


FRESHER ACADEMY

SQL

TrungDVQ@fsoft.com.vn

7/20/2020

[Type the abstract of the document here. The abstract is typically a short summary of the contents of the document. Type the abstract of the document here. The abstract is typically a short summary of the contents of the document.]

	<p style="text-align: center;">CODE:</p> <p style="text-align: center;"><u>PRACTICE_ADVENTUREWORKS2008</u></p> <p>Assignment topic : SQL Practice</p> <p>Assignment duration : N/A</p>	<p style="text-align: center;">FRESHER ACADEMY</p>
---	---	---

I. SELECT Queries

Exercise 1.1: Using the SELECT Statement

Use the **AdventureWorksLT2008** database to complete this exercise.

1. Write a SELECT statement that lists the **customers** along with their ID numbers. Include the lastnames, first names, and company names.
2. Write a SELECT statement that lists the name, product number, and color of each **product**.
3. Write a SELECT statement that lists the customer ID numbers and sales order ID numbers from the **SalesLT.SalesOrderHeader** table.

Exercise 1.2: Filtering Data

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query using a WHERE clause that displays all the employees listed in the **HumanResources.Employee** table who have the job title **Research and Development Engineer**.
Display the business entity ID number, the login ID, and the title for each one.
2. Write a query using a WHERE clause that displays all the names in Person.Person with the middle name J. Display the first, last, and middle names along with the ID numbers.
3. Write a query displaying all the columns of the Production.ProductCostHistory table from the rows that were modified on June 17, 2003. Be sure to use one of the features in SQL Server Management Studio to help you write this query. In SQL Server Management Studio, expand the AdventureWorks2008 database. Expand Tables. Right-click the Production.ProductCostHistory table, and choose "Select table as." Select "Select to" and New Query Editor Window. Then type in the WHERE clause.
4. Rewrite the query you wrote in question 1, changing it so that the employees who do not have the title **Research and Development Engineer** are displayed.

5. Write a query that displays all the rows from the `Person.Person` table where the rows were modified after December 29, 2000. Display the business entity ID number, the name columns, and the modified date.
6. Rewrite the last query so that the rows that were not modified on December 29, 2000, are displayed.
7. Rewrite the query from question 5 so that it displays the rows modified during December 2000.
8. Rewrite the query from question 5 so that it displays the rows that were not modified during December 2000.

Exercise 1.3: Filtering with Wildcards

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query that displays the product ID and name for each product from the `Production.Product` table with the name starting with *Chain*.
2. Write a query like the one in question 1 that displays the products with *helmet* in the name.
3. Change the last query so that the products without *helmet* in the name are displayed.
4. Write a query that displays the business entity ID number, first name, middle name, and last name from the `Person.Person` table for only those rows that have *E* or *B* stored in the middle name column.
5. Explain the difference between the following two queries:

```
SELECT FirstName  
FROM Person.Person  
WHERE LastName LIKE 'Ja%es';
```

```
SELECT FirstName  
FROM Person.Person  
WHERE LastName LIKE 'Ja_es';
```

Exercise 1.4: Filtering with Multiple Predicates

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query displaying the order ID, order date, and total due from the Sales.SalesOrderHeader table. Retrieve only those rows where the order was placed during the month of September 2001 and the total due exceeded \$1,000.
2. Change the query in question 1 so that only the dates September 1–3, 2001, are retrieved. See whether you can figure out three different ways to write this query.
3. Write a query displaying the sales orders where the total due exceeds \$1,000. Retrieve only those rows where the salesperson ID is 279 or the territory ID is 6.
4. Change the query in question 3 so that territory 4 is included.
5. Explain when it makes sense to use the IN operator.

Exercise 1.5: Working with Nothing

Use the **AdventureWorks2008** database to complete this exercise. *Make sure you consider how NULL values will affect your results.*

1. Write a query displaying the ProductID, Name, and Color columns from rows in the Production.Product table. Display only those rows where no color has been assigned.
2. Write a query displaying the ProductID, Name, and Color columns from rows in the Production.Product table. Display only those rows in which the color is not blue.
3. Write a query displaying ProductID, Name, Style, Size, and Color from the Production.Product table. Include only those rows where at least one of the Style, Size, or Color columns contains a value.

Exercise 1.6: Performing a Full-Text Search

Use the **AdventureWorks2008** database to complete this exercise. *Be sure to take advantage of the full-text indexes in place when writing the queries.*

1. Write a query using the Production.ProductReview table. Use CONTAINS to find all the rows that have the word *socks* in the Comments column. Return the ProductID and Comments columns.
2. Write a query using the Production.Document table. Use CONTAINS to find all the rows that have the word *reflector* in any column that is indexed with Full-Text Search. Display the Title and FileName columns.
3. Change the query in question 2 so that the rows containing *seat* are not returned in the results.

Exercise 1.7: Writing Expressions Using Operators

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query that displays in the “AddressLine1 (City PostalCode)” format from the Person.Address table.
2. Write a query using the Production.Product table displaying the product ID, color, and name columns. If the color column contains a NULL value, replace the color with *No Color*.
3. Modify the query written in question 2 so that the description of the product is displayed in the “Name: Color” format. Make sure that all rows display a value even if the Color value is missing.
4. Write a query using the Production.Product table displaying a description with the “ProductID: Name” format.

Hint: You will need to use a function to write this query.

Exercise 1.7: Using Mathematical Operators

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query using the Sales.SpecialOffer table. Display the difference between the MinQty and MaxQty columns along with the SpecialOfferID and Description columns.

2. Write a query using the Sales.SpecialOffer table. Multiply the MinQty column by the DiscountPct column. Include the SpecialOfferID and Description columns in the results.
3. Write a query using the Sales.SpecialOffer table that multiplies the MaxQty column by the DiscountPCT column. If the MaxQty value is null, replace it with the value 10. Include the SpecialOfferID and Description columns in the results.
4. Describe the difference between division and modulo.

Exercise 1.8: Using String Functions

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query that displays the first 10 characters of the AddressLine1 column in the Person.Address table.
2. Write a query that displays characters 10 to 15 of the AddressLine1 column in the Person.Address table.
3. Write a query displaying the first name and last name from the Person.Person table all in uppercase.

Exercise 1-9: Using Date Functions

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query that calculates the number of days between the date an order was placed and the date that it was shipped using the Sales.SalesOrderHeader table. Include the SalesOrderID, OrderDate, and ShipDate columns.
2. Write a query that displays only the date, not the time, for the order date and ship date in the Sales.SalesOrderHeader table.
3. Write a query that adds six months to each order date in the Sales.SalesOrderHeader table.
Include the SalesOrderID and OrderDate columns.
4. Write a query that displays the year of each order date and the numeric month of each order date in separate columns in the results. Include the SalesOrderID and OrderDate columns.
5. Change the query written in question 4 to display the month name instead.

Exercise 1-10: Using Mathematical Functions

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query using the Sales.SalesOrderHeader table that displays the SubTotal rounded to two decimal places. Include the SalesOrderID column in the results.
2. Modify the query in question 1 so that the SubTotal is rounded to the nearest dollar but still displays two zeros to the right of the decimal place.
3. Write a query that calculates the square root of the SalesOrderID value from the Sales.SalesOrderHeader table.
4. Write a statement that generates a random number between 1 and 10 each time it is run.

Exercise 1.11: Using Functions in the WHERE and ORDER BY Clauses

Use the **AdventureWorks2008** database to complete this exercise.

1. Write a query using the Sales.SalesOrderHeader table to display the orders placed during 2001 by using a function. Include the SalesOrderID and OrderDate columns in the results.
2. Write a query using the Sales.SalesOrderHeader table listing the sales in order of the month the order was placed and then the year the order was placed. Include the SalesOrderID and OrderDate columns in the results.
3. Write a query that displays the PersonType and the name columns from the Person.Person table. Sort the results so that rows with a PersonType of IN, SP, or SC sort by LastName. The other rows should sort by FirstName. Hint: Use the CASE function.