Assignment 9 - Subqueries (50 points)
Due Date: Monday, April 3rd , 2017, 11:59 PM

Objectives

The goal of the assignment is to get you to think about subqueries. This assignment focuses on how to write a subquery, use the IN operator when writing a subquery, use comparison operators when writing a subquery, write a correlated subquery including the use of the EXISTS operator, and use the ORDER BY clause when writing a subquery.

This assignment reinforces the following objectives:

This assignment uses the tables associated with the *bookstore* database.

- Learn the formal subquery definition and write a subquery.
- Learn the subquery restrictions.
- Use the IN operator when writing a subquery.
- Nest subqueries at multiple levels.
- Use comparison operators when writing a subquery.
- Use the ALL and ANY keywords when writing a subquery.
- Write a correlated subquery including the use of the EXISTS operator.
- Use the ORDER BY clause when writing a subquery.

Write SQL statements to perform the following queries:

Query 1: Returns the same result set as this SELECT statement, but don't use a join. Instead, use a subquery in a WHERE clause that uses the IN keyword.

```
SELECT DISTINCT category_name
FROM category c JOIN product p
  ON c.category_id = p.category_id
ORDER BY category_name;
```

Query 2: Answers this question: Which products have a list price that's greater than the average list price for all products?

Return the product_name and list_price columns for each product. Sort the results by the list_price column in descending sequence. Use a subquery.

```
Sample Run:
+------------------------+------------+
| product_name           | list_price |
+------------------------+------------+
| HP Envy 750qe Desktop  |    2517.00 |
| Dell Inspiron 17 5000  |    1415.00 |
| HP Pavilion 15t Star   |    1299.00 |
```

Query 3: Display the category_name column from the CATEGORY table.
Return one row for each category that has assigned to at least one product in the PRODUCT table. To do that, use a subquery with the EXISTS operator.

Query 4: Display the three columns: email_address, order_id, and the order total for each customer. To do this, you can group the result set by the email_address and order_id columns.

In addition, you must calculate the order total from the columns in the ORDERITEMS table. Use a subquery.

Sample Run:

```
+---------------------------+----------+-------------+
| email_address             | order_id | order_total |
+---------------------------+----------+-------------+
| allan.sherwood@yahoo.com  |        1 |      924.30 |
| allan.sherwood@yahoo.com  |        3 |     3306.29 |
| barryz@gmail.com          |        2 |     1671.69 |
```

Query 5: Use a subquery to return one row per customer, representing the customer's oldest order (the one with the earliest date). Each row should include these three columns: email_address, order_id, and order_date.

Sample Run:

```
+---------------------------+----------+---------------------+
| email_address             | order_id | order_date          |
+---------------------------+----------+---------------------+
| allan.sherwood@yahoo.com  |        1 | 2015-03-28 09:40:28 |
| barryz@gmail.com          |        2 | 2015-03-28 11:23:20 |
```

Query 6: For each product, whose items were sold in more than one sales transaction, display the product ID, product name, and list price. Use a subquery.

Sample Run:

```
+------------+---------------------+------------+
| product_id | product_name        | list_price |
+------------+---------------------+------------+
|       1234 | Dell XPS 8700       |     999.00 |
|       2234 | Lenovo H50 Desktop  |    1199.00 |
```

Query 7: Display the last name and first name of customers who have purchased an item that costs more than $300 (item_price). Use a subquery.

Sample Run:

```
+-----------+------------+
| Last_Name | First_Name |
+-----------+------------+
| Sherwood  | Allan      |
| Zimmer    | Barry      |
```

Query 8: Display the last name, first name, and email address of the customers who made the purchase with order IDs 1, 2, and 3. Use a subquery.

Query 9: Display the last name, first name, and email address of customers who have purchased an item that was supplied by a vendor with a company name that begins with the letter H. Use a subquery.

Sample Run:

```
+-----------+------------+---------------------------+
| Last_Name | First_Name | email_address             |
+-----------+------------+---------------------------+
| Zimmer    | Barry      | barryz@gmail.com          |
| Brown     | Christine  | christineb@solarone.com   |
| Goldstein | David      | david.goldstein@hotmail.com |
| Wilson    | Frank Lee  | frankwilson@sbcglobal.net |
```

Query 10: For each product that has more than two items sold within all sales transactions, retrieve the product id, product name, and product price. Use a subquery.

```
Sample Run:
+------------+----------------------+------------+
| product_id | product_name         | list_price |
+------------+----------------------+------------+
|       1234 | Dell XPS 8700        |     999.00 |
```

Submission:

- For each of the queries above, submit the query and the result from running the query. Please use the provided SQL file to write your submissions.
- You will need to label your assignment with your first initial, last name, and the name of the assignment. **Example:** hibrahim_assignment9.sql and hibrahim_assignment9.txt
- Zip the files to upload to Canvas. **Example:** hibrahim_assignment9.zip
- Submit the zipped file containing the script and output TXT via Canvas.
- Remember to include the query number as a comment at each step.
- Read your output TXT file before you submit it.