Unit 7 Assignment

Unit Outcomes

- Work with Aggregate Functions.
- Use The GROUP BY and HAVING Clauses.

Course Outcomes:

IT350-4: Create aggregated business report datasets including summarizing, grouping and formatting of output, as well as filtering of summarized data.

Directions:

Using the Adventure Works Database

Create a separate screenshot for each question, Copy your answer separately as text, and place each under the question in a Word document. See Unit 1 Assignment for an example and place each under the question in a Word document. Submit to the Dropbox for the unit Assignment.

Answer all of the following twelve questions using SQL select statements executed in SQL Server Management Studio.

Review The Adventure Works data base schema diagram found in Doc Sharing.

Remember that you can expand the tables in the Object Explorer to see the list of table names and then expand the table to see the list of column names.

The Assignment questions for Units 2 through 9 are included here in the Assignment section for reference and convenience. The questions are also in the Labs for each unit.

It is essential that you do the Assignment questions at the same time that you are doing the labs. Some of the tables used in the Assignments are created while you are doing the Labs.

Many questions will work without the Lab information, but some will not. Do the Assignment questions in the same sequence as found in the Labs. This should be quicker and easier for you as well!

--Using Aggregate Functions

- 1. Write a query that provides the total of sales dollars for all orders placed in December 2004 along with the number of orders. Use Sales.SalesOrderHeader.
- 2. Write a query that shows the lowest cost product in Production.product table.

- 3. Write a query that provides a count of the products on each order along with the total order amount using only Sales.SalesOrderDetail table.

--- How to Group on an Expression

- 4. Write a query that shows the average order amount by month and territoryID.

---Using the WHERE Clause

- 5. Write a query that shows the average order amount by month and territory number for all territories except (5 and 6.)

---Using the HAVING Clause

- 6. Write a query that provides a count of the products on each order along with the total order amount using only Sales.SalesOrderDetail table only for orders with more than six products

---Using DISTINCT and GROUP BY

---Using DISTINCT in an Aggregate Expression

--USE AdventureWorks2012;
--GO

-----SELECT COUNT(*) AS CountOfRows,
----- COUNT(SalesPersonID) AS CountOfSalesPeople,
----- COUNT(DISTINCT SalesPersonID) AS CountOfUniqueSalesPeople
-----FROM Sales.SalesOrderHeader;
-----2
-----SELECT SUM(TotalDue) AS TotalOfAllOrders,
-----SUM(Distinct TotalDue) AS TotalOfDistinctTotalDue
-----FROM Sales.SalesOrderHeader;

- 7. Provide a brief accurate explanation as to how the results from the above query would differ if the DISTINCT keyword was not used in the above two queries.

---Writing Aggregate Queries with Two Tables

--SELECT c.CustomerID, c.AccountNumber, COUNT(*) AS CountOfOrders,
-- SUM(TotalDue) AS SumOfTotalDue
--FROM Sales.Customer AS c
--LEFT OUTER JOIN Sales.SalesOrderHeader AS s ON c.CustomerID = s.CustomerID
--GROUP BY c.CustomerID, c.AccountNumber
--ORDER BY c.CustomerID;
----3
--SELECT c.CustomerID, c.AccountNumber, COUNT(s.SalesOrderID) AS CountOfOrders,
-- SUM(COALESCE(TotalDue, 0)) AS SumOfTotalDue
--FROM Sales.Customer AS c
--LEFT OUTER JOIN Sales.SalesOrderHeader AS s ON c.CustomerID = s.CustomerID
--GROUP BY c.CustomerID, c.AccountNumber
--ORDER BY c.CustomerID;

- 8. Explain what Coalesce does in the query above. How do the results differ if you do not use it?

- 9. Why is a Left Outer Join used in the above query, how do the results differ if you use an Inner Join instead? Which results would be considered most correct? Why?

- 10. Write the question that would lead to the above query being the answer.

- 11. Write a query joining the Person.Person, Sales.Customer, and Sales.SalesOrderHeader tables to return a list of the customer names along with a count of the orders placed.

- 12. Write a query using the Sales.SalesOrderHeader, Sales.SalesOrderDetail, and Production.Product tables to display the total sum of products by ProductID and OrderDate.

Review the grading rubric below before beginning this Activity.

Assignment grading rubric = 40 points

<table>
<thead>
<tr>
<th>Assignment Criteria</th>
<th>Points possible</th>
<th>Points earned by student</th>
</tr>
</thead>
<tbody>
<tr>
<td>Includes a correct and complete answer to each of the twelve (12) questions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>- 1. Write a query that provides the total of sales dollars for all orders placed in December 2004 along with the number of orders. Use Sales.SalesOrderHeader.</td>
<td>0 - 3</td>
<td></td>
</tr>
<tr>
<td>- 2. Write a query that shows the lowest cost product in Production.product table.</td>
<td>0 - 3</td>
<td></td>
</tr>
<tr>
<td>- 3. Write a Query that provides a count of the products on each order along with the total order amount using only Sales.SalesOrderDetail table.</td>
<td>0 - 4</td>
<td></td>
</tr>
<tr>
<td>- 4. Write a query that shows the average order amount by month and territoryID.</td>
<td>0 - 4</td>
<td></td>
</tr>
<tr>
<td>- 5. Write a query that shows the average order amount by month and territory number for all territories except (5 and 6.)</td>
<td>0 - 3</td>
<td></td>
</tr>
<tr>
<td>- 6. Write a Query that provides a count of the products on each order along with the total order</td>
<td>0 - 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Question</td>
<td>Points</td>
</tr>
<tr>
<td>---</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>1</td>
<td>amount using only Sales.SalesOrderDetail table only for orders with more than six products.</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>- 7. Provide a brief accurate explanation as to how the results from the above query would differ if the DISTINCT keyword was not used in the above two queries.</td>
<td>0 - 4</td>
</tr>
<tr>
<td>3</td>
<td>- 8. Explain what Coalesce does in the query above. How do the results differ if you do not use it?</td>
<td>0 - 4</td>
</tr>
<tr>
<td>4</td>
<td>- 9. Why is a Left Outer Join used in the above query, how do the results differ if you use an Inner Join instead? Which results would be considered most correct? Why?</td>
<td>0 - 3</td>
</tr>
<tr>
<td>5</td>
<td>- 10. Write the question that would lead to the above query being the answer.</td>
<td>0 - 3</td>
</tr>
<tr>
<td>6</td>
<td>- 11. Write a query joining the Person.Person, Sales.Customer, and Sales.SalesOrderHeader tables to return a list of the customer names along with a count of the orders placed.</td>
<td>0 - 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

12. Write a query using the Sales.SalesOrderHeader, Sales.SalesOrderDetail, and Production.Product tables to display the total sum (line total) of products by ProductID and OrderDate.

Column Total | 0 - 40

Copyright Kaplan University