Lab 1. Aggregation and Simple Joins

Submission:

- If you decide to skip the lab, make sure you submit the **check-off questions** (highlighted with green background) in a text document named **<lastname>_<firstname>_checkoff.sql** on **Brightspace** before **Wednesday noon.**
- **All students** are expected to submit your answers to all lab questions in a text document with the name **<lastname>_<firstname>_lab1.sql** by the due date, on **Gradescope**.
- Please include both your code and the results in the **.sql** documents for full credits. For detailed requirements, please refer to the Submission Guideline on Brightspace.

Please use Eagle database for all questions in this lab.

Objectives:

- Practice performing single- and multi- table queries that incorporate aggregation
- Practice performing single- and multi- table queries that include a subquery
- Practice performing single- and multi- table queries that use standard Oracle-provided functions

Notes

- Hard code only those values explicitly stated in the problem. Never hard-code values determined dynamically.
- Do NOT include "extra" tables or columns in any query. It reduces query efficiency and increases the likelihood of error. The columns that should be displayed are underlined. ***Including unnecessary or unused tables will result in point deductions.***
- Submissions that fail to follow the format will receive a 50% penalty.
- This lab is worth 50 points in total. You earn 25 with the no-zero policy, and the other 25 will be earned by your submitted answers.

Questions

Order of Execution	
1 1pt	What's the order of execution of the query below? Please include the answer using SQL comments in your submission. Use the same format as below but please update the numbers to reflect the execution order. 1. SELECT NAME, COUNT(*), AVG(RATING) 2. FROM BOOKSHELF 3. WHERE RATING>1 4. GROUP BY CATEGORY_NAME 5. HAVING CATEGORY_NAME LIKE 'A%' 6. ORDER BY COUNT(*);
Aggregation and Subqueries	
2 1pt	Identify *distinctly* all telephone area codes (e.g., the first 3 digits of the phone number) of Colorado (state is 'CO') customers.
3 1pt	Identify all telephone area codes of Colorado customers, as well as the number of customers in each area. Sort the results by the number of customers in *descending* order.
4 1pt	Identify the specific telephone area code containing the *largest* number of Colorado customers. Display the area code only. **(DO NOT hardcode the area code)**
5 1pt	Identify all customers living in the most popular Colorado area code. Display their name in last name, (comma) first name format (e.g. Simpson, Lisa), the city and state in which they live, and their telephone number. *(Hint: the code from the previous question is a good starting point for this question)*
6 1pt	Briefly explain why might we want to know the results of question 4? What business implications does it have? Please include the answer using SQL comments in your submission.
7 1pt	Identify the customer ID and number of orders placed by each customer during August 2010 (OrderDate is August 2010). Sort the results by the number of orders in *descending* order.
8 1pt	Identify the maximum number of orders placed by a customer during August 2010. Display only the maximum number of orders.
9 1pt	Identify the customer ID of the customer who placed the largest number of orders during August 2010, as well as the number of orders placed. *(Hint: the code from the previous questions is a good starting point for this question)*
10 1pt	Identify all customers who placed *greater* than the average number of orders during August 2010. Display the customer ID and the number of orders each of these customers placed. Sort the results by number of orders in *descending* order.

11 1pt	Identify all customers who placed *fewer* than the average number of orders during August 2010. Display the customer ID and the number of orders each of these customers placed. Sort the results by number of orders in *ascending* order.
12 1pt	Briefly explain why a business might want to know the results of questions 10 & 11? Please include the answer using SQL comments in your submission.
Inner Join	
13 1pt	Display the customer ID, company name, contact name in last name, (comma) first name format, (e.g. Simpson, Lisa), and order date in MM.DD.YYYY format (e.g. 12.30.2010) for all Indiana customers who placed orders in 2010. Sort the results by order date from *the oldest to the most recent*.
14 1pt	Display the company name, contact name in title first Initial (dot) last name format e.g. Ms. L. Simpson), order date, and required date for all orders placed by customer with ID C-300001. Sort the results by order date from *the oldest to the most recent*.
15 1pt	For all orders containing 'BOARD GAMES' software, display the order ID, part number, part description, unit price, order quantity, and category name. Sort the results by order quantity in *descending* order.
16 1pt	For all items ordered by customer ID C-300001 on July 14t, 2010, display the order ID, part number, part description, unit price, and order quantity. Sort the results by order quantity from *largest to smallest*.
17 1pt	For all items ordered by 'Bankruptcy Help' (company name) during 2011, display the order date in MM.DD.YYYY format (e.g. 12.30.2010), order ID, part number, part description, unit price, and order quantity. Sort the results first by order date, with the *most recent displayed first*. Then within a given date, sort by quantity, with the *greatest displayed first*.
18 1pt	For all items ordered by 'Bankruptcy Help' (company name) during 2011, display the order date in MM.DD.YYYY format (e.g. 12.30.2010), order ID, part number, and part description. In addition, calculate and display the line item total for each item. To calculate the line item total, multiply the unit price by the number of units ordered. Sort the results first by order date, with the *most recent displayed first*. Then within a given date, sort by quantity, with the *greatest displayed first*. *(Hint: modify your code from question 17)*
Inner Join with Aggregation	
19 1pt	Display the customer ID, company name, contact name in last name, (comma) first name format (e.g. Simpson, Lisa), and number of orders placed (NOT order quantity) for all Indiana customers who placed orders in January of 2011. Sort the results by numbers of orders placed in *ascending* order.

20 1pt	Display the category name and the average stock level of each category. Display up to 2 decimal places for the average stock level. Sort the results by average stock level in *ascending* order.
21 1pt	Display the category detail and the number of part types in each category (NOT stock level). A category detail consists of category name and category description, and it is formatted as category name: (colon) description (e.g. Software: Games, maps). Sort the results by number of part types in *ascending* order.
Putting it all together (Inner Join with Aggregation and Subqueries)	
22 1 pt	Display the weight of the heaviest part in the Software category (CategoryName is 'Software').
23 1pt	For each of the Power, Software, and Storage category, display the category name and the weight of the heaviest part in the category. Sort the results by category name in *ascending* order. *(Hint: the code from the previous question is a good starting point for this question)*
24 1.5pt	For each of the Power, Software, and Storage category, display the category name, the weight of the heaviest part in the category, and the corresponding part description(s) of the heaviest part(s). Sort the results by category name in *ascending* order. *(Hint: the code from the previous question is a good starting point for this question)*
25 0.5pt	Is there anything that can be changed to make it run faster? If no, why? If yes, how? Please include the answer using SQL comments in your submission.