelby    | Noble      |
|   1202 | Holly     | Anthony    |
|   1203 | Tyler     | Pope       |
|   1204 | Thomas    | Duran      |
|   1205 | Claire    | Gomez      |
|   1207 | Iva       | Ramos      |
|   1208 | Ollie     | Cantrell   |
|   1209 | Rena      | Mathis     |
```

5. Write a query to display the patron ID, first and last name, and patron type for all patrons whose last name begins with the letter "C," sorted by patron ID (Figure P7.73). (P73)

```
SELECT * FROM PATRON
WHERE LEFT(PAT_LNAME,1) = 'C'
ORDER BY PAT_ID;
```

```
MariaDB [Ch07_Fact]> SELECT * FROM PATRON
    -> WHERE LEFT(PAT_LNAME,1) = 'C'
    -> ORDER BY PAT_ID;
+--------+-----------+-----------+----------+
| PAT_ID | PAT_FNAME | PAT_LNAME | PAT_TYPE |
+--------+-----------+-----------+----------+
|   1160 | robert    | carter    | Faculty  |
|   1208 | Ollie     | Cantrell  | Student  |
|   1210 | Keith     | Cooley    | STUdent  |
+--------+-----------+-----------+----------+
3 rows in set (0.00 sec)
```

6. Write a query to display the author ID, first and last name of all authors whose year of birth is unknown. Sort the results by author ID (Figure P7.74). (P74)

```
SELECT AU_ID, AU_FNAME, AU_LNAME FROM AUTHOR
WHERE AU_BIRTHYEAR IS NULL
ORDER BY AU_ID;
```

```
MariaDB [Ch07_Fact]> SELECT AU_ID, AU_FNAME, AU_LNAME FROM AUTHOR
    -> WHERE AU_BIRTHYEAR IS NULL
    -> ORDER BY AU_ID;
+-------+----------+-----------+
| AU_ID | AU_FNAME | AU_LNAME  |
+-------+----------+-----------+
|   229 | Carmine  | Salvadore |
|   262 | Xia      | Chiang    |
|   559 | Rachel   | McGill    |
+-------+----------+-----------+
3 rows in set (0.00 sec)
```

7. Write a query to display the number of books in the FACT system (Figure P7.78). (P78)

```
SELECT COUNT(*) AS 'Number of Books' FROM BOOK;
```

```
MariaDB [Ch07_Fact]> SELECT COUNT(*) AS 'Number of Books' FROM BOOK;
+-----------------+
| Number of Books |
+-----------------+
|              20 |
+-----------------+
1 row in set (0.01 sec)
```

8. Write a query to display the number of different book subjects in the FACT system (Figure P7.79). (P79)

```
SELECT COUNT(DISTINCT Book_Subject) AS 'Number of Subjects' FROM BOOK;
```

```
MariaDB [Ch07_Fact]> SELECT COUNT(DISTINCT Book_Subject) AS 'Number of Subjects' FROM BOOK;
+-------------------+
| Number of Subjects |
+-------------------+
|                 4 |
+-------------------+
1 row in set (0.01 sec)
```

9. Write a query to display the number of books that are available (not currently checked out) (Figure P7.80). (P80)

```
SELECT COUNT(DISTINCT A.BOOK_NUM)
FROM BOOK AS A
LEFT JOIN CHECKOUT AS B ON A.BOOK_NUM = B.BOOK_NUM
WHERE CHECK_IN_DATE IS NOT NULL OR CHECK_NUM IS NULL;
```

```
MariaDB [Ch07_Fact]> SELECT COUNT(DISTINCT A.BOOK_NUM)
    -> FROM BOOK AS A
    -> LEFT JOIN CHECKOUT AS B ON A.BOOK_NUM = B.BOOK_NUM
    -> WHERE CHECK_IN_DATE IS NOT NULL OR CHECK_NUM IS NULL;
+----------------------------+
| COUNT(DISTINCT A.BOOK_NUM) |
+----------------------------+
|                         20 |
+----------------------------+
1 row in set (0.00 sec)
```

10. Write a query to display the highest book cost in the system (Figure P7.81). (P81)

```
SELECT MAX(BOOK_COST) FROM BOOK;
```

```
MariaDB [Ch07_Fact]> SELECT MAX(BOOK_COST) FROM BOOK;
+----------------+
| MAX(BOOK_COST) |
+----------------+
|         129.95 |
+----------------+
1 row in set (0.00 sec)
```

11. Write a query to display the lowest book cost in the system (Figure P7.82). (P82)

```
SELECT MIN(BOOK_COST) FROM BOOK;
```

```
MariaDB [Ch07_Fact]> SELECT MIN(BOOK_COST) FROM BOOK;
+----------------+
| MIN(BOOK_COST) |
+----------------+
|          49.95 |
+----------------+
1 row in set (0.00 sec)
```

12. Write a query to display the number of different patrons who have ever checked out a book (Figure P7.83). (P83)

```
SELECT COUNT(DISTINCT(A.PAT_ID)) AS 'DIFFERENT PATRONS'
FROM PATRON AS A
JOIN CHECKOUT AS B ON A.PAT_ID=B.PAT_ID
JOIN BOOK AS C ON B.BOOK_NUM=C.BOOK_NUM;
```

```
MariaDB [Ch07_Fact]> SELECT COUNT(DISTINCT(A.PAT_ID)) AS 'DIFFERENT PATRONS'
    -> FROM PATRON AS A
    -> JOIN CHECKOUT AS B ON A.PAT_ID=B.PAT_ID
    -> JOIN BOOK AS C ON B.BOOK_NUM=C.BOOK_NUM;
+-------------------+
| DIFFERENT PATRONS |
+-------------------+
|                33 |
+-------------------+
1 row in set (0.00 sec)
```

13. Write a query to display the author ID and the number of books written by that author. Sort the results in descending order by number of books, then in ascending order by author ID (Figure P7.85). (P85)

```
SELECT A.AU_ID, COUNT(B.BOOK_NUM) AS 'Books Written'
FROM AUTHOR AS A
JOIN WRITES AS B ON A.AU_ID = B.AU_ID
JOIN BOOK AS C ON B.BOOK_NUM=C.BOOK_NUM
GROUP BY AU_ID
ORDER BY COUNT(B.BOOK_NUM) DESC, A.AU_ID;
```

```
MariaDB [Ch07_Fact]> SELECT A.AU_ID, COUNT(B.BOOK_NUM) AS 'Books Written'
    -> FROM AUTHOR AS A
    -> JOIN WRITES AS B ON A.AU_ID = B.AU_ID
    -> JOIN BOOK AS C ON B.BOOK_NUM=C.BOOK_NUM
    -> GROUP BY AU_ID
    -> ORDER BY COUNT(B.BOOK_NUM) DESC, A.AU_ID;
+-------+---------------+
| AU_ID | Books Written |
+-------+---------------+
|   262 |             3 |
|   460 |             3 |
|   185 |             2 |
|   229 |             2 |
|   251 |             2 |
|   383 |             2 |
|   394 |             2 |
|   559 |             2 |
|   218 |             1 |
|   273 |             1 |
|   284 |             1 |
|   438 |             1 |
|   581 |             1 |
|   592 |             1 |
|   603 |             1 |
+-------+---------------+
15 rows in set (0.00 sec)
```

14. Write a query to display the patron ID, first and last name of all patrons who have never checked out any book. Sort the result by patron last name and then first name (Figure P7.101). (P101)

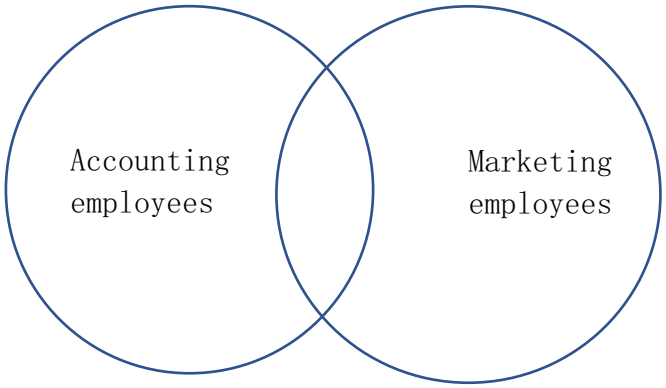
```
SELECT A.PAT_ID, A.PAT_FNAME, PAT_LNAME
FROM PATRON AS A
LEFT JOIN CHECKOUT AS B ON A.PAT_ID = B.PAT_ID
```

```
WHERE CHECK_NUM IS NULL
ORDER BY A.PAT_LNAME, A.PAT_FNAME;
```

```
MariaDB [Ch07_Fact]> SELECT A.PAT_ID, A.PAT_FNAME, PAT_LNAME
    -> FROM PATRON AS A
    -> LEFT JOIN CHECKOUT AS B ON A.PAT_ID = B.PAT_ID
    -> WHERE CHECK_NUM IS NULL
    -> ORDER BY A.PAT_LNAME, A.PAT_FNAME;
+--------+-----------+-----------+
| PAT_ID | PAT_FNAME | PAT_LNAME |
+--------+-----------+-----------+
|   1166 | Vera      | Alvarado  |
|   1180 | Nadine    | Blair     |
|   1238 | Erika     | Bowen     |
|   1208 | Ollie     | Cantrell  |
|   1227 | Alicia    | Dickson   |
|   1205 | Claire    | Gomez     |
|   1239 | Elton     | Irwin     |
|   1240 | Jan       | Joyce     |
|   1243 | Roberto   | Kennedy   |
|   1242 | Mario     | King      |
|   1237 | Brandi    | Larson    |
|   1167 | Alan      | Martin    |
|   1182 | Jamal     | Melendez  |
|   1201 | Shelby    | Noble     |
|   1244 | Leon      | Richmond  |
|   1200 | Lorenzo   | Torres    |
|   1241 | Irene     | West      |
+--------+-----------+-----------+
17 rows in set (0.00 sec)
```

Rick’ s Questions:

1. Using the Ch07_LargeCo ER Diagram, draw a Venn diagram (as seen in the Relational Sets lecture) of employees who have worked for the “Accounting” department and who have worked for the “Marketing” department.



2. Write a word statement describing the overlapping area in the diagram.

   The overlapping area represents employees that worked in both accounting department and marketing department.

3. Write an SQL query that will return only the names of those employees who appear in the overlapping region of the diagram. Note that you will not be able to test it in MySQL.

```
(SELECT A.EMP_FNAME, A.EMP_LNAME
FROM LGEMPLOYEE AS A
JOIN LGDEPARTMENT AS B ON A.DEPT_NUM = B.DEPT_NUM
WHERE DEPT_NAME = 'ACCOUNTING')
INTERSECT
(SELECT C.EMP_FNAME, C.EMP_LNAME
FROM LGEMPLOYEE AS C
JOIN LGDEPARTMENT AS D ON C.DEPT_NUM = D.DEPT_NUM
WHERE DEPT_NAME = 'MARKETING');
```

4. Write an SQL query that will return the names of all employees who have purchased items (are Customers too). Note that you will not be able to test it in MySQL.

```
SELECT B.EMP_FNAME, B.EMP_LNAME
FROM LGCUSTOMER AS A
JOIN LGEMPLOYEE AS B ON A.CUST_FNAME = B.EMP_FNAME AND A.CUST_LNAME =
B.EMP_LNAME;
```