## TSQL-Mod09 - Lab Exercise - Grouping and Aggregating Data

# SQL Server - TSQL Practitioner Details:

Please enter your details below:

|  |  |  |
| --- | --- | --- |
| **SQL Server - TSQL Practitioner:** | | |
|  | Name: |  |
| Email Address: |  |
| Course: |  |
| Date: |  |

# Lab Introduction

This session aims to familiarise you with: **TSQL-Mod09 - Grouping and Aggregating Data**

**In the Labs you will need to run the following Lecture Demos - .sql files in SQL Server – SSMS to familiarise yourselves with the TSQL Module topic.**

**You then write your own TSQL solutions for the following Lab Exercises .sql files in SQL Server – SSMS.**

Provide your **TSQL Exercise Solution** and SSMS Screenshots within this document.

*Note the symmetry between Lecture Demos – Lab Exercises. We then require you to produce your own ICA Demos on each of the TSQL Module topics covered.*

**Lecture TSQL Demos .sql files:**

Lecture Demo A - Using Aggregate Functions

Lecture Demo B - Using the GROUP BY Clause

Lecture Demo C - Filtering Groups with HAVING

**Lab TSQL Exercises .sql files:**

Lab Exercise 1: Using Aggregate Functions

Lab Exercise 2: Using the GROUP BY Clause

Lab Exercise 3: Filtering Groups with HAVING

Lab Exercise 4: Filtering Groups with HAVING

**In this Lab Session you will need to:**

1. Run the Lecture TSQL Demos to become familiar with the TSQL Code.
2. Review the code as it will support you in writing solutions to Lab Exercises.
3. Write TSQL solutions to Lab Exercises.
4. Provide additional comparable TSQL Demos & solutions to demonstrate your TSQL programming capabilities.

**After this Lab Session you will need to:**

**ICA TSQL Demo Requirements:** Provide towards your **ICA TSQL Portfolio** of comparable **TSQL Module - ICA Demos**:

1. Consolidate your TSQL Lab Exercises.
2. Provide *comparable* **ICA TSQL Demo Exercises** and **Solutions**.

**ICA Demos: TSQL09 – Grouping and Aggregating Data**

ICA Demo A - Using Aggregate Functions

ICA Demo B - Using the GROUP BY Clause

ICA Demo C - Filtering Groups with HAVING

# Lab Scenario Overview

* For many [**Graduate Developer Jobs**](https://www.google.com/search?q=graduate+developer+-+SQL+Server+jobs&sxsrf=ALiCzsZ2Jt_reKLVqPZSXqs7CQT3nFHQLA%3A1663318853970&ei=RTskY63tOpKkgQaI06zABQ&ved=0ahUKEwjto8HL-Zj6AhUSUsAKHYgpC1gQ4dUDCA4&uact=5&oq=graduate+developer+-+SQL+Server+jobs&gs_lcp=Cgdnd3Mtd2l6EAMyBQghEKABMgUIIRCgATIICCEQHhAWEB0yCAghEB4QFhAdMggIIRAeEBYQHTIICCEQHhAWEB0yCAghEB4QFhAdMggIIRAeEBYQHTIICCEQHhAWEB0yCAghEB4QFhAdOgoIABBHENYEELADOg0IABBHENYEELADEMkDOgQIIxAnOgUIABCABDoGCAAQHhAHOgcIIxDJAxAnOgUIABCSAzoJCAAQHhDJAxAWOgYIABAeEBY6BQgAEIYDOgcIIRCgARAKOgQIIRAVSgQIQRgASgQIRhgAUKACWI46YKY-aAJwAXgAgAGCAYgB9g6SAQQxNi41mAEAoAEByAEEwAEB&sclient=gws-wiz) you are expected to have a command of **TSQL** to interact with **Server Databases** or **RDBMS**.
* You are a **Junior Graduate Developer** tasked with working with **SQL Server Database: TSQL** to practice your **TSQL Coding skills**.
* You have a ‘**set of business requirements’** and required to generate or write **T- SQL queries** to **retrieve** relevant data from a table or **related table(s).**
* You will need to perform **calculations** upon **groups of data** and **filter** according to the **results**.
* You are to develop your own **TSQL programming skills** and provide your own **TSQL Code solutions** to the **Lab Exercises**.
* You may use the **SSMS - Design Query Editor** or write your own **TSQL Code**.



* We do provide **TSQL Module Solution** to the **Lab Exercises** to help you review and consolidate your own solutions.

# Objectives

After completing this lab, you will be able to:

* Write queries that use the GROUP BY clause.
* Write queries that use aggregate functions.
* Write queries that use distinct aggregate functions.
* Write queries that filter groups with the HAVING clause.
* Estimated Time: Lab - 120 Minutes + ICA - 120 Minutes

## Review a SQL Server Video

# Task A: Preparation for TSQL Topic

1. Open the **TSQL Module lecture powerpoint**
2. Review **one** of the recommended **SQL Server videos** (slide 8?) from the Developer Community. You only need a quick overview on the TSQL topic but refer to others as you see fit.
3. This will help you prepare to run the lecture TSQL demos and write your own TSQL solutions to the Lab Exercises.

|  |
| --- |
| SQL Server Video Resource(s) consulted for this topic: |

[insert the recommended videos from the Lecture in this section]

[highlight the video you reviewed]

|  |
| --- |
| SQL Server Web Resource(s) consulted for this topic: |

[insert the recommended Web Resource – Hyperlinks from the Lecture slides here]

[highlight those Web Resource – Hyperlinks you have reviewed and recommend]

## Connect to SQL Server 2019 using SSMS

# Task B: Run SQL Server.

1. Launch **SQL Server Management Studio SSMS**
   1. **TU Labs:** run **AppsAnwhere** and search for **SQL** or **SSMS**.
   2. **Home:** from your task bar search for **SQL** or **SSMS**
2. Connect to **SQL Server 2019** engine:
   1. Server Name: [**PC name] or** [pc number] or [localhost]
   2. Authentication: [windows authentication]

## Lab Setup Steps:

# Task C: Download the .sql work files for the Lab Session.

1. Download the TSQL Module **Lecture TSQL Demos, Lab Exercises and Lab Solutions .sql files** onto a folder on your hard drive:

eg. **D:\RnD\TSQL09-Grouping and Aggregation** folder.

## Review the Lecture TSQL Demo A: Writing Queries That Use the GROUP BY Clause

# Task 1: Reviewing the Lecture Demo in preparation for the Lab Exercises

1. Open the **TSQL-Mod09 - Lecture Demo A - Using Aggregate Functions .sql file in SSMS**.
   * **TU Labs:** open in Notepad and copy the **TSQL Code** into **SSMS** – **New Query window**.
   * **Home:** double click to open the **.sql file** automatically in **SSMS** – **New Query window**.
2. Note the **.sql files** comprises of business requirement in the form of Question and/or Task requiring TSQL to obtain the desired data or results from the database.
3. Discuss in your study group what each segment or steps of **TQL code** does and the **results** you expect to see. What tables? What rows, columns, or data.
4. Run the **TSQL code** and examine the results.
5. The lecture **TSQL Lecture Demos** will support you when writing **TSQL Code** for the **Lab Exercises** below.

## TSQL-Mod09 Lab Exercise 1: Using Aggregate Functions

**Scenario:** As a Graduate SQL Developer, you want a better understanding of your corporate data. Usually, the best approach for an initial project is to get an overview of the main tables and columns, so you can better understand different business requirements.

As a Graduate SQL Developer, you will be generating or writing TSQL Code to demonstrate your capability to query the sample SQL Databases we have provided.

# Exercise 1: Writing Queries That Use the GROUP BY Clause

**Scenario**

The sales department want to create additional upsell opportunities from existing customers. The staff need to analyze different groups of customers and product categories, depending on several business rules. Based on these rules, you will write SELECT statements to retrieve the needed rows from the Sales.Customers table.

The main tasks for this exercise are as follows:

1. Prepare the Lab Environment
2. Write a SELECT Statement to Retrieve Different Groups of Customers
3. Add an Additional Column from the Sales.Customers Table
4. Write a SELECT Statement to Retrieve the Customers with Orders for Each Year
5. Write a SELECT Statement to Retrieve Groups of Product Categories Sold in a Specific Year

# Task 2: Write a SELECT Statement to Retrieve Different Groups of Customers

1. Open in SSMS:

**TSQL-Mod09 Lab Exercise 1: Using Aggregate Functions .sql**

1. Ensure that you are connected to the **TSQL** database.
2. Write a SELECT statement that will return groups of customers who made a purchase. The SELECT clause should include the **custid** column from the **Sales.Orders** table, and the **contactname** column from the **Sales.Customers** table. Group both columns and filter only the orders from the sales employee whose empid equals five.
3. Write a SELECT statement that will return all rows and all columns from the **Sales.Customers** table.
4. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
5. Ensure you provide your **TSQL Code** and **SSMS Screenshot Evidence** below:

|  |
| --- |
| -- TSQL Code |
| ---------------------------------------------------------------------  -- TSQL-Mod09 - Grouping and Aggregating Data  -- Author: [insert your name] [date]  -- TSQL-Mod09 - Lab Exercise 1 - Using Aggregate Functions  ---------------------------------------------------------------------  USE TSQL;  GO  ---------------------------------------------------------------------  -- Task 2  -- Write a SELECT statement that will return groups of customers that made a purchase.  -- The SELECT clause should include the custid column from the Sales.Orders table and the  -- contactname column from the Sales.Customers table.  -- Group by both columns and filter only the orders from the sales employee whose empid equals five.  -- Execute the written statement and review the results you have obtained.  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 3: Add an Additional Column From the Sales.Customers Table

1. Copy the T-SQL statement in task 1 and modify it to include the **city** column from the **Sales.Customers** table in the SELECT clause.
2. Execute the query.
3. You will get an error. What is the error message? Why?
4. Correct the query so that it will execute properly.
5. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
6. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- TSQL-Mod09 - Grouping and Aggregating Data  -- Author: [insert your name] [date]  ---------------------------------------------------------------------  -- Task 3  -- Copy the T-SQL statement in task 2 and modify it to include the city column  -- from the Sales.Customers table in the SELECT clause.  -- Execute the query. You will get an error. What is the error message? Why?  -- Correct the query so that it will execute properly.  -- Execute the query and compare the results that you got with the desired  -- results shown in the file 53 - Lab Exercise 1 - Task 2 Result.txt.  ---------------------------------------------------------------------  -- error example |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 4: Write a SELECT Statement to Retrieve the Customers with Orders for Each Year

1. Write a SELECT statement that will return groups of rows based on the **custid** column and a calculated column **orderyear** representing the order year based on the **orderdate** column from the **Sales.Orders** table. Filter the results to include only the orders from the sales employee whose empid equals five.
2. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
3. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 4  -- Write a SELECT statement that will return groups of rows based on the custid  -- column and a calculated column orderyear representing the order year based on  -- the orderdate column from the Sales.Orders table. Filter the results to include  -- only the orders from the sales employee whose empid equal five.  -- Execute the written statement and compare the results with your study group.  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 5: Write a SELECT Statement to Retrieve Groups of Product Categories Sold in a Specific Year

1. Write a SELECT statement to retrieve groups of rows based on the **categoryname** column in the **Production.Categories** table. Filter the results to include only the product categories that were ordered in the year 2008.
2. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
3. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 4  -- Write a SELECT statement that will return groups of rows based on the custid  -- column and a calculated column orderyear representing the order year based on  -- the orderdate column from the Sales.Orders table. Filter the results to include  -- only the orders from the sales employee whose empid equal five.  -- Execute the written statement and compare the results with your study group.  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

## Review the TSQL-Mod09 - Lecture Demo B - Using the GROUP BY Clause - AW2019

# Task 1: Reviewing the Lecture Demo in preparation for the Lab Exercises

1. Open the **TSQL-Mod09 - Lecture Demo B - Using the GROUP BY Clause -** **AW2019.sql file in SSMS**.
   * Home: double click to open the **.sql file** to open automatically in SSMS in notepad and copy the **TSQL code** into **SSMS**.
   * TU Labs: open in Notepad and copy the TSQL Code into SSMS – New Query window.
2. Note the **.sql files** comprises of Question, Task or business requirement and the corresponding TSQL code solutions to obtain the desired data or results from the database.
3. Discuss in your study group what each segment or steps of **TQL code** does and the **results** you expect to see. **What tables? What rows, columns or data are we working with?**
4. Run the **TSQL code** and examine the results.
5. The lecture TSQL demos will support you when writing TSQL Code for the Lab Exercises below.

## TSQL-Mod09 - Lab Exercise 2 - Using the GROUP BY Clause

**Scenario:** As a Graduate SQL Developer, you want a better understanding of your corporate data. Usually, the best approach for an initial project is to get an overview of the main tables and columns, so you can better understand different business requirements.

As a Graduate SQL Developer, you will be generating or writing TSQL Code to demonstrate your capability to query the sample SQL Databases we have provided.

# Exercise 2: Writing Queries That Use Aggregate Functions

**Scenario**

The marketing department wants to launch a new campaign, so the staff need to gain a better insight into the existing customers’ buying behavior. You should create different sales reports, based on the total and average sales amount per year and per customer.

The main tasks for this exercise are as follows:

* 1. Write a SELECT statement to Retrieve the Total Sales Amount Per Order
  2. Add Additional Columns
  3. Write a SELECT Statement to Retrieve the Sales Amount Value Per Month
  4. Write a SELECT Statement to List All Customers with the Total Sales Amount and Number of Order Lines Added

# Task 1: Write a SELECT statement to Retrieve the Total Sales Amount Per Order

1. Open the T-SQL script for **Lab Exercise 2.sql**. Ensure that you are connected to the **TSQL** database.
2. Write a SELECT statement to retrieve the **orderid** column from the **Sales.Orders** table and the total sales amount per orderid. (Hint: multiply the **qty** and **unitprice** columns from the **Sales.OrderDetails** table.) Use the alias **salesamount** for the calculated column. Sort the result by the total sales amount in descending order.
3. Execute your TSQL Code and compare the results that you achieved with your study group or consolidate with the model solution provided.

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- TSQL-Mod09 - Grouping and Aggregating Data  -- Author: [insert your name] [date]  -- TSQL-Mod09 - Lab Exercise 2 - Using the GROUP BY Clause  ---------------------------------------------------------------------  USE TSQL;  GO  ---------------------------------------------------------------------  -- Task 1  --  -- Write a SELECT statement to retrieve the orderid column from the Sales.Orders  -- table and the total sales amount per orderid. (Hint: Multiply the qty and  -- unitprice columns from the Sales.OrderDetails table.) Use the alias salesmount  -- for the calculated column. Sort the result by the total sales amount in  -- descending order.  -- Execute the written statement and compare the results that you got with the desired results shown in in the solution .sql.  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 2: Add Additional Columns

1. Copy the T-SQL statement in task 1 and modify it to include the total number of order lines for each order and the average order line sales amount value within the order. Use the aliases **nooforderlines** and **avgsalesamountperorderline**, respectively.
2. Execute your TSQL Code and compare the results that you achieved with your study group or consolidate with the model solution provided.

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 2  --  -- Copy the T-SQL statement in task 1 and modify it to include the total number of  -- order lines for each order and the average order line sales amount value within  -- the order. Use the aliases nooforderlines and avgsalesamountperorderline,  -- respectively.  --  -- Execute the written statement and compare the results that you got with the  -- recommended result shown in the file 63 - Lab Exercise 2 - Task 2 Result.txt.  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 3: Write a SELECT Statement to Retrieve the Sales Amount Value Per Month

1. Write a select statement to retrieve the total sales amount for each month. The SELECT clause should include a calculated column named **yearmonthno** (YYYYMM notation), based on the **orderdate** column in the **Sales.Orders** table and a total sales amount (multiply the **qty** and **unitprice** columns from the **Sales.OrderDetails** table). Order the result by the **yearmonthno** calculated column.
2. Execute your TSQL Code and compare the results that you achieved with your study group or consolidate with the model solution provided.

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 3  --  -- Write a select statement to retrieve the total sales amount for each month. The SELECT clause should include a calculated column named yearmonthno (YYYYMM notation) based on the orderdate column in the Sales.Orders table and a total sales amount (multiply the qty and unitprice columns from the Sales.OrderDetails table). Order the result by the yearmonthno calculated column.  --  -- Execute your TSQL Code and compare the results that you achieved with your study group or consolidate with the model solution provided.  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 4: Write a SELECT Statement to List All Customers with the Total Sales Amount and Number of Order Lines Added

1. Write a select statement to retrieve all the customers (including those who did not place any orders) and their total sales amount, maximum sales amount per order line, and number of order lines.
2. The SELECT clause should include the **custid** and **contactname** columns from the **Sales.Customers** table and four calculated columns based on appropriate aggregate functions:
   1. **totalsalesamount**, representing the total sales amount per order
   2. **maxsalesamountperorderline**, representing the maximum sales amount per order line
   3. **numberofrows**, representing the number of rows (use \* in the COUNT function)
   4. **numberoforderlines**, representing the number of order lines (use the **orderid** column in the COUNT function)
3. Order the result by the **totalsalesamount** column.
4. Execute your TSQL Code and compare the results that you achieved with your study group or consolidate with the model solution provided.
5. Notice that the **custid** 22 and 57 rows have a NULL in the columns with the SUM and MAX aggregate functions. What are their values in the **COUNT** columns? Why are they different?

|  |
| --- |
| *-- TSQL Code:* |
| -- Task 4  --  -- Write a select statement to retrieve all the customers (including those that did not place any orders) and their total sales amount, maximum sales amount per order line, and number of order lines.  --  -- The SELECT clause should include the custid and contactname columns from the Sales.Customers table and four calculated columns based on appropriate aggregate functions:  -- totalsalesamount, representing the total sales amount per order  -- maxsalesamountperorderline, representing the maximum sales amount per order line  -- numberofrows, representing the number of rows (use \* in the COUNT function)  -- numberoforderlines, representing the number of order lines (use the orderid column in the COUNT function)  --  -- Order the result by the totalsalesamount column.  -- Execute your TSQL Code and compare the results that you achieved with your study group or consolidate with the model solution provided.  -- Notice that the custid 22 and 57 rows have a NULL in the columns with the SUM and MAX aggregate functions. What are their values in the COUNT columns? Why are they different?  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

## Review the TSQL-Mod09 - Lecture Demo C - Fiiltering Groups with HAVING - AW2019

# Task 1: Reviewing the Lecture Demo in preparation for the Lab Exercises

1. Open:

**TSQL-Mod09 - Lecture Demo C - Filtering Groups with HAVING - AW2019.sql file in SSMS**.

* + Home: double click to open the **.sql file** to open automatically in SSMS in notepad and copy the **TSQL code** into **SSMS**.
  + TU Labs: open in Notepad and copy the TSQL Code into SSMS – New Query window.

1. Note the **.sql files** comprises of Question, Task or business requirement and the corresponding TSQL code solutions to obtain the desired data or results from the database.
2. Discuss in your study group what each segment or steps of **TQL code** does and the **results** you expect to see. **What tables? What rows, columns or data are we working with?**
3. Run the **TSQL code** and examine the results.
4. The lecture TSQL demos will support you when writing TSQL Code for the Lab Exercises below.

## TSQL-Mod09 - Lab Exercise 3 - Filtering Groups with HAVING

# Exercise 3: Writing Queries That Use Distinct Aggregate Functions

**Scenario**

The marketing department want to have some additional reports that display the number of customers who made any order in a specific time period and the number of customers based on the first letter in the contact name.

The main tasks for this exercise are as follows:

1. Modify a SELECT Statement to Retrieve the Number of Customers
2. Write a SELECT Statement to Analyze Segments of Customers
3. Write a SELECT Statement to Retrieve Additional Sales Statistics

# Task 1: Modify a SELECT Statement to Retrieve the Number of Customers

1. Open the **TSQL-Mod09 - Lab Exercise 3 - Filtering Groups with HAVING.sql**.
2. Ensure that you are connected to the **TSQL** database.
3. A junior analyst prepared a T-SQL statement to retrieve the number of orders and the number of customers for each order year.
4. Copy the provided T-SQL code and execute it in SSMS:

USE TSQL;

GO

SELECT

YEAR(orderdate) AS orderyear, COUNT(orderid) AS nooforders, COUNT(custid) AS noofcustomers FROM Sales.Orders

GROUP BY YEAR(orderdate);

1. Observe the results. Notice that the number of orders is the same as the number of customers. Why?
2. Amend the T-SQL statement to show the correct number of customers who placed an order for each year.
3. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
4. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- TSQL-Mod09 - Grouping and Aggregating Data  -- Author: [insert name][student#][date]  -- TSQL-Mod09 - Lab Exercise 3 - Filtering Groups with HAVING  ---------------------------------------------------------------------  USE TSQL;  GO  ---------------------------------------------------------------------  -- Task 1  --  -- A junior analyst prepared a T-SQL statement to retrieve the number of orders and the number of customers for each order year. Observe the provided T-SQL statement and execute it:  --  -- Observe the result and notice that the number of orders is the same as the number of customers. Why?  --  -- Correct the T-SQL statement to show the correct number of customers that placed an order for each year.  --  -- Execute the written statement and compare the results  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 2: Write a SELECT Statement to Analyze Segments of Customers

1. Write a SELECT statement to retrieve the number of customers based on the first letter of the values in the **contactname** column from the **Sales.Customers** table. Add an additional column to show the total number of orders placed by each group of customers. Use the aliases **firstletter**, **noofcustomers** and **nooforders**. Order the result by the **firstletter** column.
2. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
3. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 2  --  -- Write a SELECT statement to retrieve the number of customers based on the first letter of the values in the contactname column from the Sales.Customers table. Add an additional column to show the total number of orders placed by each group of customers. Use the aliases firstletter, noofcustomers and nooforders. Order the result by the firstletter column.  --  -- Execute the written statement and compare the results  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

* + **Task 3: Write a SELECT Statement to Retrieve Additional Sales Statistics**

1. Copy the T-SQL statement in exercise 1, task 5, and modify to include the following information about each product category—total sales amount, number of orders, and average sales amount per order. Use the aliases **totalsalesamount, nooforders**, and **avgsalesamountperorder**, respectively.
2. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
3. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 3  --  -- Copy the T-SQL statement in exercise 1, task 4, and modify to include the following information about for each product category: total sales amount, number of orders, and average sales amount per order. Use the aliases totalsalesamount, nooforders, and avgsalesamountperorder, respectively.  --  -- Execute the written statement and compare the results  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

**Results**: After this exercise, you should have an understanding of how to apply a DISTINCT aggregate function.

# Exercise 4: Writing Queries That Filter Groups with the HAVING Clause

**Scenario**

The sales and marketing departments were satisfied with the reports you provided to analyze customers’ behavior. Now they would like to have the results filtered, based on the total sales amount and number of orders. So, in the final exercise, you will learn how to filter the result, based on aggregated functions, and learn when to use the WHERE and HAVING clauses.

The main tasks for this exercise are as follows:

1. Write a SELECT Statement to Retrieve the Top 10 Customers
2. Write a SELECT Statement to Retrieve Specific Orders
3. Apply Additional Filtering
4. Retrieve the Customers with More Than 25 Orders

# Task 1: Write a SELECT Statement to Retrieve the Top 10 Customers

1. Open the T-SQL script **81 - Lab Exercise 4.sql**. Ensure that you are connected to the **TSQL** database.
2. Write a SELECT statement to retrieve the top 10 customers (by total sales amount) who spent more than $10,000. Display the **custid** column from the **Orders** table and a calculated column that contains the total sales amount, based on the **qty** and **unitprice** columns from the **Sales.OrderDetails** table. Use the alias **totalsalesamount** for the calculated column.
3. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
4. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- TSQL-Mod09 - Grouping and Aggregating Data  -- Author: [insert name][student#][date]  -- TSQL-Mod09 - Lab Exercise 4 - Filtering Groups with HAVING  ---------------------------------------------------------------------  USE TSQL;  GO  ---------------------------------------------------------------------  -- Task 1  --  -- Write a SELECT statement to retrieve the top 10 customers by total sales amount that spent more than $10,000 in terms of sales amount. Display the custid column from the Orders table and a calculated column that contains the total sales amount based on the qty and unitprice columns from the Sales.OrderDetails table. Use the alias totalsalesamount for the calculated column.  --  -- Execute the written statement and compare the results  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 2: Write a SELECT Statement to Retrieve Specific Orders

1. Write a SELECT statement against the **Sales.Orders** and **Sales.OrderDetails** tables, and display the **empid** column and a calculated column representing the total sales amount. Filter the results to group only the rows with an order year 2008.
2. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
3. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 2  --  -- Write a SELECT statement against the Sales.Orders and Sales.OrderDetails tables and display the empid column and a calculated column representing the total sales amount. Filter the result to group only the rows with an order year 2008.  --  -- Execute the written statement and compare the results  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

* + **Task 3: Apply Additional Filtering**

1. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
2. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:
3. Apply an additional filter to show only employees with empid equal to 3.
4. Execute the written statement and compare the results that you achieved with the recommended results.
5. Did you apply the predicate logic in the WHERE clause or the HAVING clause? Which do you think is better? Why?

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 3  --  -- Copy the T-SQL statement in task 2 and modify it to apply an additional filter to retrieve only the rows that have a sales amount higher than $10,000.  --  -- Execute the written statement and compare the results that you got with the recommended result  --  -- Apply an additional filter to show only employees with empid equal number 3.  --  -- Execute the written statement and compare the results  --  -- Did you apply the predicate logic in the WHERE or in the HAVING clause? Which do you think is better? Why?  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

# Task 4: Retrieve the Customers with More Than 25 Orders

1. Write a SELECT statement to retrieve all customers who placed more than 25 orders and add information about the date of the last order and the total sales amount. Display the **custid** column from the **Sales.Orders** table and two calculated columns— **lastorderdate** based on the **orderdate** column, and **totalsalesamount** based on the **qty** and **unitprice** columns in the **Sales.OrderDetails** table.
2. Execute your TSQL Code and compare the results that you achieved with your study group and consolidate with the model solution provided.
3. Ensure you provide your TSQL Code and SSMS Screenshot Evidence below:

|  |
| --- |
| *-- TSQL Code:* |
| ---------------------------------------------------------------------  -- Task 4  --  -- Write a SELECT statement to retrieve all customers who placed more than 25 orders and add information about the date of the last order and the total sales amount. Display the custid column from the Sales.Orders table and two calculated columns: lastorderdate based on the orderdate column and totalsalesamount based on the qty and unitprice columns in the Sales.OrderDetails table.  --  -- Execute the written statement and compare the results  --------------------------------------------------------------------- |

|  |
| --- |
| **SSMS Screenshot Evidence and/or Result:** |
|  |

**Results**: After this exercise, you should understand how to use the HAVING clause.

## Module Review and Takeaways

In this lesson, you have learned how to:

* + List the built-in aggregate functions provided by SQL Server.
  + Write queries that use aggregate functions in a SELECT list to summarize all the rows in an input set.
  + Describe the use of the DISTINCT option in aggregate functions.
  + Write queries using aggregate functions that handle the presence of NULLs in source data.

## ICA Requirements

ICA Requirements: You also need provide a portfolio of comparable TSQL Module - ICA Demos:

**ICA Demos: TSQL09 – Grouping and Aggregating Data**

ICA Demo A - Using Aggregate Functions

ICA Demo B - Using the GROUP BY Clause

ICA Demo C - Filtering Groups with HAVING

# Review the sample ICA Demos

Sample **ICA Demos: TSQL09 – Grouping and Aggregating Data by Mansha Nawaz**

Provided are draft ICA Submissions by [module leader: m. Nawaz] for this TSQL Module. Note you replicate this activity for all the TSQL Modules you cover in the Lab sessions.

**.sql file:**

ICA TSQL-Mod09 - ICA Demos - Grouping and Aggregating Data - Human Resources by Mansha Nawaz

*complete.*

**Powerpoint file:**

ICA TSQL-Mod09 - ICA Demos - Grouping and Aggregating Data - Human Resources by Mansha Nawaz

*incomplete.*

Note: You may produce a Report rather than a Powerpoint.

# ICA Advice:

Develop comparable TSQL Demo as indicated above. Your TSQL Demo examples should demonstrating your command of the TSQL Modules you cover.

For each set of TSQL Module Demos:

* Write a simple scenario and indicate what you going to be doing in terms of a TSQL Demo. (Sentence or Paragraph – similar comments as provided in the examples provided)
* Your TSQL Demos should be comparable with what has been covered in either Lecture or Lab Demos.
* Identify steps you may go through for your TSQL Demos.
* Save your **TSQL Demos .sql** files as they contribute to your ICA - TSQL portfolio.
* Copy & paste both the TSQL Code and SSMS Screenshot in a ICA PowerPoint or Report.
* Remember we advise: Lab 2hr session + ICA TSQL Demo 2hr session
* Time permitting you can always come back and improve on your solutions.
* It is important you proceed and cover all desired TSQL Modules 09-18 in your Lab sessions.