Sorting Rows

Exercise Handout

Contents

Overview

In this set of exercises, you will work with the Products table from the Northwind database to produce a list of products sorted by category and by unit price within those categories.

In the second exercise, you will revisit an earlier product query to sort the output.

The third exercise asks you to produce a query of unique countries from the Customers table.

The fourth exercise involves listing the ten most expensive products.

Objectives

At the end of this lab, you will be able to:

- write a query that sorts on a single column
- write a query that sorts on multiple columns
- write a query that sorts on a calculated column
- write a query that selects unique values
- write a query that returns only a specified number of rows

Setup: Launch SQL Server Management Studio (if necessary)

1. Launch the virtual machine.
2. Launch SQL Server Management Studio.
3. Connect to the server.

Exercise 1: Basic ORDER BY clauses

Northwind Traders are attempting to learn more about their product pricing. You have been asked to produce a list of current products, sorted by category and then by unit price in reverse order.

In this exercise, you will use basic ORDER BY clauses.

The main tasks for this exercise are as follows:

1. Write a query that selects only the rows in the Products table which have a value of 0 (zero) in the Discontinued column.
2. Modify the query so that it only returns non-discontinued Products sorted by their CategoryID.
3. Modify the query to include an additional sort that sorts on UnitPrice from highest to lowest.

Task 1: Write a query that only returns current products

1. Create a new query and save it with a name of "CurrentProducts.sql".
2. Write a query that uses the Northwind database and displays the ProductID, ProductName, CategoryID, Discontinued and UnitPrice columns from the Products table and only selects rows with a Discontinued value of 0.
3. Execute the query and verify that it returns 69 rows.

Task 2: Write a query that sorts current products by category

1. Modify your existing query.
2. Add an ORDER BY clause to the query that sorts the results based on the CategoryID column.
3. Execute the query and verify that it returns the 69 rows but that all the products with a CategoryID of 1 are together, all those with CategoryID 2 are together, *etc*.

Task 3: Write a query that sorts current products by category and unit price

1. Modify your existing query.
2. Add an additional statement to the ORDER BY clause that sorts the results in reverse order on UnitPrice.
3. Execute the query and verify that your 69 rows are still sorted by CategoryID but that within those categories the most expensive products are displayed first.

Exercise 2: Sorting on calculated columns

Northwind Traders are still trying to get a handle on their stock levels. They are really pleased with a stock query you wrote earlier but now that they know you can sort data, they'd like you to make it easier to see which products have the highest value of future stock.

In this exercise, you will work on an earlier query and add an ORDER BY clause to it.

The main tasks for this exercise are as follows:

1. Re-use the following query from an earlier exercise.
2. Modify the query so that it sorts the results in reverse order of future stock value.

Task 1: Re-use an existing query

1. Create a copy of the script file "StockList.sql".
2. Rename the new file "SortedStockList.sql".

   NOTE: If you can't find the query file, the SQL is reproduced below:

```
SELECT
  ProductID, ProductName, UnitPrice, UnitsInStock, UnitsOnOrder,
  UnitPrice * UnitsInStock AS CurrentStockValue,
  (UnitsInStock + UnitsOnOrder) * UnitPrice AS FutureStockValue
FROM
  Products
```

3. Open the query in SSMS.
4. Execute the query and verify that it returns 77 rows.

Task 2: Write a query that sorts on a calculated column

1. Modify the existing query.
2. Add an order by clause that sorts on the value of the calculated "FutureStockValue" column, in reverse order.

   NOTE: There are three different ways of achieving this.
3. Execute the query and verify that the first product returned is Cote de Blaye, product id 38.

Exercise 3: SELECT DISTINCT

You have been asked to produce a list of the unique countries in which Northwind Traders has customers.

In this exercise, you will use the DISTINCT modifier on a SELECT statement.

The main tasks for this exercise are as follows:

1. Write a query that selects the Country column from the Customers table.
2. Modify the query so that it only returns a single row for duplicate country names.

Task 1: Write a query that selects the customer's countries

1. Create a new query and save it with a name of "CustomerCountries.sql".
2. Write a query that uses the Northwind database and selects the Country column from the Customers table.
3. Execute the query and verify that it returns 91 rows.

Task 2: Select distinct rows

1. Modify the existing query.
2. Modify the SELECT statement so that it looks for distinct values only.
3. You're expecting 21 rows now. Notice that the countries are sorted alphabetically as well.

Exercise 4: SELECT TOP

You have been asked to produce a list of the top ten most expensive products based on unit price.

In this exercise, you will use the TOP modifier.

The main task for this exercise is as follows:

1. Write a query that selects the top ten rows from the Products table sorted on reverse value of unit price.

Task 1: Write a query of the top ten most expensive products

1. Create a new query and save it with a name of “MostExpensiveProducts.sql”.
2. Write a query that uses the Northwind database and selects the ProductID, ProductName and UnitPrice columns from the Products table.
3. Order the results in descending order of UnitPrice.
4. Restrict the results to the first ten results.
5. You should retrieve 10 rows. Cote de Blaye, unsurprisingly, should be the first.

Learn. To Change.

QA.com