Due Date: Tuesday April 08, 2014 11:00 PM
Points: 35 points max
Turn In: The script and spool files turned in via the assignment drop box

## General Directions

Use the books databases.

These tasks focus on the use of Subqueries. Consequently, you **must use subqueries** to solve the problems. In many cases you could solve the task without the use of subqueries- **but that will not earn any credit** for the assignment.

The following rules are set so that you will learn to use subqueries

- You must do each task with a single query. The query will have sub components but it will be a single query.
- Do not use a join of any kind and do not use a correlated subquery. Do not use a comma join.
- Do not use the view that I provided in the books script- that contains joins; do not create your own views.
- Do not use set operations (Union)
- Do not use a variable.
- Queries that use sub queries tend to be longer and harder to read if they are not formatted properly. Queries that are hard to read will lose points. The key words From, Where, Group By, Order By, Having start new lines and need to align. Subqueries are indented; the indentation should be 2 or 3 spaces. See examples of formatting below

The following rules are rules for writing good queries; these are enforced for this and future assignments.

- If you use column aliases, then use different column aliases in each subquery. Using the same column alias may be legit but it makes your query very hard to read.
- Do not hard code a literal for the current year.
- If you are testing a date or datetime column, use temporal functions. **Do not use wildcard patterns or regular expressions for testing dates.**
- Use only the data supplied in the task to write the query and do not make assumption about the data that are not supported by the create table statements.
- A book with an order quantity of 0 is still considered a book that is ordered. (Some people are testing that the quantity >0- do **not** do that to determine if a book has been ordered.). An order header without any detail lines is still an order.

Acceptable alignment for a subquery. Note that you can easily find the subquery.

```
select cust_id,  ord_id
,  (
     select sum(quantity_ordered)
     from a_oe.order_details  OD
     where OH.ord_id = OD.ord_id
   ) as "NumItemsPerOrder"
from a_oe.order_headers  OH;
```

Not acceptable- the subquery runs on a single line making it harder to see the components

```
select cust_id,  ord_id,
( select sum(quantity_ordered) from a_oe.order_details OD where OH.ord_id = OD.ord_id) as
"NumItemsPerOrder"
from  a_oe.order_headers  OH;
```

Not acceptable: The Select for the subquery needs to be indented a few spaces. It is hard to see where the the subquery starts and ends.

```
select cust_id,  ord_id,  (
select sum(quantity_ordered)
from a_oe.order_details  OD
where OH.ord_id = OD.ord_id) as "NumItemsPerOrder"
from a_oe.order_headers  OH;
```

Not acceptable: The Select for the subquery needs to be indented a few spaces- not a lot of spaces. This also is hard to read.

```
select cust_id,  ord_id,  (
                          select sum(quantity_ordered)
                          from a_oe.order_details  OD
                          where OH.ord_id = OD.ord_id) as "NumItemsPerOrder"
from a_oe.order_headers  OH;
```

## Tasks

**Task 01:** Display the book id and title for any books which someone has ordered and the book is categorized as **either** an SQL book and a database book or possibly both topics. Use the Topic_id to filter for DB and SQL. Sort by the book_id.

**Task 02:** Display the book id and title for any books which someone has ordered and the book is categorized as **both** an SQL book and a database book. Use the Topic_id to filter for DB and SQL. Sort by the book_id.

**Task 03:** Display the book id and title for any books which someone has ordered and the same book is categorized as an SQL book but it is **not** categorized as a database book Use the Topic_id to filter for DB and SQL. Sort by the book_id.

**Task 04:** Display the ID, title of the books and the publication year for the book with the largest sales amount; include ties. For this query, use the total extended cost when determining the sales of a book.

**Task 05:** Display the ID and last name of the customers who have bought in the current year any book by the author(s) with the last name of "Celko". ( Think about how many tables this needs- start from the inner most subquery ( the one that filters for Celko)- and work out.)

**Task 06:** Display the customer id and last name for customers with at least three and no more than five orders last year. Sort by the customer id.

**Task 07:** Display the book id and title for any books where we have orders for more than 500 copies of the book. Use the quantity attribute.

**Task 08:** Display the book id and title for any books which have a list price that is more than the list price of all of the books with a topic id of DB.