Filtering Rows

Exercise Handout

Learn. To Change.

Contents

Overview

In this set of exercises, you will first work with the Products table from the Northwind database to produce a list of products in a certain category, adding various different filters to the query.

Then you will work with the Orders table to find orders that were placed between certain dates for certain customers.

The third exercise involves working with string comparisons on the Customers table.

The last exercise involves finding rows containing NULL values in the Customers table.

The “optional extras” are some final suggestions for additional tasks that you may choose to attempt if you are feeling confident. They are intended to inspire you to play around with the various filter operations once you’ve completed the core tasks and are entirely optional.

Objectives

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

- write a query that uses an equality filter.
- write a query that uses a comparison filter.
- write a query that uses the IN and BETWEEN operators.
- write a query statement that looks for text within a column.
- write a query that correctly identifies NULL values.

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 WHERE clauses

Northwind Traders are attempting to learn more about their currently stocked beverages. You have been asked to produce a list of current products that are in that category and that have a high value.

In this exercise, you will use basic WHERE 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 with a CategoryID of 1 (beverages).
3. Modify the query to include an additional filter that removes Products with a unit price less than 40.

Task 1: Write a query that only returns current products

1. Create a new query and save it with a name of "CurrentBeverageProducts.sql".
2. Write a query that uses the Northwind database and displays the ProductID, ProductName, CategoryID, Discontinued and UnitPrice columns from the Products table.
3. Execute the query and verify that it returns 77 rows.
4. Add a WHERE clause to the query that asks for only those products with a Discontinued value of 0 (zero).
5. Execute the query and verify that it returns only 69 rows.

Task 2: Write a query that only returns current products in category 1

1. Modify your existing query.
2. Add an additional statement to the WHERE clause that further limits the results to products with a CategoryID of 1.
3. Execute the query and verify that it now returns only 11 rows.

Task 3: Write a query that only returns expensive, current products in category 1

1. Modify your existing query.
2. Add an additional statement to the WHERE that further limits the results to products with a unit price greater than or equal to 40.
3. Execute the query and verify that it now retrieves just 2 products – Cote de Blaye and Ipoh Coffee.

Exercise 2: Using IN and BETWEEN

Northwind Traders have had some complaints about orders that were placed in a certain month for certain customers. You have been asked to produce a query of all the orders that were placed by those customers in that month.

In this exercise, you will use the IN and BETWEEN operators.

The main tasks for this exercise are as follows:

1. Write a query that selects only the rows in the Orders table which were placed by either Alfreds Futterkiste (CustomerID 'ALFKI'), Ernst Handle ('ERNSH') or Simon's bistro ('SIMOB').
2. Modify the query so that it only returns orders for those customers placed between the 1st August and the 31st August 1997.
3. Ensure that the query uses IN and BETWEEN operators.

Task 1: Write a query that only returns orders for some customers

1. Create a new query and save it with a name of "CustomerComplaints.sql".
2. Write a query that uses the Northwind database and displays the OrderID, CustomerID and OrderDate columns from the Orders table.
3. Limit the results to only those orders with a CustomerID of either 'ALFKI', 'ERNSH' or 'SIMOB'.
4. Remember that string literals need to be enclosed in 'single quotes'.
5. Execute the query and verify that it returns 43 rows.

Task 2: Write a query that returns orders for some customers in August 1997

1. Modify your existing query.
2. Further limit the results of the query so that it only returns orders with an OrderDate greater than or equal to the 1st August 1997 and an OrderDate of less than or equal to the 31st August 1997.
3. Remember that date literals, like string literals, need to be enclosed in 'single quotes' ideally using the format 'YYYYMMDD'.

4. Execute the query and verify that there are now just 3 rows – OrderIDs 10633, 10642 and 10643.
5. Make sure the query returns the three rows we want. 37 rows and 73 rows are wrong for different reasons.

Task 3: Ensure that the query uses IN and BETWEEN

1. Modify your existing query.
2. If it doesn't already, ensure that the query uses IN for the CustomerIDs and BETWEEN for the OrderDates.
3. Notice that the query is much easier to read and doesn't require so many parentheses.
4. Execute the query and verify that it still returns 3 rows.

Exercise 3: String comparisons

Northwind Traders Marketing department has asked you to produce a list of all your customer contacts who are involved in the sales function.

In this exercise, you will use the LIKE operator and wildcard characters.

The main tasks for this exercise are as follows:

1. Write a query that selects rows from the Customers table where the Contact's job description starts with the word 'Sales'.
2. Modify the query so that it only returns contact details for anyone with the word 'sales' anywhere in their job description.
3. Experiment with case sensitivity.

Task 1: Write a query that only returns contact details for some customers

1. Create a new query and save it with a name of "SalesContacts.sql".
2. Write a query that uses the Northwind database and selects the CustomerID, CompanyName, ContactName, ContactTitle and Phone columns from the Customers table.
3. Execute the query and verify that it returns 91 rows.
4. Now add a WHERE clause which looks for a ContactTitle that begins with the word 'sales'.
5. You should retrieve 40 rows.

Task 2: Use leading and trailing wildcards

1. Modify the existing query.
2. Modify the LIKE comparison so that it looks for Contacts where the word 'sales' appears anywhere in their ContactTitle column.
3. You're expecting 43 rows now – there are 3 customer contacts whose job title is "Assistant Sales Agent".

Task 3: Investigate case sensitivity in SQL Server

1. Modify the existing query.
2. Try looking for SALES, or Sales, or sales, or even SaLeS. You should still see the same 43 rows – by default, SQL Server is configured to be case insensitive.

Exercise 4: NULL comparisons

Northwind Traders Marketing department wants to update all their contact details to ensure that they have fax numbers for everyone. You have been asked to produce a list of all your customer contacts that don't have a Fax number listed.

In this exercise, you will use the IS NULL operator.

The main tasks for this exercise are as follows:

1. Write a query that selects rows from the Customers table where the Fax column has a *NULL* value.

Task 1: Write a query that only lists customers with no fax number listed

1. Create a new query and save it with a name of "FaxlessContacts.sql".
2. Write a query that uses the Northwind database and selects the CustomerID, CompanyName, ContactName, ContactTitle, Phone and Fax columns from the Customers table.
3. Execute the query and verify that it returns 91 rows.
4. Now add a WHERE clause which looks for *null* values in the Fax column.
5. You should retrieve 22 rows.

Optional extras

1. Modify the “CurrentBeverageProducts.sql” expensive products query from Exercise 1 so that it looks for current products in both category 1 and category 8.
2. Modify the “CustomerComplaints.sql” problem orders query from Exercise 2 so that it looks at a bigger range of dates but also limits the results to only certain EmployeeIDs (say 1,3,5,7 and 9).
3. Try using an “= NULL” comparison instead of an IS NULL in the query from exercise 4. This should return no rows.

Learn. To Change.

QA.com