

70-761^{Q&As}

Querying Data with Transact-SQL

Pass Microsoft 70-761 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.geekcert.com/70-761.html

100% Passing Guarantee 100% Money Back Assurance

Following Questions and Answers are all new published by Microsoft Official Exam Center

Instant Download After Purchase

100% Money Back Guarantee

- 😳 365 Days Free Update
- 800,000+ Satisfied Customers





QUESTION 1

DRAG DROP

Note: This question is part of a series of questions that use the same scenario. For your convenience, the scenario is repeated in each question. Each question presents a different goal and answer choices, but the text of the scenario is exactly the same in each question in this series.

Start of repeated scenario

You have a database that contains the tables shown in the exhibit. (Click the Exhibit button.)

| a | lesSummary | | | E | Imployee | | 的研究的智慧 |
|---|-----------------|-----------|-------------|-----|--------------|-------------|-------------|
| | Column Name | Data Type | Allow Nulls | | Column Name | Data Type | Allow Nulls |
| 8 | SalesSummaryKey | int | | -C+ | 8 Employeel | smallint | |
| | SalesYear | smallint | | | EmployeeCode | char(6) | |
| | SalesQuarter | smallint | | | Firstwame | varchar(30) | |
| | SalesMonth | smallint | | 0 | MiddleName | varchar(30) | \square |
| | SalesDate | date | | 6 | LastName | varchar(40) | |
| | ProductCode | char(12) | | 5 | Title | varchar(50) | |
| | CustomerCode | char(6) | | | ManagerID | smallint | \square |
| | EmployeeCode | char(б) | allo | | | | |
| | RegionCode | char(2) | NO A | | | | |
| | SalesAmount | money < | | | | | |

You review the Employee table and make the following observations:

Every record has a value in the ManagerID except for the Chief Executive Officer (CEO).

The FirstName and MiddleName columns contain null values for some records.

The valid values for the Title column are Sales Representative manager, and CEO.

You review the SalesSummary table and make the following observations:

The ProductCode column contains two parts: The first five digits represent a product code, and the last seven digits represent the unit price. The unit price uses the following pattern: ####.##.

You observe that for many records, the unit price portion of the ProductCode column contains values.

The RegionCode column contains NULL for some records.

Sales data is only recorded for sales representatives.

You are developing a series of reports and procedures to support the business. Details for each report or procedure follow.

Sales Summary report: This report aggregates data by year and quarter. The report must resemble the following table.



| SalesYear | SalesQuarter | YearSalesAmount | QuarterSalesAmount |
|-----------|--------------|-----------------|--------------------|
| 2015 | 1 | 2000.00 | 1000.00 |
| 2015 | 2 | 2000.00 | 500.00 |
| 2015 | 3 | 2000.00 | 250.00 |
| 2015 | 4 | 2000.00 | 250.00 |
| 2016 | 1 | 3500.00 | 500.00 |
| 2016 | 2 | 3500.00 | 1000.00 |

Sales Manager report: This report lists each sales manager and the total sales amount for all employees that report to the sales manager.

Sales by Region report: This report lists the total sales amount by employee and by region. The report must include the following columns: EmployeeCode, MiddleName, LastName, RegionCode, and SalesAmount. If MiddleName is NULL, FirstName must be displayed. If both FirstName and MiddleName have null values, the world Unknown must be displayed/ If RegionCode is NULL, the word Unknown must be displayed.

Report1: This report joins data from SalesSummary with the Employee table and other tables. You plan to create an object to support Report1. The object has the following requirements:

be joinable with the SELECT statement that supplies data for the report

can be used multiple times with the SELECT statement for the report

be usable only with the SELECT statement for the report

not be saved as a permanent object

Report2: This report joins data from SalesSummary with the Employee table and other tables. You plan to create an object to support Report1. The object has the following requirements:

be joinable with the SELECT statement that supplies data for the report

can be used multiple times for this report and other reports

accept parameters

be saved as a permanent object

Sales Hierarchy report: This report aggregates rows, creates subtotal rows, and super-aggregates rows over the SalesAmount column in a single result-set. The report uses SaleYear, SaleQuarter, and SaleMonth as a hierarchy. The result set must not contain a grand total or cross-tabulation aggregate rows.

Current Price Stored Procedure: This stored procedure must return the unit price for a product when a product code is supplied. The unit price must include a dollar sign at the beginning. In addition, the unit price must contain a comma every three digits to the left of the decimal point, and must display two digits to the left of the decimal point. The stored procedure must not throw errors, even if the product code contains invalid data.

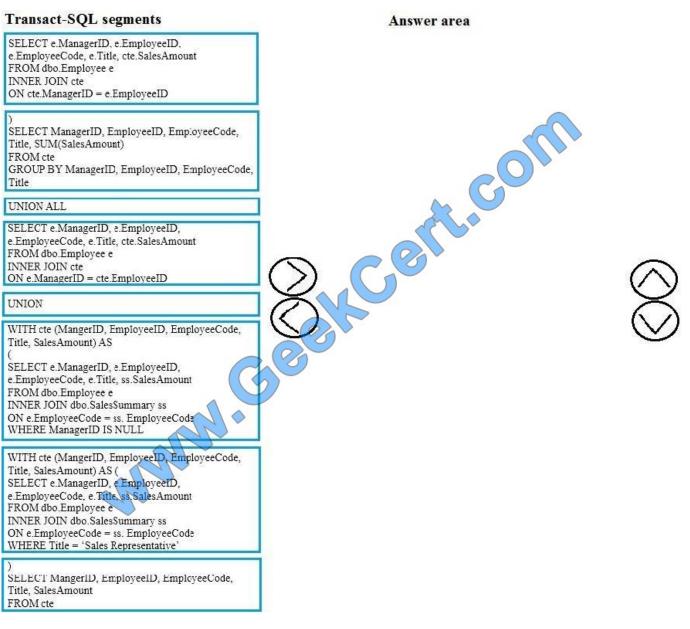
End of Repeated Scenario

You need to create the query for the Sales Managers report.

Which four Transact-SQL segments should you use to develop the solution? To answer, move the appropriate Transact-SQL segments from the list of Transact-SQL segments to the answer area and arrange them in the correct order.



Select and Place:



Correct Answer:



Transact-SQL segments Answer area SELECT e.ManagerID, e.EmployeeID, e.EmployeeCode, e.Title, cte.SalesAmount WITH cte (MangerID, EmployeeID, EmployeeCode, FROM dbo.Employee e Title, SalesAmount) AS (INNER JOIN cte SELECT e.ManagerID, e.EmployeeID, ON cte.ManagerID = e.EmployeeID e.EmployeeCode, e.Title, ss.SalesAmount, FROM dbo.Employee e SELECT ManagerID, EmployeeID, EmployeeCode, Title, SUM(SalesAmount) INNER JOIN dbo.SalesSummary ss FROM cte ON e.EmployeeCode = ss. EmployeeCode GROUP BY ManagerID, EmployeeID, EmployeeCode, WHERE Title = 'Sales Representative Title UNION ALL SELECT ManagerID, EmployeeID, EmployeeCode, SELECT e ManagerID, e EmployeeID, e.EmployeeCode, e.Title, cte.SalesAmount Title, SUM(SalesAmount) FROM dbo.Employee e FROM cte INNER JOIN cte ON e.ManagerID = cte.EmployeeID GROUR BY ManagerID, EmployeeID, EmployeeCode, Title UNION WITH cte (MangerID, EmployeeID, EmployeeCode, INION Title, SalesAmount) AS SELECT e.ManagerID, e.EmployeeID, SELECT e.ManagerID, e.EmployeeID, e.EmployeeCode, e.Title, ss.SalesAmount FROM dbo.Employee e e.EmployeeCode, e.Title, cte.SalesAmount INNER JOIN dbo.SalesSummary ss FROM dbo.Employee e ON e.EmployeeCode = ss. EmployeeCode INNER JOIN cte WHERE ManagerID IS NULL ON e.ManagerID = cte.EmployeeID WITH cte (MangerID, EmployeeID, EmployeeCode, Title, SalesAmount) AS (SELECT e ManagerID, EmployeeID, e.EmployeeCode, e.Title, ss.SalesAmount FROM dbo.Employee e INNER JOIN dbo.SalesSummary ss ON e.EmployeeCode = ss. EmployeeCode WHERE Title = 'Sales Representative SELECT MangerID, EmployeeID, EmployeeCode, Title, SalesAmount

FROM cte

From scenario: Sales Manager report: This report lists each sales manager and the total sales amount for all employees that report to the sales manager. Box 1:..WHERE Title=\\'Sales representative\\'

The valid values for the Title column are Sales Representative manager, and CEO.

First we define the CTE expression.

Note: A common table expression (CTE) can be thought of as a temporary result set that is defined within the execution scope of a single SELECT, INSERT, UPDATE, DELETE, or CREATE VIEW statement. A CTE is similar to a derived

table in that it is not stored as an object and lasts only for the duration of the query. Unlike a derived table, a CTE can be self-referencing and can be referenced multiple times in the same query.

Box 2:

Use the CTE expression one time.

Box 3: UNION



Box 4:

Use the CTE expression a second time.

References: https://technet.microsoft.com/en-us/library/ms190766(v=sql.105).aspx

QUESTION 2

You have a database that includes the tables shown in the exhibit. (Click the exhibit button.)

| Customer (Sales) | SalesOrderHeader (Sales |
|--------------------|---|
| CustomerID | SalesOrderID |
| PersonID | RevisionNumber |
| StoreID | OrderDate |
| TerritoryID | DueDate |
| IN O DIDISOLA MANY | ShipDate |
| AccountNumber | Status |
| rowguid | OnineOrderFlag SalesOrderNumber |
| ModifiedDate | PutchaseOrderNumber |
| | AccountNumber |
| | CustomerID |
| | SalesPersonID |
| | |
| | TerritoryID |
| | TerritoryID BillToAddressID |
| | |
| | BillToAddressID ShipToAddressID |
| | BillToAddressID ShipToAddressID ShipMethodID CreditCardID |
| | BillToAddressID ShipToAddressID ShipMethodID CreditCardID CreditCardApprovalCode |
| | BillToAddressID ShipToAddressID ShipMethodID CreditCardID CreditCardApprovalCode CurrencyRateID |
| and. | BillToAddressID ShipToAddressID ShipMethodID CreditCardID CreditCardApprovalCode CurrencyRateID SubTotal |
| mm | BillToAddressID ShipToAddressID ShipMethodID CreditCardID CreditCardApprovalCode CurrencyRateID SubTotal TaxAmt |
| mm | BillToAddressID ShipToAddressID ShipMethodID CreditCardID CreditCardApprovalCode CurrencyRateID SubTotal TaxAmt Freight |
| MMM | BillToAddressID ShipToAddressID ShipMethodID CreditCardID CreditCardApprovalCode CurrencyRateID SubTotal TaxAmt Freight TotalDue |
| MMM | BillToAddressID ShipToAddressID ShipMethodID CreditCardID CreditCardApprovalCode CurrencyRateID SubTotal TaxAmt Freight |

You need to create a list of all customers and the date that the customer placed their last order. For customers who have not placed orders, you must substitute a zero for the order ID and 01/01/1990 for the date. Which Transact-SQL statement should you run?



| FR ON GR | LECT C.CustomerID, COALESCE(MAX(OrderDate), '19000101') OM Sales.Customer C LEFT OUTER JOIN Sales.SalesOrderHeader SOH C.CustomerID = SOH.CustomerID OUP BY C.CustomerID DER BY C.CustomerID |
|----------------|---|
| A. | SELECT C.CustomerID, COALESCE(MAX(OrderDate), '19000101') FROM Sales.Customer C LEFT OUTER JOIN Sales.SalesOrderHeader SOH ON C.CustomerID = SOH.CustomerID GROUP BY C.CustomerID ORDER BY C.CustomerID |
| В. | SELECT C.CustomerID, MAX(OrderDate) FROM Sales.Customer C LEFT OUTER JOIN Sales.SalesOrderHeader SOH ON C.CustomerID = SOH.CustomerID ORDER BY C.CustomerID |
| | SELECT C.CustomerID, MAX(OrderDate) FROM Sales.Customer C CROSS JOIN Sales.SalesOrderHeader SOH GROUP BY C.CustomerID ORDER BY C.CustomerID |
| | SELECT C.CustomerID, SOH.SalesOrderID, MAX(OrderDate) FROM Sales.Customer C LEFT OUTER JOIN Sales.SalesOrderHeader SOH ON C.CustomerID = SOH.CustomerID GROUP BY C.CustomerID, SOH.SalesOrderID ORDER BY C.CustomerID |

A. B. C. D.

Correct Answer: A

COALESCE evaluates the arguments in order and returns the current value of the first expression that initially does not evaluate to NULL. References: https://docs.microsoft.com/en-us/sql/t-sql/language-elements/coalesce-transact-sql

QUESTION 3

You have a table named Table1 that contains 200 million rows. Table1 contains a column named SaleDate that has a data type of DateTime2(3). Users report that the following query runs slowly.

Select SalesPerson, count(*) FROM table1 Where year (SaleDate) = 2017 GROUP BY SalesPerson



You need to reduce the amount of time it takes to run the query.

What should you use to replace the WHERE statement?

- A. WHERE SaleDate >= \\'2017-01-01\\' AND SaleDate
- B. WHERE cast(SaleDate as varchar(10)) BETWEEN \\'2017-01-01\\' AND \\'2017-12-31\\'
- C. WHERE cast(SaleDate as date) BETWEEN \\'2017-01-01\\' AND \\'2017-12-31\\'
- D. WHERE 2017 = year(SaleDate)

Correct Answer: C

References: https://docs.microsoft.com/en-us/sql/t-sql/queries/select-transact-sql?view=sql-server-2017

QUESTION 4

CREATE TABLE CourseParticipants(CourseID INT NOT NULL, CourseDate DATE NOT NULL, LocationDescription VARCHAR(100) NOT NULL, NumParticipants INT NOT NULL)

You are a developing a solution to manage employee training records. You have the following Transact-SQL statement:

You need to create a stored procedure that returns the total number of participants for a specific course.

How should you complete the procedure? To answer, drag the appropriate Transact-SQL segments to the correct locations. Each Transact-SQL segment may be used once, more than once, or not at all. You may need to drag the split bar

between panes or scroll to view content.

NOTE: Each correct selection is worth one point.

Select and Place:



Transact-SQL segments Answer Area CREATE PROCEDURE uspGetParticipantsPerCourse Gase KCerte BIGINT @courseID BIGINT, @totalParticipants INT INPUT INT OUTPUT SET NOCOUT ON; SELECT @totalParticipants = SUM(NumParticipants) INPUT FROM CourseParticipants **GROUP BY CourseID** HAVING CourseID = RETURN GO Correct Answer: Transact-SQL segments Answer Area CREATE PROCEDURE uspGetParticipantsPerCourse BIGINT acourseID @courseID BIGINT, @totalParticipants @totalParticipants INT OUTPUT INT INPUT (A) INT OUTPUT SET NOCOUT ON; SELECT @totalParticipants = SUM(NumParticipants) INPUT FROM CourseParticipants **GROUP BY CourseID** acourseID HAVING CourseID = RETURN

GO



QUESTION 5

DRAG DROP

Note: This question is part of a series of questions that use the same scenario. For your convenience, the scenario is repeated in each question. Each question presents a different goal and answer choices, but the text of the scenario is

exactly the same in each question in this series.

You are developing a database to track customer orders. The database contains the following tables: Sales.Customers, Sales.Orders, and Sales.OrderLines. The following table describes the columns in Sales.Customers.

| Column name | Data type | Constraints |
|----------------------------|-------------------|-------------------------------|
| CustomerID | int | primary key |
| CustomerName | nvarchar(100) 🛛 🕜 | or does not allow null values |
| PhoneNumber | nvarchar(20) 🔊 | does not allow null values |
| AccountOpenedDate | date √ 🍼 | does not allow null values |
| StandardDiscountPercentage | decimat (18,3) | does not allow null values |
| CreditLimit | decimal(18,2) | null values are permitted |
| IsOnCreditHold | dbit | does not allow null values |
| DeliveryLocation | geography | does not allow null values |
| PhoneNumber | nvarchar(20) | does not allow null values |

The following table describes the columns in Sales.Orders.

| Column name | Data type | Constraints |
|-------------|-----------|--|
| OrderID | int | primary key |
| CustomerID | int | foreign key to the Sales.Customers table |
| OrderDate | date | does not allow null values |

The following table describes the columns in Sales.OrderLines.

| Column name | Data type | Constraints |
|-------------|-----------------|---------------------------------------|
| OrderLinelD | int | primary key |
| OrderID | int | foreign key to the Sales.Orders table |
| Quantity | int | does not allow null values |
| UnitPrice | decimal(18,2) 💉 | null values are permitted |
| TaxRate | decimal(18,3) | does not allow null values |

You need to create a function that accepts a CustomerID as a parameter and returns the following information:

all customer information for the customer



the total number of orders for the customer

the total price of all orders for the customer

the average quantity of items per order How should you complete the function definition? To answer, drag the appropriate Transact-SQL segment to the correct locations. Transact-SQL segment may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.

Select and Place:

Transact-SQL segments Answer Area CREATE FUNCTION Sales.GetCustomerInformation(@CustomerID int) COUNT Transact-SQL segment SUM AS RETURN (AVG SELECT C.CustomerName, C.PhoneNumber, C.AccountOpenedDate, C.StardardDiscountPercentage C. CreditLimit, C.IsOnCreditHold, ORDER BY 0 . (O.OrderID) AS TotalNumberOfOrders, Transact-SQL segment ē GROUP BY . Transact-SQL segment (OL.UnitPrice) AS TotalOrderPrice, RETURNS INT RETURNS NULL ON NULL INPUT Transact SOL segment (OL.Quantity) AS AverageOrderQuantity RETURNS TABLE FROM Sales.Customers C NN Sales.Orders AS 0 ON O.CustomerID = C.CustomerID OIN Sales.OrderLines AS OL ON OL.OrderID = 0.OrderID HERE C.CustomerID = @CustomerID Transact-SQL segment C.CustomerName, C.PhoneNumber, C.AccountOpenedDate, C.StandardDiscountPercentage, C.CreditLimit, C.IsOnCreditHold

Correct Answer:

Transact-SQL segments

Answer Area

| | | CREATE FUNCTION Sales.GetCustomerInformation(@CustomerID int) RETURNS TABLE |
|----------------------------|---|--|
| ORDER BY | 0 | AS RETURN (SELECT C.CustomerName, C.PhopeNumber, C.AccountOpenedDate, C.StardardDiscountPercentage, C.CreditLimit, C.IsOnCreditHold, |
| RETURNS INT | • | SUM (OL.UnitPrice) AS TotalOrderPrice, |
| RETURNS NULL ON NULL INPUT | ! | AVG (OL.Quantity) AS AverageOrderQuantity |
| | | JOIN Sales.Orders AS 0 ON O.CustomerID = C.CustomerID JOIN Sales.OrderLines AS OL ON OL.OrderID = O.OrderID WHERE C.CustomerID = @CustomerID |
| | | GROUP BY C.CustomerName, C.PhoneNumber, C.AccountOpenedDate |
| | | C.StandardDiscountPercentage, C.CreditLimit, C.IsOnCreditHold |



Box1: RETURNS TABLE

The function should return the following information: all customer information for the customer the total number of orders for the customer the total price of all orders for the customer the average quantity of items per order Box 2: COUNT The function should return the total number of orders for the customer. Box 3: SUM The function should return the total price of all orders for the customer. Box 3. AVG The function should return the average quantity of items per order. Box 4: GROUP BY Need to use GROUP BY for the aggregate functions. References: https://msdn.microsoft.com/en-us/library/ms186755.aspx

70-761 PDF Dumps

70-761 Practice Test

70-761 Braindumps



To Read the Whole Q&As, please purchase the Complete Version from Our website.

Try our product !

100% Guaranteed Success
100% Money Back Guarantee
365 Days Free Update
Instant Download After Purchase
24x7 Customer Support
Average 99.9% Success Rate
More than 800,000 Satisfied Customers Worldwide
Multi-Platform capabilities - Windows, Mac, Android, iPhone, iPod, iPad, Kindle

We provide exam PDF and VCE of Cisco, Microsoft, IBM, CompTIA, Oracle and other IT Certifications. You can view Vendor list of All Certification Exams offered:

https://www.geekcert.com/allproducts

Need Help

Please provide as much detail as possible so we can best assist you. To update a previously submitted ticket:



One Year Free Update



Free update is available within One Year after your purchase. After One Year, you will get 50% discounts for updating. And we are proud to boast a 24/7 efficient Customer Support system via Email.



To ensure that you are spending on quality products, we provide 100% money back guarantee for 30 days

Money Back Guarantee

from the date of purchase



Security & Privacy

We respect customer privacy. We use McAfee's security service to provide you with utmost security for your personal information & peace of mind.

Any charges made through this site will appear as Global Simulators Limited. All trademarks are the property of their respective owners. Copyright © geekcert, All Rights Reserved.